

**PLAN DE ESTUDIOS 2011
ESPECIALIDAD DE
CIENCIA DE LA COMPUTACION**

PRIMER CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CM131	Cálculo Diferencial	4	2	--	--	--	06	5	Ninguno
CM141	Cálculo Vectorial I	4	2	--	--	--	06	5	Ninguno
CQ111	Química I	4	(3)	(3)	--	--	07	5	Ninguno
CF121	Física I	4	(3)	(3)	--	--	07	5	Ninguno
CC101	Introducción a la Ciencia de la Computación	2	--	--	--	--	02	2	Ninguno
Total							28	22	

SEGUNDO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CM132	Cálculo Integral	4	2	--	--	--	06	5	CM131
CM142	Cálculo Vectorial II	4	2	--	--	--	06	5	CM141
CQ112	Química II	4	(3)	(3)	--	--	07	5	CQ111
CF122	Física II	4	(3)	(3)	--	--	07	5	CF121
CC102	Introducción a la Programación	1		2	1	--	04	2	CC101
Total							30	22	

TERCER CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CF221	Física III	4	(3)	(3)	--	--	07	5	CF122, CM132, CM142, CQ112
CM254	Introducción a la Matemática Discreta	2	(2)	--	(2)	--	04	3	CM132, CM142
CM261	Álgebra Lineal I	4	2	--	--	--	06	5	CM132, CM142
CC201	Introducción a la Programación Orientada a Objetos	3	--	3	--	--	06	4	CC102
AHD65	Constitución y Derechos Humanos	2	--	--	--	--	02	2	Ninguno
CL002	Inglés I	1	--	3	--	--	04	2	Ninguno
Total							29	21	

CUARTO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CM211	Calculo Diferencial e Integral Avanzado	6	3	--	--	--	09	7	CM132, CM142
CM094	Estructura de datos	4	--	2	--	--	06	5	CM254, CM261

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

CC212	Arquitectura de Computadoras	3	--	3	--	--	06	4	CM254, CC102
CC262	Algoritmos	3	(3)	(3)	--	--	06	4	CM254, CC102
CL003	Inglés II	1	--	3	--	--	04	2	CL002
Total							31	22	

QUINTO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CM274	Introducción a la Estadística y Probabilidades	4	(2)	(2)	--	--	06	5	CM211
CC301	Algoritmos Paralelos	3	(3)	(3)	--	--	06	4	CM094, CC262
CC331	Base de Datos	3	--	3	--	--	06	4	CM094, CC262
CC361	Sistemas Operativos	3	--	3	--	--	06	4	CC212
CH061	Biología	2	(2)	(2)	--	--	04	3	CQ112, CF122, CC102
Total							28	20	

SEXTO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CM334	Análisis Numérico I	4	(2)	(2)			06	5	CM211, CM261
CC302	Lenguaje Paralelo Distribuido Orientado a Objetos	3	--	3	--	--	06	4	CC201, CC301
CC322	Computación Gráfica	3	--	3	--	--	06	4	CC301
CC342	Teoría de la Computación	3	--	3	--	--	06	4	CC301, CC361
CC362	Computación Centrada en Redes	3		3			06	4	CC361
Total							30	21	

SEPTIMO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CC401	Programación de Aplicaciones en Redes	3	--	3	--	--	06	4	CC302, CC362
CC441	Inteligencia Artificial	3	--	3	--	--	06	4	CM274, CC301
CC461	Compiladores	3	--	3	--	--	06	4	CC342, CC302
CC471	Biología Computacional	3	2	2			06	4	CH061, CC302
CC481	Administración de Redes	3	--	3	--	--	06	4	CC362
Total							30	20	

OCTAVO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CC462	Sistemas Concurrentes y Distribuidos	3		3			06	4	CC401
CC472	Ingeniería de Software I	3		3			06	4	CC302

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

CC482	Núcleo y Redes para Computación Paralela	1	--	3	--	--	04	2	CC461, CC481
CH007	Ciencia, Tecnología y Sociedad Formación General Electivos	2	1	--	--	--	03	2	Ninguno
Total								19	12

NOVENO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CC541	Seminario de Tesis I	--	--	--	8	--	08	4	Aprobar 8 ciclos
CC581	Introducción a la Computación Grid y Cloud	3	--	3	--	--	06	4	CC462
CC571	Sistemas Operativos Avanzados Electivos	1	--	3	--	--	04	2	CC401, CC482
Total								18	10

DECIMO CICLO

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CC542	Seminario de Tesis II	--	--	--	8	--	08	4	CC541
CC562	Modelamiento y Simulación Electivos	3	--	3	--	--	06	4	CC441, CC201, CM334
Total								14	8

CURSOS ELECTIVOS

LOS CURSOS ELECTIVOS TIENEN COMO PRE-REQUISITO GENERAL HABER APROBADO SEIS CICLOS DE ESTUDIOS (Art. 209 del Estatuto de la UNI)

CURSOS ELECTIVOS DE ESPECIALIDAD

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CC001	Tópicos Especiales de Ciencia de la Computación I	2	--	3	--	--	05	3	Ninguno
CC002	Tópicos Especiales de Ciencia de la Computación II	3	--	3	--	--	06	4	Ninguno
CC003	Tópicos Especiales de Ciencia de la Computación III	4	--	3	--	--	07	5	Ninguno
CC011	Estructura e Interpretación de Programas	3	--	2	--	--	05	4	CC201, CC342
CC012	Herramientas para el uso de Software abierto	3	--	3	--	--	06	4	CC102, CC361

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CC021	Tópicos Especiales de Software	3		3			06	4	CC302,CC322, CC331
CC022	Tópicos en Computación Gráfica	3		3			06	4	CC322
CC023	Introducción a Lenguajes Formales	1		2	1		04	2	CC262,CM094
CC032	Interacción Humano Computador	3		3			06	4	CC201, CC302
CC033	Tópicos en Seguridad	3		3			06	4	CC302
CC034	Patrones de Lenguaje Paralelo	1		3			04	2	CC301
CC041	Flujo de Trabajo	3		2			05	4	CC302
CC042	Computación Molecular Biológica	3		3			06	4	CC201
CC043	Tópicos en Robotica	3		3			06	4	CC511
CC044	Arquitectura y Programación de FPGA y VHDL	3		3	--	--	05	4	Ninguno
CC045	Introducción al Compilador Paralelo	3		3			06	4	CC4_1
CC051	Scripts con Sintaxis en Paralelo	3		2			05	4	CC102
CC052	Browser Paralelo	3		3			06	4	CC051
CC053	Lenguaje Funcional Paralelo	3	--	3	--	--	06	4	CC302, CC342
CC054	Motores de Búsqueda	3	--	3	--	--	06	4	CC342
CC055	Introducción a la Robótica	3	--	3	--	--	06	4	CC441, CF221
CC056	Calidad de Software	3		3			06	4	CC472
CC057	Ingeniería Web	3		3			06	4	CC472
CC062	Sistemas Embebidos en Tiempo Real	3		3			06	4	CC201, CC422
CC063	Física Computacional	3		3			06	4	CF221, CC102
CC064	Química Computacional	3		3			06	4	CQ112, CC102
CC065	Computación Bioinspirada	2	2	2			06	4	CC441
CC066	Programación de Dispositivos Móviles	1	2	2			05	3	CC362
CC067	Seguridad en Computación	1	2	2			05	3	CM094, CC262
CC071	Tópicos en Base de Datos	3		3			06	4	CC331
CC081	Formación de Empresas de Base Tecnológica	2	2				04	3	Ninguno

CURSOS ELECTIVOS COMPLEMENTARIOS

Grupo 1: Física, Matemática, Química

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CM321	Introducción a las Ecuaciones Diferenciales Ordinarias	4	2				06	5	CM211,CM261
CM322	Ecuaciones Diferenciales Parciales I	4	2				06	5	CM321
CM361	Introducción a las Estructuras Algebraicas	4	2				06	5	CM261
CM015	Series de Tiempo	4	2				06	5	CM274
CM032	Métodos Numéricos para Ecuaciones Diferenciales Parciales	4	(2)	(2)			06	5	CM334
CM034	Teoría de Simulación	3		2			05	4	CM334
CM038	Métodos Numéricos del Álgebra	4		2			06	5	CM334, CM261
CM054	Métodos Numéricos de Optimización I	4	(2)	(2)			06	5	CM334
CM072	Tópicos de Investigación I	3	3				06	4	Ninguno
CM090	Introducción a los Elementos Finitos	4	2				06	5	CM334, CM261
CM095	Análisis de Algoritmos	4	2				06	5	CM094
CM096	Teoría de Grafos y Algoritmos	4	2				06	5	CM095
CF222	Física IV	4	(3)	(3)	-	--	07	5	CF221
CF252	Métodos Matemáticos para Físicos I	6	4				10	8	CF221, CM211, CM261,CC102
CF381	Introducción a la Física Moderna	4	3				07	5	CF221, CM211, CC102
CF001	Tópicos Especiales de Física I							2	
CF002	Tópicos Especiales de Física II							3	
CF003	Tópicos Especiales de Física III							4	
CF004	Tópicos Especiales de Física IV							5	

Grupo 2: Formación General

Código	Curso	T	P	L	S	E	TH	C	Pre-requisito
CH091	Ética General	3					03	3	Ninguno
CH092	Ética Profesional	3					03	3	Ninguno
CH093	Metodología del Estudio	2	2				04	3	Ninguno
AHL01	Lenguaje y Técnicas de la Comunicación	3					03	3	Ninguno
CH033	Metodología de la Investigación Científica	2	1				03	2	Ninguno
CH046	Gestión Científica y								Ninguno

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

	Tecnológica	2	1	03	2	
CH003	Economía General	2	1	03	2	Ninguno
CL001	Inglés	2	8	10	6	Ninguno

ES OBLIGATORIO LLEVAR POR LO MENOS 04 CRÉDITOS EN CURSOS DE FORMACIÓN GENERAL DE ENTRE LOS QUE FIGURAN EN ESTA TABLA

Grupo 3: CREDITOS DE ACTIVADES NO-LECTIVAS

Actividad	Dedicación	Créditos	Pre-requisito
XP200 Prácticas Pre-profesionales II	300 horas	2	Aprobar 8vo. Ciclo
YA100 Ayudantía Académica I	44 horas	1	Aprobar 7mo. ciclo y curso correspondiente
XA100 Actividades Diversas I		1	Ninguno
- Expociencia	Certificado	0.5	
- Ayudantía Administrativa	22 horas	0.5	
- Asistencia a Congresos	22 horas	0.5	
- Expositor de Congresos	Certificado	0.5	
- Ayudantía de Investigación	44 horas	1.0	
- Deportistas destacados	Certificado	0.5	

TOTAL DE CRÉDITOS EXIGIDOS AL ALUMNO PARA EGRESAR:

item	Créditos	Especificación
01	178	- Créditos de cursos obligatorios
02	28	- Créditos de cursos electivos de los cuales por lo menos 14 deben corresponder a cursos ofrecidos de especialidad y 14 créditos de cursos de Formación General, complementarios u otros de especialidad
03	02	- Créditos por Prácticas Pre-profesionales (XP200, ver grupo 3)
04	01	- Crédito por Ayudantía Académica I (YA100 , ver grupo 3)
05	01	- Crédito por Actividades Diversas I (XA100, ver grupo 3)
Total	210	

SISTEMA DE EVALUACION

Para todos los cursos de la especialidad el sistema de evaluación será el siguiente:

SISTEMA G

Promedio de prácticas y/o laboratorios	Peso 1
Examen Parcial	Peso 1
Examen Final	Peso 1

Como excepción, el curso CC101, del primer año, se registrá por el sistema J de evaluación.

SISTEMA J

Examen Parcial

Peso 1

Examen Final

Peso 1

Los cursos comunes se registrarán por el sistema de evaluación ya establecido.

PLANA DOCENTE 2010-2011

Nº	Nombre	Categoría	Institución del grado	Especialidad	Línea de Investigación
1	Dr. Álvaro Ernesto Cuno Parari	PR TP20h	UFRJ, Brasil	Ingeniería de Sistemas de Información	Computación Gráfica y Diseño de Software
2	Dr. Glen Darío Rodríguez Rafael	PR TP20h	TUT – Japón	Ingeniería Electrónica e Informática	Computación Paralela y Simulación Numérica
3	Dr. Carlos Javier Solano Salinas	AS DE	CBPF, Brasil	Física	Física Computacional
4	Dr. Luis Alberto Navarro Huamaní	AS TC	UFRJ, Brasil	Estadística	Estadística y Computación Evolutiva
5	Dr. Misael León	AS TC	CAB, Argentina	Física	Física Computacional
6	Bch. Adolfo Vicente Chamorro Gomez	JP TC	UNI	Física	Física Computacional

NUCLEO DE CONOCIMIENTOS

1 DS. Matemáticas Discretas (43 horas como mínimo)

- 1.1 DS/Funciones, Relaciones y Conjuntos. (6 horas)
- 1.2 DS/Lógica Básica (10 horas)
- 1.3 DS/Técnicas de Prueba. (12 horas)
- 1.4 DS/Conceptos Básicos de Conteo. (5 horas)
- 1.5 DS/Grafos y Arboles. (4 horas)
- 1.6 DS/Probabilidad Discreta. (6 horas)

2 PF. Fundamentos de Programación (47 horas como mínimo)

- 2.1 PF/Construcciones fundamentales. (9 horas)
- 2.2 PF/Algoritmos y Resolución de Problemas. (6 horas)
- 2.3 PF/Estructuras de Datos. (10 horas)
- 2.4 PF/Recursividad. (4 horas)
- 2.5 PF/Programación Orientada a Eventos. (4 horas)
- 2.6 PF/Orientación a Objetos. (8 horas)
- 2.7 PF/Fundamentos de seguridad de la Información. (4 horas)
- 2.8 PF/Programación segura. (2 horas)

3 AL. Algoritmos y Complejidad (31 horas como mínimo)

- 3.1 AL/Análisis Básico de Algoritmos. (4 horas)
- 3.2 AL/Estrategias Algorítmicas (6 horas)
- 3.3 AL/Algoritmos Fundamentales. (12 horas)
- 3.4 AL/Algoritmos Distribuidos. (3 horas)
- 3.5 AL/Computabilidad Básica (6 horas)
- 3.6 AL/Clases de Complejidad P y NP.
- 3.7 AL/Teoría de Autómatas
- 3.8 AL/Análisis Avanzado de Algoritmos.
- 3.9 AL/Algoritmos Criptográficos.
- 3.10 AL/Algoritmos Geométricos
- 3.11 AL/Algoritmos Paralelos.

4 AR. Arquitectura y Organización (36 horas como mínimo)

- 4.1 AR/Lógica Digital y Representación de Datos. (7 horas)
- 4.2 AR/Arquitectura y Organización de Computadores. (9 horas)
- 4.3 AR/Interfaces y Estrategias de I/O. (3 horas)
- 4.4 AR/Arquitectura de Memoria. (5 horas)
- 4.5 AR/Organización Funcional. (6 horas)
- 4.6 AR/Multiprocesamiento. (6 horas)
- 4.7 AR/Mejoras de Desempeño
- 4.8 AR/Arquitecturas Distribuidas.
- 4.9 AR/Dispositivos.
- 4.10 AR/Tendencias en Computación.

5 OS. Sistemas Operativos (16 horas como mínimo)

- 5.1 OS/Visión General de los Sistemas Operativos. (2 horas)
- 5.2 OS/Principios de los Sistemas Operativos. (2 horas)

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- 5.3 OS/Concurrencia. (6 horas)
- 5.4 OS/Planeamiento y Despacho. (3 horas)
- 5.5 OS/Administración de Memoria. (3 horas)
- 5.6 OS/Administración de Dispositivos.
- 5.7 OS/Seguridad y Protección.
- 5.8 OS/Sistema de Archivos.
- 5.9 OS/Sistemas Empotrados y de Tiempo Real.
- 5.10 OS/Tolerancia a Fallas.
- 5.11 OS/Evaluación del Desempeño de Sistemas.
- 5.12 OS/*Scripting*.
- 5.13 OS/Análisis Forense Digital.
- 5.14 OS/Modelos de seguridad.

6 NC. Computación Centrada en Redes (18 horas como mínimo)

- 6.1 NC/Introducción. (2 horas)
- 6.2 NC/Comunicación de Redes. (7 horas)
- 6.3 NC/Seguridad de Red. (6 horas)
- 6.4 NC/Organización de la Web. (3 horas)
- 6.5 NC/Aplicaciones en redes.
- 6.6 NC/Administración de Redes.
- 6.7 NC/Compresión y Descompresión.
- 6.8 NC/Tecnología de Datos Multimedia.
- 6.9 NC/Computación Móvil e Inalambrica.

7 PL. Lenguajes de Programación (21 horas como mínimo)

- 7.1 PL/Visión General de los Lenguajes de Programación. (2 horas)
- 7.2 PL/Maquinas Virtuales. (1 hora)
- 7.3 PL/Introducción a la Traducción de Lenguajes. (2 horas)
- 7.4 PL/Declaración y Tipos. (3 horas)
- 7.5 PL/Mecanismos de Abstracción. (3 horas)
- 7.6 PL/Programación Orientada a Objetos. (10 horas)
- 7.7 PL/Programación Funcional.
- 7.8 PL/Sistemas de Traducción del Lenguaje.
- 7.9 PL/Tipos.
- 7.10 PL/Semántica de los Lenguajes de Programación.
- 7.11 PL/Diseño de Lenguajes de Programación.

8 HC. Interacción Humano-Computador (8 horas como mínimo)

- 8.1 HC/Fundamentos de la Interacción Hombre-Computador (HCI) (6 horas)
- 8.2 HC/Construcción de Interfaces Gráfica de Usuario. (2 horas)
- 8.3 HC/Evaluación de Software Centrado en el usuario.
- 8.4 HC/Desarrollo de Software Centrado en el Humano.
- 8.5 HC/Diseño de la Interfaz Gráfica de Usuario.
- 8.6 HC/Programación de Interfaces Gráficas de Usuario.
- 8.7 HC/Aspectos de Sistemas de Multimedia.
- 8.8 HC/Aspectos de Colaboración y Comunicación.
- 8.9 Diseño de interacción para nuevos ambientes.
- 8.10 Factores humanos y seguridad.

9 GV. Computación Gráfica y Visual (3 horas como mínimo)

- 9.1 GV/Técnicas Fundamentales en Computación Gráfica y Visual. (2 horas)

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- 9.2 GV/Sistemas Gráficos (1 hora)
- 9.3 GV/Comunicación Gráfica
- 9.4 GV/Modelamiento Geométrico
- 9.5 GV/Rendering Básico
- 9.6 GV/Rendering Avanzado.
- 9.7 GV/Técnicas Avanzadas.
- 9.8 GV/Animación por Computador.
- 9.9 GV/Visualización.
- 9.10 GV/Realidad Virtual.
- 9.11 GV/Visión Computacional.
- 9.12 GV/Geometría Computacional.
- 9.13 GV/Programación de motores de juegos.

10 IS. Sistemas Inteligentes (10 horas como mínimo)

- 10.1 IS/Tópicos Fundamentales en Sistemas Inteligentes. (1 hora)
- 10.2 IS/Búsqueda y Satisfacción de la Restricción. (5 horas)
- 10.3 IS/Razonamiento basado en conocimiento. (4 horas)
- 10.4 IS/Búsqueda Avanzada.
- 10.5 IS/Representación Avanzada del Conocimiento y Razonamiento.
- 10.6 IS/Agentes.
- 10.7 IS/Procesamiento de Lenguaje Natural.
- 10.8 IS/Aprendizaje de Máquina.
- 10.9 IS/Sistemas de Planeamiento.
- 10.10 IS/Robotica.
- 10.11 IS/Percepción.

11 IM. Administración de Información (11 horas como mínimo)

- 11.1 IM/Modelos de Información. (4 horas)
- 11.2 IM/Sistemas de Base de Datos. (3 horas)
- 11.3 IM/Modelamiento de Datos. (4 horas)
- 11.4 IM/Indexación.
- 11.5 IM/Base de Datos Relacionales.
- 11.6 IM/Lenguajes de Consultas de Base de Datos.
- 11.7 IM/Diseño de Bases de Datos Relacionales.
- 11.8 IM/Procesamiento de Transacciones.
- 11.9 IM/Bases de Datos Distribuidas.
- 11.10 IM/Diseño Físico de Bases de Datos.
- 11.11 IM/Minería de Datos.
- 11.12 IM/Almacenamiento y Recuperación de Información.
- 11.13 IM/Hipermedia.
- 11.14 IM/Sistemas Multimedia.
- 11.15 IM/Librerías Digitales.

12 SP. Asuntos Sociales y Profesionales (16 horas como mínimo)

- 12.1 SP/Historia de la Computación. (1 hora)
- 12.2 SP/Contexto Social de la Computación. (3 horas)
- 12.3 SP/Herramientas Analíticas (2 horas)
- 12.4 SP/Ética Profesional. (3 horas)
- 12.5 SP/Riesgos. (2 horas)
- 12.6 SP/Operaciones de seguridad.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- 12.7 SP/Propiedad Intelectual. (3 horas)
- 12.8 SP/Privacidad y Libertades Civiles. (2 horas)
- 12.9 SP/Crimen Informático
- 12.10 SP/Economía en Computación.
- 12.11 SP/Estructuras de Trabajo Filosóficas

13 SE. Ingeniería de Software (29 horas como mínimo)

- 13.1 SE/Diseño de Software. (8 horas)
- 13.2 SE/Usando APIs. (3 horas)
- 13.3 SE/Herramientas y Entornos de Software. (3 horas)
- 13.4 SE/Procesos de Software. (2 horas)
- 13.5 SE/Especificación de Requerimientos. (4 horas)
- 13.6 SE/Validación y verificación de software. (3 horas)
- 13.7 SE/Evolución del Software. (3 horas)
- 13.8 SE/Administración de Proyectos de Software. (3 horas)
- 13.9 SE/Computación Basada en Componentes.
- 13.10 SE/Métodos Formales.
- 13.11 SE/Confiabilidad del Software.
- 13.12 SE/Desarrollo de Sistemas Especializados.
- 13.13 SE/Evaluación de riesgos.
- 13.14 SE/Mejorando la programación: robustez y seguridad.

14 CN. Ciencia Computacional (Sin mínimo de horas)

- 11 CN/Modelamiento y Simulación.
- 12 CN/Investigación de Operaciones.
- 13 CN/Computación Paralela.

SÍLABOS DE LOS CURSOS

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

I. DESCRIPCION DEL CURSO

CURSO	Introducción a la Ciencia de la Computación
CODIGO	CC101
CREDITOS	02 (dos)
PRE-REQUISITO	Ninguno
CHARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (02)

II. OBJETIVO DEL CURSO

Brindar un panorama del área del conocimiento que es cubierta en la ciencia de la Computación.

III. SUMILLA

La Ciencia de la Computación es un campo de estudio enorme con muchas especialidades y aplicaciones. Este curso brindará a sus participantes, una visión panorámica de la informática y mostrará sus campos mas representativos, como son: Algoritmos, Estructuras de de Datos, Sistemas Operativos, Bases de Datos, etc.

Contribución a los resultados

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- a) Habilidad para aplicar conocimiento de Computación y de matemática apropiadas para la disciplina.
- b) Habilidad para analizar un problema e identificar y definir los requerimientos computacionales apropiados para su solución.
- c) Habilidad para entender correctamente las implicatorias profesionales, éticas, legales, de seguridad y sociales de la profesión.
- d) Habilidad para analizar el impacto local y global de la Computación sobre los individuos, organizaciones y sociedad.
- e) Habilidad para incorporarse a un proceso de aprendizaje profesional continuo.
- f) Habilidad para usar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- g) Habilidad para aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida.

IV. CONTENIDO ANALITICO

4.1 Lógica Básica.

Tópicos. I: Lógica proposicional. II: Conectivos lógicos III: Tablas de verdad.

Objetivos. 1. Aplicar métodos formales de lógica simbólica proposicional y de predicados.

4.2 Fundamentos de Programación.

Tópicos. I: Sintaxis básica y semántica de un lenguaje de mas alto nivel. II: Variables, tipos, expresiones, y asignaciones. III: Estructuras de control condicionales e iterativas.

Objetivos. 1. Analizar y explicar el comportamiento de programas simples involucrando las estructuras de programación fundamental cubiertas por esta unidad.

4.3 Algoritmos y Resolución de Problemas.

Tópicos. I: El rol de los algoritmos en el proceso de solución de problemas. II: El Concepto y propiedades de algoritmos.

Objetivos. 1. Discutir la importancia de los algoritmos en el proceso de solución de problemas. 2. Crear algoritmos para resolver problemas simples. 3. Usar pseudocódigo o un lenguaje de programación para implementar, probar y depurar algoritmos para resolver problemas simples.

4.4 Estructuras de Datos Fundamentales.

Tópicos. I: Tipos primitivos. II: Arreglos.

Objetivos. 1. Discutir la representación y uso de tipos de datos primitivos y estructuras de datos incorporadas en el lenguaje.

4.5 Análisis de Algoritmos Básicos.

Tópicos. I: Identificar la diferencias entre casos de mejor, mediano y peor comportamiento.

Objetivos. 1. Determinar la complejidad de tiempo y espacios de algoritmos simples.

4.6 Estrategias Algorítmicas.

Tópicos. I: Algoritmos de fuerza bruta (brute-force). II: Dividir y conquistar.

Objetivos. 1. Describir el defecto de los algoritmos de fuerza bruta. 2. Implementar un algoritmo de dividir y conquistar para solucionar un problema apropiado.

4.7 Lógica Digital y Sistemas Digitales.

Tópicos. I: Vista panorámica e historia de la arquitectura de computadores.

Objetivos. 1. Demostrar cierto entendimiento de los bloques de construcción básicos y su rol en el desarrollo histórico de la arquitectura del computador.

4.8 Representación de Datos a Nivel de Máquina.

Tópicos. I: Bits, bytes y palabras. II: Representación datos numéricos y bases numéricas. III: Sistemas de punto flotante y fijos. IV: Representación de complemento a dos y señalizada. V: Representación de datos no numéricos (código de caracteres, datos gráficos). VI: Representación de registros y arrays.

Objetivos. 1. Explicar la razón de uso de diferentes formatos para representar datos numéricos. 2. Explicar cómo los enteros negativos son almacenados en las representaciones de complemento a dos y señalizada. 3. Convertir datos numéricos de un formato a otro. 4. Discutir como la representación de números de longitud fija afecta la precisión y exactitud. 5. Describir la representación interna de datos no numéricos.

4.9 Organización de Máquina a Nivel de Ensamble.

Tópicos. I: Organización básica de la máquina de Von Neumann. II: Unidad de control, instrucción de búsqueda (fetch), de-codificación, y ejecución. III: Lenguaje de programación de máquina y ensamblador.

Objetivos. 1. Explicar la organización de la máquina clásica de Von Neumann y su unidad Funcional mayor. 2. Explicar como una instrucción es ejecutada en una máquina clásica de Von Neumann. 3. Resumir como las instrucciones son representadas tanto a nivel de máquina y en el contexto simbólico de ensamblaje. 4. Escribir segmentos simples de programa en lenguaje ensamblador.

4.10 Organización de Sistemas de Memoria y Arquitectura.

Tópicos. I: Sistemas de almacenamiento y su tecnología.

Objetivos. 1. Identificar los tipos principales de la tecnología de memoria.

4.11 Comunicación e Interfase.

Tópicos. I: Estructuras de interrupción: vectorizadas y priorizadas, reconocimiento de una interrupción. II: Buses: protocolos de buses, arbitraje, acceso directo a memoria (DMA). III: Introducción a redes.

Objetivos. 1. Identificar varios tipos de buses en un sistema de computador. 2. Comparar las configuraciones de redes comunes.

4.12 Visión General de los Sistemas Operativos.

Tópicos. I: Rol y propósito de los sistemas operativos. II: Historia del desarrollo de los sistemas operativos. III: Funcionalidad de un sistema operativo típico.

Objetivos. 1. Explicar los objetivos y funciones de los sistemas operativos modernos. 2. Describir como los sistemas operativos han evolucionado en el tiempo desde sistemas primitivos batch a sofisticados sistemas multiusuarios.

4.13 Introducción a la Computación Centrada en Redes.

Tópicos. I: Historia de las redes y de la Internet.

Objetivos. 1. Discutir la evolución de las primeras redes y de la Internet. 2. Demostrar la habilidad para usar efectivamente un conjunto de aplicaciones de red incluyendo e-mail, telnet, FTP, newgroups, navegadores web, cursos en línea, y mensajería instantáneos.

4.14 Visión General de los Lenguajes de Programación.

Tópicos. I: Historia de los lenguajes de programación II: Breve revisión de los paradigmas de programación. a) Lenguajes procedimentales b) Lenguajes orientados a objetos. c) Lenguajes Funcionales. d) Lenguajes declarativos y no algorítmicos. e) Lenguajes de scripts.

Objetivos. 1. Listar la evolución de los lenguajes de programación identificando como es que su historia nos ha conducido a los paradigmas actuales. 2. Identificar al menos una característica distintiva para cada uno de los paradigmas de programación cubiertos en esta unidad.

4.15 Máquinas Virtuales.

Tópicos. I: El concepto de máquina virtual.

Objetivos. 1. Describir la importancia y poder de la abstracción en el contexto de máquinas virtuales.

4.16 Programación Orientada a Objetos.

Tópicos. I: Diseño orientado a objetos. II: Clases y subclases. III: Herencia (sobre-escritura, despacho dinámico). IV: Polimorfismo (polimorfismo de subtipo vs. herencia).

Objetivos. 1. Justificar la filosofía de diseño orientado a objetos y los conceptos de encapsulación, abstracción, herencia y polimorfismo.

4.17 Programación Funcional.

Tópicos. I: Panorama general y motivación de los lenguajes Funcionales.

Objetivos. 1. Delinear las fortalezas y debilidades del paradigma de programación Funcional.

4.18 Sistemas de Traducción del Lenguaje.

Tópicos. I: Aplicación de las expresiones regulares en analizadores léxicos. II: Análisis sintáctico (sintaxis concreta y abstracta, árboles de sintaxis abstracta).

Objetivos. 1. Describir los pasos y algoritmos usados por lenguajes traductores.

4.19 Fundamentos de la Interacción Hombre-Computador

Tópicos. I: Motivación: ¿Qué es lo que nos importa de las personas? II: Contexto de HCI (herramientas, hipermedios web, comunicación).

Objetivos. 1. Discutir las razones del desarrollo de software centrado en el humano. 2. Listar la ciencia básica de la interacción social y psicológica.

4.20 Tópicos Fundamentales en Sistemas Inteligentes.

Tópicos. I: Historia de la inteligencia artificial. II: Cuestiones filosóficas. III: La prueba de Turing.

Objetivos. 1. Describir la prueba de Turing y el experimento de pensamiento del “Cuarto Chino”.

4.21 Modelos de Información y Sistemas.

Tópicos. I: Historia y motivación para los sistemas de información.

Objetivos. 1. Comparar y contrastar la información con datos y conocimiento. 2. Listar la evolución de los sistemas de información desde las visiones tempranas hasta las propuestas modernas, distinguiendo sus respectivas capacidades y potenciales futuros.

4.22 Sistemas de Base de Datos.

Tópicos. I: Historia y motivación de los sistemas de base datos. II: Componentes de los sistemas de base de datos.

Objetivos. 1. Explicar las características que distinguen los métodos de base de datos de los métodos tradicionales de programación con archivos de datos. 2. Citar las metas básicas como funciones, modelos, componentes, aplicaciones y el impacto social de los sistemas de bases de datos. 3. Describir los componentes de un sistema de base de datos y dar ejemplos de su uso.

4.23 Modelamiento de Datos.

Tópicos. I: Modelamiento de datos. II: Modelos conceptuales (incluyendo entidad-relación, y UML).

Objetivos. 1. Describir los conceptos de modelado y la notación del modelo entidad-relación y UML, incluyendo su uso en modelamiento de datos.

4.24 Base de Datos Relacionales.

Tópicos. I: Álgebra relacional y cálculo relacional.

Objetivos. 1. Demostrar consultas en el álgebra relacional.

4.25 Contexto Social de la Computación.

Tópicos. I: Introducción a las implicaciones sociales de la computación II: Crecimiento, control y acceso a la Internet.

Objetivos. 1. Interpretar el contexto social de una implementación particular. 2. Describir las formas positivas o negativas en las cuales la Computación altera los modos de interacción entre las personas.

4.26 Propiedad Intelectual.

Tópicos. I: Fundamentos de la propiedad intelectual. II: Copyrights, patentes y secretos del negocio. III: Piratería de software.

Objetivos. 1. Distinguir entre patentes, copyright, y protección de secretos del negocio. 2. Discutir el fondo legal del copyright en las leyes Nacionales e internacionales 3. Explicar como las leyes de patentes y el copyright pueden variar internacional mente 4. Resumir el desarrollo histórico de las patentes de software.

4.27 Diseño de Software.

Tópicos. I: Conceptos fundamentales de diseño y principios.

Objetivos. 1. Discutir las propiedades del buen diseño de software.

4.28 Usando APIs.

Tópicos. I: Programación API.

Objetivos. 1. Explicar el valor de las interfaces para programación de aplicaciones (APIs) en el desarrollo de software.

4.29 SE3. Herramientas de Software y Entornos.

Tópicos. I: Entornos de programación.

Objetivos. 1. Analizar y evaluar un conjunto de herramientas en una área dada del desarrollo de software (ejemplo, administración, modelamiento, o pruebas).

4.30 Procesos de Software.

Tópicos. I: Ciclo de vida del software y modelos de procesos.

Objetivos. 1. Explicar el ciclo de vida del software y sus fases incluyendo las entregas que son producidas.

V. BIBLIOGRAFIA

- Brookshear, J. G. (2005). Computer Science: An Overview, 8/E. Addison-Wesley.
- Forouzan, B. and Mosharraf, F. (2008). Foundations of Computer Science, 2nd Ed., 2008. Thomson Course Technology.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Introducción a la Programación
CODIGO	CC102
CREDITOS	02 (dos)
PRE-REQUISITO	CC101
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (01), LABORATORIO (02), SEMINARIO (01)

DOCENTE

FUNDAMENTACION DEL CURSO

Este es el primer curso en la secuencia de los cursos introductorios a la informática. La programación es uno de los pilares de la informática; cualquier profesional del área, necesitará programar para concretizar sus modelos y propuestas.

Este curso busca introducir a los participantes en los conceptos fundamentales de este arte. Los tópicos incluyen tipos de datos, estructuras de control, funciones, listas, recursividad y la mecánica de la ejecución, prueba y depuración.

El curso también ofrecerá una introducción al contexto histórico y social de la informática y una revisión del ámbito de esta disciplina.

SUMILLA

1. SP/Historia de la Computación 2. PL/Visión General de los Lenguajes de Programación 3. PL/Declaración y Tipos. 4. PF/Construcciones fundamentales. 5. PL/Programación Funcional. 6. PF/Recursividad. 7. AL/Algoritmos Fundamentales. 8. PL/Mecanismos de Abstracción 9. PF/Algoritmos y Resolución de Problemas. 10. DS/Grafos y Árboles. 11. AL/Computabilidad Básica 12. PL/Maquinas Virtuales. 13. PL/Programación Orientada a Objetos. 14. SE/Especificación de Requerimientos. 15. SE/Diseno de Software. 16. SE/Herramientas y Entornos de Software. 17. SE/Usando APIs.

CONTRIBUCION A LOS RESULTADOS

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- Habilidad para aplicar conocimiento de computación y de matemática apropiadas para la disciplina.
- Habilidad para analizar un problema e identificar y definir los requerimientos computacionales apropiados para su solución.

- c) Habilidad para diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas.
- d) Habilidad para trabajar efectivamente en equipos para cumplir con un objetivo común.
- e) Habilidad para usar técnicas y herramientas actuales necesarias para la práctica de la computación.
- f) Habilidad para aplicar los principios de desarrollo y diseño en la construcción de sistemas de software de complejidad variable.

CONTENIDO ANALITICO

1 Historia de la Computación. (1 hora) [SP1- 1 hora del núcleo]

Tópicos. I: Historia del hardware de computadoras, software, redes. II: Pioneros de la computación.

Objetivos. 1. Listar las contribuciones de varios pioneros en el campo de la computación. 2. Comparar la vida diaria antes y después del advenimiento de las computadoras personales e Internet. 3. Identificar las tendencias continuamente significativas en la historia del campo de la computación.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005], [Bird, 1998]

2 Visión General de los Lenguajes de Programación (1 hora) [PL1 – 1 (de 2) horas del núcleo]

Tópicos. I: Historia de los lenguajes de programación.

Objetivos. 1. Listar la evolución de los lenguajes de programación identificando como es que su historia nos ha conducido a los paradigmas actuales. 2. Identificar al menos una característica distintiva para cada uno de los paradigmas de programación cubiertos en esta unidad.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005], [Bird, 1998]

3 Declaración y Tipos. (1 hora) [PF1 – 1 (de 9) horas del núcleo]

Tópicos. I: Declaración de modelos (enlace, visibilidad, alcance, y tiempo de vida). II: Vista general de la revisión de tipos. III: Recolección de basura.

Objetivos. 1. Explicar el valor de los modelos de declaración, especialmente con respecto a la programación en mayor escala. 2. Identificar y describir las propiedades de una variable, tales como su: dirección asociada, valor, ámbito, persistencia y tamaño. 3. Discutir la incompatibilidad de tipos. 4. Demostrar las diferentes formas de enlace, visibilidad, ámbito, manejo del tiempo de vida. 5. Defender la importancia de los tipos y el chequeo de tipos para brindar abstracción y o seguridad.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

4 Fundamentos de programación. (6 horas) [PF1 – 6 (de 9) horas del núcleo]

Tópicos. I: Sintaxis básica y semántica de un lenguaje de más alto nivel. II: Variables, tipos, expresiones, y asignaciones. III: Entrada y salida simple. IV: Estructuras de control condicionales e iterativas. V: Funciones y paso de parámetros. VI: Descomposición estructurada.

Objetivos. 1. Analizar y explicar el comportamiento de programas simples involucrando las estructuras de programación fundamental cubiertas por esta unidad. 2. Modificar y extender programas cortos que usan condicionales estándar, estructuras de control iterativo y funciones. 3. Diseñar, implementar, probar y depurar un programa que use cada una de las siguientes estructuras fundamentales de programación: cálculos básicos, entrada y salida simple, estructuras estándar condicionales e iterativas y definición de funciones. 4. Escoger la estructura apropiada condicional y/o iterativa para una estructura de programación dada. 5. Aplicar técnicas de descomposición estructurada o funcional para dividir un programa en pequeñas partes.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

5 Programación Funcional. (1 hora)

Tópicos. I: Panorama general y motivación de los lenguajes funcionales. II: Recursión sobre listas, números naturales, árboles, y otros datos definidos recursivamente. III: Pragmáticas (depuración dividir y vencerás, persistencia de las estructuras de datos). IV: Cerraduras y uso de funciones como datos (conjuntos infinitos, flujos).

Objetivos. 1. Delinear las fortalezas y debilidades del paradigma de programación funcional. 2. Diseñar, codificar, probar y depurar programas usando el paradigma funcional.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

6 Recursividad. (1 hora) [PF1 – (1 de 9) horas del núcleo]

Tópicos. I: El concepto de recursividad. II: Funciones matemáticas recursivas. III: Procedimientos recursivos simples. IV: Estrategias de dividir y conquistar. V: Backtracking recursivo. VI: Implementación de recursividad.

Objetivos. 1. Describir el concepto de recursividad y dar ejemplos de su uso. 2. Identificar el caso base y el caso general de un problema definido recursivamente. 3. Comparar soluciones iterativas y recursivas para problemas elementales tal como factorial. 4. Describir la técnica dividir y conquistar. 5. Implementar, probar y depurar funciones y procedimientos recursivos simples. 6. Describir como la recursividad puede ser implementada usando una pila. 7. Discutir problemas para los cuales el backtracking es una solución apropiada.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

7 Algoritmos de Computación Fundamental. (2 horas) [AL3 - 2 (de 6) horas del núcleo]

Tópicos. I: Algoritmos numéricos simples. II: Búsqueda secuencial y binaria. III: Algoritmos cuadráticos de ordenamiento (selección, inserción). IV: Algoritmos de tipo $O(N \log N)$ (Quicksort, heapsort, mergesort). V: Árboles de búsqueda binaria.

Objetivos. 1. Implementar los algoritmos cuadráticos mas comunes y los algoritmos de ordenamiento $O(N \log N)$. 2. Diseñar e implementar una función (hash) apropiada para una aplicación. 3. Discutir la eficiencia computacional de los principales algoritmos de ordenamiento, búsqueda y (hashing).

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

4.8 Mecanismos de Abstracción. (2 horas) [PL5 – 2 (de 3) horas del núcleo]

Tópicos. I: Procedimientos, funciones e iteradores como mecanismos de abstracción. II: Parámetros de tipos y tipos parametrizados. III: Módulos en lenguajes de programación.

Objetivos. 1. Explicar como los mecanismos de abstracción soportan la creación de componentes de software reusables. 2. Defender la importancia de la abstracción especialmente con respecto a la programación en mayor escala.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

4.9 Estructuras de Datos Fundamentales. (4 horas) [PF3 – 6 (de 14) horas del núcleo]

Tópicos. I: Tipos primitivos. II: Cadenas y procesamiento de cadenas. III: Representación de datos en memoria. IV: Estructuras enlazadas.

Objetivos. 1. Discutir la representación y uso de tipos de datos primitivos y estructuras de datos incorporadas en el lenguaje. 2. Describir aplicaciones comunes para cada estructura de datos en la lista de temas. 3. Implementar estructuras de datos definidas por el usuario en un lenguaje de alto nivel. 4. Escribir programas que usan cada una de las siguientes estructuras de datos: arreglos, registros, cadenas, listas enlazadas, pilas, colas y tablas hash.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

10 Grafos y Árboles. (1 hora)

Tópicos. I: Grafos no dirigidos. II: Grafos dirigidos. III: Árboles de expansión.

Objetivos. 1. Ilustrar con ejemplos la terminología básica de teoría de grafos, y algunas de las propiedades y casos especiales de cada una.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

11 . Algoritmos y Resolución de Problemas. (3 horas) [PF2 3 (de 6) horas del núcleo]

Tópicos. I: Estrategias para la solución de problemas. II: El rol de los algoritmos en el proceso de solución de problemas. III: Estrategias de implementación para algoritmos. IV: Estrategias de depuración. V: El Concepto y propiedades de algoritmos.

Objetivos. 1. Discutir la importancia de los algoritmos en el proceso de solución de problemas. 2. Identificar las propiedades necesarias de un buen algoritmo. 3. Crear algoritmos para resolver problemas simples. 4. Usar pseudocódigo o un lenguaje de programación para implementar, probar y depurar algoritmos para resolver problemas simples.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

12 . Computabilidad Básica. (1 hora) [AL5 – 1 (de 6) horas del núcleo]

Tópicos. I: Gramáticas libres del contexto. II: Problemas tratables e intratables. III: Funciones no computables. IV: Funciones no computables.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Objetivos. 1. Explicar cómo algunos problemas no tienen solución algorítmica.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

14 . Programación Orientada a Objetos. (2 horas) [PL6 – 2 (de 10) horas del núcleo]

Tópicos. I: Encapsulación y ocultamiento de la información. II: Separación de comportamiento e implementación.

III: Clases y subclases. IV: Polimorfismo (polimorfismo de subtipo vs. herencia). V: Jerarquía de clases.

Objetivos. 1. Describir como los mecanismos de clases soportan encapsulación y ocultamiento de la información.

2. Diseñar, implementar y probar la implementación de la relación es-un IsKindOf entre objetos usando jerarquía de clases y herencia. 3. Comparar y contrastar las nociones de sobrecarga y sobre-escritura de métodos en un lenguaje orientado a objetos.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

15 . Requerimientos de Software y Especificaciones. (1 hora) [SE5 – 1 (de 4) hora del núcleo]

Tópicos. I: Conceptos básicos de técnicas de especificación formal.

Objetivos. 1. Traducir en lenguaje natural una especificación de requerimientos de software escrita en un lenguaje de especificación formal comúnmente usado.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

16 . Diseño de Software. (1 hora) [SE1 – 1 (de 8) horas del núcleo]

Tópicos. I: Conceptos fundamentales de diseño y principios.

Objetivos. 1. Discutir las propiedades del buen diseño de software.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

17 Herramientas de Software y Entornos. (1 hora) [SE5 – 1 (de 4) horas del núcleo]

Tópicos. I: Entornos de programación. II: Análisis de requerimientos y herramientas de modelamiento de diseño.

Objetivos. 1. Seleccionar con justificación un apropiado conjunto de herramientas para soportar el desarrollo de un rango de productos de software.

Bibliográfica: [Thompson, 1999], [Paz-Valderrama, 2005]

BIBLIOGRAFÍA

Bird, R. J. (1998). Introducción to Functional Programming using Haskell. Prentice-Hall Series in Computer Science. Prentice-Hall Europe, London, UK, second edition.

Paz-Valderrama, A. (2005). Apuntes: Programación funcional en haskell. Sociedad Peruana de Computación.

Thompson, S. (1999). The Craft of Functional Programming, 2E. Addison Wesley.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Introducción a la Programación Orientada a Objetos	
CODIGO	CC201	
CREDITOS	04 (cuatro)	
PRE-REQUISITO	CC102	
CARACTER	Obligatorio	
MODALIDAD	Semestral	
HORAS	TEORIA (03)	LABORATORIO (03)

OBJETIVO DEL CURSO

- El curso servirá como puente entre el paradigma de la programación Funcional y el orientado al objeto, además introducirá a los participantes en los diversos temas del área de Computación como: algoritmos, estructuras de datos, ingeniería del software, etc.
- Introducir al alumno a los fundamentos del paradigma de orientación a objetos, permitiendo o asimilar los conceptos necesarios para desarrollar un sistema de información.
- Presenta el lenguaje C++ e introduce la programación orientada a objetos. Introduce el concepto de templates y librerías, para poder programar en paralelo y hacer uso de los múltiples cores/núcleos localizados en un solo chip en forma simultánea.
- También se presenta OpenMP como una de las extensiones al lenguaje C y C++ para programar en paralelo desde un punto de vista de procesadores sin importar si se tiene o no computadores con múltiple núcleo.

SUMILLA

Desarrollo del paradigma orientado a objetos para que se permita modelar abstracciones de los elementos del problema y usar técnicas de encapsulamiento, modularidad, polimorfismo y herencia para dar soluciones computacionales a problemas de diseño sencillos del mundo real. El estudiante también aprenderá el uso de templates y librerías para aprender a programar en paralelo. OpenMP será usado para programar en paralelo.

CONTENIDO ANALITICO

4.1 Visión General de los Lenguajes de Programación.

Tópicos. I: Breve revisión de los paradigmas de programación. a) Lenguajes procedimentales b) Lenguajes orientados a objetos. c) Lenguajes Funcionales. d) Lenguajes declarativos y no algorítmicos. e) Lenguajes de scripts. II: Comparación entre programación Funcional y programación imperativa III: Historia de los lenguajes de programación.

Objetivos. 1. Listar la evolución de los lenguajes de programación identificando como es que su historia nos ha conducido a los paradigmas actuales. 2. Identificar al menos una característica distintiva para cada uno de los paradigmas de programación cubiertos en esta unidad. 3. Evaluar las ventajas y desventajas entre los diferentes paradigmas, considerando temas tales como: eficiencia de espacio, esencia en el tiempo (para ambas partes computadora y programador), seguridad, y el poder de las expresiones.

4.2 Máquinas Virtuales.

Tópicos. I: El concepto de máquina virtual. II: Lenguajes intermedios.

Objetivos. 1. Describir la importancia y poder de la abstracción en el contexto de máquinas virtuales. 2. Explicar los beneficios de los lenguajes intermedios en el proceso de compilación.

4.3 Declaración y Tipos.

Tópicos. I: La concepción de tipos como un conjunto de valores unidos a un conjunto de operaciones. II: Declaración de modelos (enlace, visibilidad, alcance, y tiempo de vida). III: Vista general o de la revisión de tipos.

Objetivos. 1. Explicar el valor de los modelos de declaración, especialmente con respecto a la programación en mayor escala. 2. Identificar y describir las propiedades de una variable, tales como o su: dirección asociada, valor, ámbito, persistencia y tamaño. 3. Discutir la incompatibilidad de n tipos. 4. Demostrar las diferentes formas de enlace, visibilidad, ámbito, manejo del tiempo de vida. 5. Defender la importancia de los tipos y el chequeo de tipos para brindar abstracción y seguridad.

4.4 Mecanismos de Abstracción.

Tópicos. I: Procedimientos, funciones e iteradores como mecanismos de abstracción. II: Mecanismos de parametrización (referencia vs. valor). III: Módulos en lenguajes de programación.

Objetivos. 1. Explicar como los mecanismos de abstracción soportan la creación de componentes de software reusables. 2. Demostrar la diferencia entre paso de parámetros por valor y por referencia. 3. Defender la importancia de la abstracción especialmente con respecto a la programación en o mayor escala.

4.5 Programación Orientada a Objetos.

Tópicos. I: Diseño orientado a objetos. II: Encapsulación y ocultamiento de la información. III: Separación de comportamiento e implementación. IV: Clases y subclasses. V: Herencia (sobre-escritura, despacho dinámico). a

Objetivos. 1. Justificar la filosofía de diseño orientado a objetos y los conceptos de encapsulación, abstracción, herencia y polimorfismo. 2. Diseñar, implementar, probar, y depurar programas simples en un lenguaje de programación orientado a objetos. 3. Describir como los mecanismos de clases soportan encapsulación y ocultamiento de la información. 4. Diseñar, implementar y probar la implementación de la relación es-un IsKindOf entre objetos usando jerarquía de clases y herencia. 5. Comparar y contrastar las nociones de sobrecarga y sobre-escritura de métodos en un lenguaje orientado a objetos.

4.6 Fundamentos de Programación.

Tópicos. I: Sintaxis básica y semántica de un lenguaje de mas alto nivel. II: Variables, tipos, expresiones, y asignaciones. III: Entrada y salida simple. IV: Estructuras de control condicionales e iterativas. V: Funciones y paso de parámetros. VI: Descomposición estructurada.

Objetivos. 1. Analizar y explicar el comportamiento de programas simples involucrando las estructuras de programación fundamental cubiertas por esta unidad. 2. Modificar y extender programas cortos que usan condicionales estándar, estructuras de control iterativo y funciones. 3. Diseñar, implementar, probar y depurar un programa que use cada una de las siguientes estructuras fundamentales de programación: cálculos básicos, entrada y salida simple, estructuras estándar o condicionales e iterativas y definición de funciones. 4. Escoger la estructura apropiada condicional o y/o iterativa para una estructura de programación dada. 5. Aplicar técnicas de descomposición estructurada o Funcional para dividir un programa en pequeñas partes. 6. Describir los mecanismos de paso de parámetros.

4.8 Estructuras de Datos Fundamentales.

Tópicos. I: Tipos primitivos. II: Arreglos. III: Registros. IV: Cadenas y procesamiento de o cadenas. V: Punteros y referencias.

Objetivos. 1. Discutir la representación y uso de tipos de datos primitivos y estructuras de datos incorporadas en el lenguaje. 2. Implementar estructuras de datos definidas por el usuario en un lenguaje de alto nivel. 3. Escribir programas que usan cada una de las siguientes estructuras de datos: arreglos, registros, cadenas, listas enlazadas, pilas, colas y tablas hash.

4.9 Recursividad.

Tópicos. I: Procedimientos recursivos simples. II: Implementación de recursividad.

Objetivos. 1. Comparar soluciones iterativas y recursivas para problemas elementales tal como factorial. 2. Implementar, probar y depurar funciones y procedimientos recursivos simples.

4.10 Programación Orientada a Eventos.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Tópicos. I: Métodos para la manipulación de eventos. II: Propagación de eventos. III: Manejo de excepciones.
Objetivos. 1. Explicar la diferencia entre programación orientada a eventos y programación por línea de comandos. 2. Diseñar, codificar, probar y depurar programas de manejo de eventos simples que respondan a eventos del usuario. 3. Desarrollar código que responda a las condiciones de excepción lanzadas durante la ejecución.

4.11 Análisis de Algoritmos Básicos.

Tópicos. I: Identificar las diferencias entre casos de mejor, mediano y peor comportamiento.

Objetivos. 1. Determinar la complejidad de tiempo y espacios de algoritmos simples.

4.12 Algoritmos de Computación Fundamental.

Tópicos. I: Búsqueda secuencial y binaria. II: Algoritmos cuadráticos de ordenamiento (selección, inserción).

Objetivos. 1. Resolver problemas de búsqueda y ordenación. 2. Analizar el coste de los algoritmos de búsqueda

4.13 Historia de la Computación.

Tópicos. I: Pioneros de la Computación.

Objetivos. 1. Identificar las tendencias continuamente significativas en la historia del campo de la Computación.

4.14 Diseño de Software.

Tópicos. I: Conceptos fundamentales de diseño y principios. II: Análisis y diseño orientado a objetos. III: Diseño para reuso.

Objetivos. 1. Comparar y contrastar el análisis orientado a objetos con el análisis y diseño estructural.

4.15 Requerimientos de Software y Especificaciones

Tópicos. I: Elicitación de requerimientos. II: Técnicas de modelamiento del análisis de requerimientos.

Objetivos. 1. Aplicar elementos clave y métodos comunes para la elicitación y análisis, para producir un conjunto de requerimientos de software para un sistema de tamaño medio. 2. Usar un método común, no formal para modelar y especificar (en la forma de un documento de especificación de requerimientos) los requerimientos para un sistema de software de tamaño medio.

BIBLIOGRAFIA

- <http://www.cplusplus.com/doc/tutorial/>
- <http://www.4p8.com/eric.brasseur/cppcen.html>
- Intel Threading Building Blocks por James Reinders Editorial: Sebastopol, CA : O'Reilly Media, 2007.
- The Art of C++, por Herbert Schildt. Publicado por McGraw-Hill Professional, 2004.
- C++ from the Ground Up, por Herbert Schildt. Ed: 3, Publicado por Osborne/McGraw-Hill, 2003.
- Herb Schildt's C++ Programming Cookbook, por Herbert Schildt. Publicado por McGraw-Hill Professional, 2008.
- Head First Object-Oriented Analysis and Design, Brett D. McLaughlin, Gary Pollice, Dave West, O'Reilly
- Object Modeling and User Interface Design: Designing Interactive Systems, Mark Van Harmelen, Stephanie Wilson
- Design Patterns: Elements of Reusable Object-Oriented Software, Erich Gamma, Richard Helm, Ralph Johnson, John M. Vlissides, Addison-Wesley.
- Bjarne Stroustrup, *El lenguaje de programación C++*, Addison Wesley, Madrid, 1998, ISBN 84-7829-019-2
- Margaret A. Ellis y Bjarne Stroustrup (1990), *The Annotated C++ Reference Manual*, Addison-Wesley Pub Co; ISBN 0-201-51459-1
- Aguilar, L. J. (1998). Hill/Interamericana de España. Programación Orientada a Objetos. o McGraw-Hill
- Harvey M. Deitel, P. J. (2004). Como Programar en C/C++ y Java. Pearson o Educación.
- Smith, J. A. (2001). Desarrollo de Proyectos con Programación Orientada a Objetos o con C++. Thomson Learning.
- Stroustrup, B. (2000). The C++ Programming Language. Addison-Wesley, 3rd edition.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Arquitectura de Computadores
CODIGO	CC212
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC102, CM254
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Permitir al alumno gestionar adecuadamente el hardware y el software de un sistema de cómputo.
- Garantizar el buen desempeño y la eficiencia de la futura codificación.
- Demostrar que la evolución de la arquitectura de los computadores es una consecuencia del avance y desarrollo tanto del hardware como del software.

SUMILLA

El curso comprende el diseño del conjunto de instrucciones del computador, micro-arquitectura del procesador, pipeline, coherencia del cache, memoria virtual, E/S, Interrupciones, arquitecturas super-escalares, computadoras vectoriales, y computadores paralelos.

CONTENIDO ANALITICO

4.1 Lógica Digital y Sistemas Digitales.

Tópicos. I: Bloques fundamentales de construcción (compuertas lógicas, flip-flops, contadores, registros).

Objetivos. 1. Describe la progresión de la arquitectura del computador desde tubos de vacío VLSI. 2. Diseñar un circuito simple usando los bloques de construcción fundamentales. 3. Uso de expresiones matemáticas para describir las funciones combinacionales simples y circuitos a secuenciales.

4.2 Representación de Datos a Nivel de Máquina.

Tópicos. I: Bits, bytes y palabras. II: Representación datos numéricos y bases numéricas.

Objetivos. 1. Convertir datos numéricos de un formato a otro. 2. Discutir como la representación de números de longitud fija afecta la precisión y exactitud. 3. Describir la representación interna de datos no numéricos. 4. DESCRIPCION de la representación interna de caracteres, cadenas, registros y arreglos.

4.3 Organización de Máquina a Nivel de Ensamble.

Tópicos. I: Unidad de control, instrucción de búsqueda (fetch), decodificación, y ejecución. II: Conjuntos de instrucción y tipos (manipulación de datos, control de entrada y salida). III: Lenguaje de programación de máquina y ensamblador. IV: Formatos de instrucciones.

Objetivos. 1. Explicar como una instrucción es ejecutada en una máquina clásica de Von Neumann. 2. Resumir como las instrucciones son representadas tanto a nivel de máquina y en el contexto a simbólico de ensamblaje. 3. Explicar los diferentes formatos de instrucciones, tales como dirección por instrucción, formatos de longitud variable vs. formatos de longitud fija. 4. Escribir segmentos o simples de programa en lenguaje ensamblador. 5. Demostrar como la construcción de lenguajes de programación de alto nivel fundamentales son implementados a nivel de lenguaje de máquina. 6. Explicar como las llamadas a subrutinas son manejadas a nivel de ensamblador.

4.4 Organización de Sistemas de Memoria y Arquitectura.

Tópicos. I: Codificación, compresión de datos e integridad de datos. II: Jerarquía de memoria. III: Memorias cache, (mapeo de direcciones, tamaño de bloques, políticas de reemplazo y almacenamiento). IV: Memoria Virtual (tablas de paginación, TBL).

Objetivos. 1. Identificar los tipos principales de la tecnología de memoria. 2. Explicar el uso de jerarquía de memoria para reducir la latencia de memoria efectiva. 3. Describir los principios de administración de memoria. 4. Describir el rol de la memoria cache y virtual. 5. Explicar el trabajo de un sistema con gerenciamiento de memoria virtual.

4.5 Comunicación e Interfase.

Tópicos. I: Fundamentos de entrada y salida: buffering, handshaking, entradas y salidas programadas, entradas y salidas manejadas por interrupciones. II: Estructuras de interrupción: vectorizadas y priorizadas, reconocimiento de una interrupción. III: Almacenamiento externo, organización física y drivers. IV: Buses: protocolos de buses, arbitraje, acceso directo a memoria (DMA). V: Arquitecturas RAID.

Objetivos. 1. Explicar como las interrupciones son usadas para implementar control de entrada/salida y transferencia de datos. 2. Identificar varios tipos de buses en un sistema de computador. 3. Describir el acceso a datos desde discos magnéticos. 4. Comparar las configuraciones de redes comunes. 5. Describir las ventajas y limitaciones de las arquitecturas RAID.

4.6 Organización Funcional.

Tópicos. I: Unidad de control: entendimiento de incrustación en el hardware vs. micro-programación II: Introducción al paralelismo a nivel de instrucción (IPL).

Objetivos. 1. Discutir el concepto de puntos de control y generación de señales de control usando incrustación en el hardware o micro-programación 2. Explicar el paralelismo a nivel de instrucciones básicas usando pipelining y explicar los principales riesgos del uso.

4.7 Multiprocesamiento y Arquitecturas Alternativas.

Tópicos. I: Introducción a SIMD, MIMD, VLIW, EPIC. II: Arquitectura sistólica. III: Interconexión de redes (Hipercubo, intercambio aleatorio, malla, barra transversal). IV: Sistemas de memoria compartida.

Objetivos. 1. Describir arquitecturas alternativas tales como SIMD, MIMD, y VLIW. 2. Explicar el concepto de Interconexión de redes y caracterizar diferentes métodos. 3. Discutir las características especiales que los sistemas de Multiprocesamiento presentan con respecto a la administración de memoria y describir como estas son direccionadas.

4.8 Herramientas de Diagnóstico y seguridad

Tópicos. I: Evaluación de los sistemas de cómputo II: Diagnóstico de los sistemas de procesamiento III: Seguridad y virus informáticos

Objetivos. 1. Estudiar los parámetros de rendimiento de un procesador 2. Uso de herramientas para evaluar la seguridad

4.9 Plataformas de Programación y núcleos básicos

Tópicos. I: Integrar la parte de hardware y software mediante la implementación de un núcleo básico para manejar periféricos, memoria principal, interrupciones, etc.

Objetivos. 1. Entender como interactúan las diferentes partes de software y hardware para el manejo de un sistema de cómputo.

BIBLIOGRAFIA

- Brey, B. B. (2005). The Intel Microprocessors: 8086/8088, 80186, 80286, 80386, 80486, Pentium, Pentium Pro, and Pentium II, Pentium III, Pentium 4. Prentice-Hall, 7th edition.
- Mano, M. M. (1992). Computer System Architecture. Prentice Hall, 3rd edition.
- Norton, P. (1988). Peter Norton's Assembly Language Book for the IBM PC. Peter Norton Foundation Series. Brady Publishing. 0136624537.
- D. Tullsen, S. Eggers and H. Levy, [Simultaneous Multithreading: Maximizing On-Chip Parallelism](#), in the Proceedings of the 22rd Annual International Symposium on Computer Architecture, June 1995.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- S. Adve and K. Gharachorloo, [Shared memory consistency models: a tutorial](#), in Technical Report WRL-TR 95/7, Digital Western Research Laboratory, September, 1995.
- Carpinelli, J.D. Computer Systems Organization & Architecture Addison Wesley, 2001
- Hennesy, J.L. et. al. Computer architecture a quantitative approach (2ed: 1996)
- Computer architecture: fundamentals and principles of computer design, por Joseph D. Dumas II, Joseph D. Dumas. Publicado por CRC Press, 2006.
- Computer architecture: a quantitative approach, por John L. Hennesy, David A. Patterson, Andrea C. Arpaci-Dusseau, Andrea C. Arpaci-Dusseau. Publicado por Morgan Kaufmann, 2007.
-

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Algoritmos
CODIGO	CC262
CREDITOS	04 (cuatro)
PRE-REQUISITO	CM254, CC102
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03), PRACTICA(03)/LABORATORIO(03)

OBJETIVO DEL CURSO

- Desarrollar su capacidad de abstracción.
- Introducir y desarrollar estructuras y algoritmos.

SUMILLA

Este es uno de los primeros cursos en la secuencia de los cursos introductorios a la informática.

Algoritmos y Estructura de Datos son dos de los pilares de la informática; cualquier profesional del área, necesita desarrollar sus programas en forma algorítmica para concretizar sus modelos y propuestas.

Este curso introducirá a los participantes en los conceptos fundamentales de este arte.

El curso también ofrecerá una introducción al contexto histórico y social de la informática y una revisión del ámbito de esta disciplina

Contribución a los resultados

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- a) Habilidad para aplicar conocimiento de Computación y de matemática apropiadas para la disciplina.
- b) Habilidad para analizar un problema e identificar y definir los requerimientos algorítmicos apropiados para su solución.
- c) Habilidad para diseñar, implementar y evaluar un sistema, proceso, componente para alcanzar las necesidades deseadas.
- d) Habilidad para trabajar efectivamente en equipos para cumplir con un objetivo común.
- e) Habilidad para usar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- f) Habilidad para aplicar los principios de desarrollo y diseño en la construcción de algoritmos de complejidad variable.

CONTENIDO ANALITICO

I Fundamentos

Introducción

1 El Rol de los Algoritmos en Computación

1.1 Algoritmos

1.2 Algoritmos como una tecnología

2 Comenzando

2.1 Clasificación por inserción

2.2 Analizando algoritmos

2.3 Diseñando algoritmos

3 Crecimiento de Funciones

3.1 Notación asintótica

3.2 Notación estándar y funciones comunes

4 Dividir-y-Conquistar

4.1 El problema del subarray máximo

4.2 Algoritmo de Strassen para multiplicación de matrices

4.3 El método de sustitución para resolver recurrencias

4.4 El método de recursividad de árbol para resolver recurrencias

4.5 El método maestro para resolver recurrencias

4.6 Prueba del teorema maestro

5 Análisis Probabilístico y Algoritmos aleatorios

5.1 El problema de contratación

5.2 Indicador de variables aleatorias

5.3 Algoritmos aleatorios

5.4 Análisis probabilístico y usos adicionales del indicador de variables aleatorias

II Clasificación y estadísticas de orden

Introducción

6 Heapsort

6.1 Pila

6.2 Manteniendo la propiedad de pila

6.3 Building a heap

6.4 El algoritmo heapsort

6.5 Prioridad de colas

7 Quicksort

7.1 DESCRIPCION de quicksort

7.2 Rendimiento de quicksort

7.3 Una versión aleatoria de quicksort

7.4 Análisis de quicksort

8 Ordenamiento en tiempo lineal

8.1 Límites inferiores para ordenamiento

8.2 Ordenamiento por conteo

8.3 Ordenamiento de base

8.4 Ordenamiento de cubo

9 Medianas y estadísticas de orden

9.1 Mínimo y máximo

9.2 Selección en tiempo lineal esperado

9.3 Selección en el peor de los casos de tiempo lineal

III Diseño Avanzado y Técnicas de Análisis

Introducción

15 Programación dinámica

15.1 Barra de corte

15.2 Cadena de multiplicación de matrices

15.3 Elementos de programación dinámica

15.4 Subsecuencia común mas larga

15.5 Optimo árbol de búsqueda binaria

16 Algoritmos golosos

16.1 Un problema de selección de actividad

16.2 Elementos de la estrategia golosa

16.3 Códigos de Huffman

- 16.4 Métodos matroides y codiciosos
- 16.5 Un problema de programación de tareas como un matroide

17 Análisis amortizado

- 17.1 Análisis agregado
- 17.2 El método de contabilidad
- 17.3 El método potencial
- 17.4 Tablas dinámicas

IV Algoritmos de grafo

Introducción

22 Algoritmos de grafo elementales

- 22.1 Representaciones of grafos
- 22.2 Búsqueda primero en amplitud
- 22.3 Búsqueda primero en profundidad
- 22.4 Ordenamiento topológico
- 22.5 Componentes fuertemente conectados

23 Árboles de expansión mínimos

- 23.1 Crecimiento de un árbol de expansión mínima
- 23.2 Los algoritmos de Kruskal y Prim

24 Caminos mas cortos de un solo proveedor

- 24.1 El algoritmo de Bellman-Ford
- 24.2 Caminos mas cortos de un solo proveedor en grafos dirigidos acíclicos
- 24.3 Algoritmo de Dijkstra
- 24.4 Diferencia de restricciones y caminos mas cortos
- 24.5 Pruebas de propiedades de caminos mas cortos

25 Todos los pares de rutas mas cortas

- 25.1 Caminos mas cortos y multiplicación de matrices
- 25.2 El algoritmo de Floyd-Warshall
- 25.3 El algoritmo de Johnson para grafos dispersos

26 Flujo Máximo

- 26.1 Flujo de redes
- 26.2 El método de Ford-Fulkerson
- 26.3 Máxima coincidencia en grafos bipartitos
- 26.4 Algoritmo de re-etiquetar
- 26.5 Algoritmos de re-etiquetar hacia adelante

BIBLIOGRAFÍA

T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein. Introducción to Algoritmos, 3rd Edition. MIT Press.
ISBN-10: 0-262-03384-4, ISBN-13: 978-0-262-03384-8

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Algoritmos Paralelos
CODIGO	CC301
CREDITOS	04 (cuatro)
PRE-REQUISITO	CM094, CC262
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03), PRACTICA(03)/LABORATORIO(03)

OBJETIVO DEL CURSO

- Demuestra que el uso de programación Funcional hace que el aprendizaje de algoritmos sea mucho más fácil.
- Se enseña Haskell y las librerías de C para programar en paralelo.

SUMILLA

El curso cubre una serie de algoritmos desde el punto de vista de programación.

CONTENIDO ANALITICO

CC301 Algoritmos Paralelos	
<ul style="list-style-type: none">▪ Introducción: Algoritmos y Lenguaje Funcional▪ Haskell: ecuaciones y funciones, tipo básicos y tipos construidos de los básicos, Listas, técnicas de programación Funcional de alto orden, polimorfismo y tipos algebraicos, arreglos, clase de métodos y tipo de clases▪ La eficiencia de programas Funcionales: reducción de orden, análisis de eficiencia en los programas Funcionales, y transformación▪ Tipo de datos: Listas, árboles, y arrays.▪ Tipo de datos abstractos: Stacks, colas, colas con prioridad, conjuntos, tablas, búsqueda en árboles binarios, Heaps, árboles AVL.	<ul style="list-style-type: none">▪ Sorting: sorteo en base a comparación, algoritmos básicos de sorting, y Sorting basados en árboles.▪ Algoritmos de grafos: Búsqueda siguiendo depth-first and breath-first, mínimo spanning tree, búsqueda en árboles y bosques siguiendo depth-first.▪ Diseño de arriba hacia abajo: Dividir y conquistar, búsqueda siguiendo backtracking algoritmo, búsqueda siguiendo priority-first search, y greedy algoritmos.▪ Programación dinámica: función de higher orden, multiplicaciones matriciales encadenadas, búsqueda binaria optima para árboles, y mínima trayectoria.▪ Tópicos avanzados. Procesos de redes, Monads y algoritmo paralelos

BIBLIOGRAFIA

Algorithms: A Functional Programming Approach por Fethi Rabhi y Guy Lapalme

<http://www.cs.cmu.edu/~scandal/cacm/cacm2.html>

Kumar, V; Grama, A.; Gupta, A.; Karypis, G.. Introduction to Parallel Computing, Second Edition. Addison Wesley, 2003

PVM Parallel Virtual Machine. A Users' Guide and Tutorial for Networked Parallel Computing. Geist, A.; Beguelin, A; Dongarra, J. et alt.. The MIT Press, 1994

P. S. Pacheco. Parallel Programming with MPI. Morgan Kaufmann Publishers 1997

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Bases de Datos
CODIGO	CC331
CREDITOS	04 (cuatro)
PRE-REQUISITO	CM094, CC262
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Introduce los conceptos y la evolución del sistema de base de datos.
- Cubre los aspectos de concurrencia, confiabilidad y búsqueda de información usando SQL lenguaje.

SUMILLA

El curso cubre el sistema de base de datos relacional desde sus inicios, por qué fue importante crearlo y los problemas que hoy en día se enfrenta. Se discute la arquitectura, diseño, e implementación de una base de datos relacional.

CONTENIDO ANALITICO

Modelo de Entidad relación (E/R)

Introducción al modelo de datos ER. Conjuntos de entidades y de relaciones. Dominios. Representaciones equivalentes de una relación. Cuestiones de diseño. Especialización. Agregación. Reducción a tablas

El modelo relacional. Conversión de E/R

Introducción. Esquemas, Tuplas, Tablas, Dominios. Conversión a tablas desde un modelo con relaciones. Conversión a tablas desde un modelo con generalización. Descubrimiento de llaves en las relaciones.

Dependencias Funcionales

Definición. Axiomas de Armstrong. Reglas Adicionales. Cerradura de un conjunto de atributos. Recubrimiento no redundante. Determinación de las claves de un esquema y cálculo de las claves de un esquema R.

Normalización. Dependencias de múltiple valores.

Redundancia, Anomalías de actualización y Eliminación. Primera y Segunda forma Normal. Descomposición sin Pérdida. Preservación de dependencias. Forma Normal de Boyce-Codd(BCNF). Tercera forma Normal.BCNF vs 3NF. Cuarta y Quinta forma Normal.

Álgebra Relacional

Definiciones. Operadores algebraicos relacionales: Primitivos(proyección, selección, unión, diferencia, producto cartesiano, renombre), No Primitivos(intersección, división, ensamble natural, ensamble con selección). Expresividad del álgebra relacional. Re-nombramiento Modificaciones de las bases de datos (eliminación, inserción, actualización). Operadores extendidos.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Introducción a SQL

Aplicaciones de las bases de datos. Sistemas de bases de datos frente a sistemas de archivos. Visión de los datos. Modelos de bases de datos. Lenguajes de bases de datos. Gestión de transacciones. Estructura de un sistema de bases de datos. Usuarios de bases de datos

Agregaciones, modificaciones, y operaciones avanzadas

Agregaciones y diseño de agregaciones. Modificación de datos. Clasificación: Insertar tuplas en una relación, Eliminar tuplas de una relación y Actualizar el valor de algunos componentes de tuplas existentes.

Vistas y definición de datos

Concepto de vista. Aplicaciones de vistas. Vistas en SQL. Sintaxis. Motivos por que una lista no es actualizable. Vista sobre una tabla Básica. Vista sobre una concatenación de relaciones. Definir tablas Filas y columnas. Insertar claves de índice. Crear relaciones entre tablas. Asignar tipos de datos.

Restricciones y disparadores (triggers)

Restricciones en procedimientos almacenados y disparadores. Restricciones en subconsultas. Restricciones en vistas. Definición, uso y sintaxis de trigger. Componentes y nombres de disparadores. Tipos de disparadores. Orden de activación de disparadores.

Módulos de almacenamiento persistentes

Almacenamiento y estructura de archivos. Indexación y asociación Procesamiento de consultas. Optimización de consultas.

Oracle. PL/SQL

Conceptos Preliminares. Nociones básicas de PL/SQL. Estructuras de Control. Los cursores. Manejo de errores. Procedimientos almacenados. Funciones y Triggers. Los paquetes.

Sistemas embebidos SQL. CLI, y JDBC

Introducción al desarrollo de Aplicaciones con Bases de Datos. SQL embebido SQL Dinámico. Acceso a bases de datos: ODBC, SQLJ, SQL/CLI. Entornos 4GL y bases de datos. Otras tecnologías para acceso a datos: JDBC y ADO.NET

Autorización

Violaciones de la seguridad. Control de acceso a la base de datos. Tipos de autorización. Autorizaciones y vistas. Concesión de privilegios. Eliminación de privilegios. El concepto de rol o papel. Limitaciones de la autorización SQL.

Transacciones

Introducción. Sentencias para una transacción. Transacciones anidadas. Ejemplos. Transacciones y procedimientos almacenados.

Sistemas relacionales-OO

Base de datos relacionales. Diseño de bases de datos relacionales. Bases de datos Orientadas a Objetos: ¿Qué es O.O.?, ¿Por qué O.O.?, ¿Qué es una BDOO?.

Datalog. Base de Datos orientados a objetos.

Arquitectura de Una BDOO. Desarrollo con Bases de Datos OO. Tres Enfoques de Construcción de Bases de Datos OO. Impacto de la Orientación a Objetos en la Ingeniería del Software. Ventajas en BDOOs. Rendimiento

Data y XML semi-estructurados

Modelo de Datos Semiestructurados. Estructura de los datos en XML. HTML vs. XML. Sintaxis básica de XML. DTD y XML-Schema.

Query en Datos y XML semi-estructurados

Introducción al XML. XML Query Data Model: Introducción, Conceptos, Types , Nodos, etc. XML Query Algebra XML Query(el lenguaje de consulta).

Warehouse y Minería de datos

Data Warehouse. Introducción a los Almacenes de Datos. Introducción a la minería de datos. Explotación de almacenes de datos herramientas OLAP. Problemas de Clasificación. Aprendizaje supervisado. Métodos de agrupamiento. Aprendizaje no supervisado. Diseño de almacenes de datos. Mantenimiento de almacenes de datos. Introducción a la Minería de Datos (DM). Técnicas de Minería de Datos. Web Mining.

BIBLIOGRAFIA

- Database Systems: The Complete Book (DS:CB), by Hector García-Molina, Jeff Ullman, and Jennifer Widom. Editorial: Upper Saddle River, N.J. : Pearson Prentice Hall, ©2009.
- Introducción a los Sistemas de Base de Datos. C. J. Date : Pearson Prentice Hall, ©2001.
- Bases de datos relacionales: diseño físico (Orientado al DB2 para z/OS de IBM). Enrique Rivero Cornelio, Carlos Guardia Rivas, José Carlos Reig Hernández. Editor Universidad Pontificia de Comillas de Madrid, 2004
- Abiteboul,S; Hull and Vianu, V. ; Foundations of Databases ; Addison-Wesley Publishing Company, 1995.
- Ullman, Jeffrey D.; Principles of Database and Knowledge Base Systems, Vol I Computers Science Press, 1988.
- Mendelzon, A. "Introducción a las Bases de Datos Relacionales" Ed. Pearson, 2000
- Silberschatz, A.; Korth, H.F.; Sudarshan, S. “Fundamentos de Bases de Datos”. 3ª edición. Madrid, McGraw-Hill, 1998.
- Inmon, W.H., “Building the Data Warehouse” , John Wiley, 2002.
- Elmasri, R.; Navathe, S.B. “Fundamentals of Database Systems” 3rd Edition, Addison-Wesley, 2000, cap. 26.
- Berry, M.: “Data mining techniques”. 1997.
- BRAY, T.; HOLLANDER, D.; LAYMAN, A. «Name-spaces in XML.» World Wide web Consortium. W3C Recommendation, 1999.
- Urman, S. ORACLE 8. Programación en PL/SQL. Osborne McGraw-Hill, Madrid, 1998.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Sistemas Operativos
CODIGO	CC361
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC212
CARACTER	obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Conocer los elementos básicos del diseño de los sistemas operativos
- Presenta la administración del sistema Linux como primer paso para trabajar en software abierto
- Presenta algunas de las herramientas de software abierto para que eventualmente sea parte de un proyecto de software abierto.

SUMILLA

Describe los elementos básicos de los sistemas operativos, como la calendarización y despacho, administración de memoria y de dispositivos. Asimismo se ve la administración de Linux, comandos, shells, administración del kernel, sistema de archivos. Se muestra el uso de backup en forma periódica y automatizada. Emacs, vi, Latex, y cvs.

CONTENIDO ANALITICO

4.1 Algoritmos Distribuidos.

Tópicos I: Consenso y elección. II: Detección de terminación.

Objetivos. 1. Explicar el paradigma distribuido. 2. Explicar un algoritmo distribuido simple. 3. Determinar cuando usar los algoritmos de consenso o elección.

4.2 Visión General de los Sistemas Operativos.

Tópicos I: Rol y propósito de los sistemas operativos. II: Historia del desarrollo de los sistemas operativos. III: Funcionalidad de un sistema operativo típico IV: Mecanismos de soporte a modelos cliente-servidor, dispositivos hand-held. V: Tópicos de Diseño (eficiencia, robustez, flexibilidad, portabilidad, seguridad, compatibilidad). VI: Influencias de la seguridad, redes, Multimedia, ventanas.

Objetivos. 1. Explicar los objetivos y funciones de los sistemas operativos modernos. 2. Describir como los sistemas operativos han evolucionado en el tiempo desde sistemas primitivos batch a sofisticados sistemas multiusuarios. 3. Analizar las ventajas y desventajas inherentes en el diseño de sistemas operativos. 4. Describir las funciones de un sistema operativo contemporáneo con respecto a la conveniencia, eficiencia y la habilidad para evolucionar. 5. Discutir sistemas operativos de tipos: distribuido, para redes y cliente-servidor, y como ellos difieren de un sistema operativo para un usuario. 6. Identificar las amenazas potenciales a sistemas operativos y las características de seguridad diseñadas para resguardarlos.

4.3 Principios de los Sistemas Operativos.

Tópicos I: Métodos Estructurados (monolíticos, por capas, modulares, modelos de microkernel). II: Abstracciones, procesos y recursos. III: Conceptos de APIs. IV: Necesidad de las aplicaciones y evolución de las técnicas de hardware y software. V: Organización de dispositivos. VI: Interrupciones: métodos e implementaciones. VII: Concepto de estado de usuario/sistema e y protección, transición al modo Kernel.

Objetivos. 1. Explicar el concepto de una capa lógica 2. Explicar los beneficios de la construcción de capas abstractas en las formas jerárquicas 3. Defender la necesidad de APIs y middleware. 4. Describir como los recursos computacionales son utilizados por software de aplicación y o administrados por software del sistema. 5. Contrastar el modo usuario y el modo kernel en un sistema operativo. 6. Discutir las ventajas y desventajas de utilizar procesamiento de interrupciones. 7. Comparar y contrastar las diversas formas de estructurar un sistema operativo tales como orientado a objetos, modular, microkernel, y por capas.

4.4 Concurrencia.

Tópicos I: Estados y diagramas de estados. II: Estructuras (lista, bloques de control de procesos, o etc.). III: Despachos y conmutación según el contexto. IV: El rol de las interrupciones. V: Ejecución concurrente: ventajas y desventajas. VI: El problema de exclusión mutua y algunas o soluciones. VII: Deadlock: causas, condiciones y prevención. VIII: Modelos y mecanismos (semáforos, monitores, variables de condición y rendezvous). IX: Problemas del consumidor o productor y sincronización. X: Problemas de Multiprocesamiento (Spin-Locks, re-entrada).

Objetivos. 1. Describir la necesidad de la concurrencia dentro de la estructura de un sistema operativo. 2. Demostrar los problemas potenciales en tiempo de ejecución originados por la operación concurrente de muchas tareas separadas. 3. Resumir la gama de mecanismos que pueden ser empleados en el nivel de sistemas operativos para entender los sistemas concurrentes y describir los beneficios de cada uno. 4. Explicar los diferentes estados que una tarea podrá atravesar y las estructuras de datos necesarias para soportar la administración de muchas tareas. 5. Listar o los diferentes métodos para resolver el problema de la exclusión mutua en un sistema operativo. 6. Describir las razones por las cuales utilizar interrupciones, despachos y cambio de contexto para soportar la concurrencia en un sistema operativo. 7. Crear estados y diagramas de transición para el dominio de problemas simples. 8. Discutir la utilidad de estructuras de datos, tales como pilas y colas en el manejo de concurrencia.

4.5 Calendarización y Despacho.

Tópicos I: Planeamiento preventivo y no preventivo. II: Planeamiento y políticas III: Procesos y hebras. IV: Deadlines y problemas de tiempo real.

Objetivos. 1. Comparar y contrastar los algoritmos comunes usados para el planeamiento preventivo y no preventivo de tareas en sistemas operativos, tales como prioridad, desempeño, comparación n o y esquemas de partición justa. 2. Describir las relaciones entre los algoritmos de planeamiento y o dominios de aplicación. 3. Discutir los tipos de planeamiento del procesador tales como: termino corto (short-term), termino medio (medium-term), termino largo (long-term) y entrada/salida. 4. Describir la diferencia entre procesos y hebras. 5. Comparar y contrastar las aproximaciones dinámicas y estáticas de planeamiento en tiempo real. 6. Discutir la necesidad de planeamiento a a preventivo y planeamiento por plazos (deadline).

4.6 Administración de Memoria.

Tópicos I: Revisión de la memoria física y de la administración de la memoria de hardware. II: Reo tardos, intercambios y particiones. III: Paginamiento y segmentación. IV: Políticas de asignación y de reemplazo. V: Conjuntos de trabajos y trashing. VI: Caching.

Objetivos. 1. Explicar la jerarquía de la memoria y el costo-desempeño en los intercambios. 2. Explicar el concepto de memoria virtual y como esta es soportada por el hardware y software. 3. Listar los principios de memoria virtual aplicados a caching, paginamiento y segmentación. 4. Evaluar los intercambios en términos de tamaño de memoria (memoria principal, memoria cache, memoria auxiliar) y velocidad del procesador. 5. Sustentar las distintas opciones de asignación de memoria a las tareas, citando las ventajas de cada una. 6. Describir las razones para la existencia y uso de la memoria cache. 7. Comparar y contrastar las técnicas de paginamiento y e segmentación. 8. Discutir el concepto de trashing, tanto en términos del porque de su ocurrencia y de las técnicas usadas para reconocer y administrar el problema.

4.7 Administración de Dispositivos.

Tópicos I: Características de dispositivos seriales y paralelos. II: Estrategias de buffering. III: Acceso directo a memoria.

Objetivos. 1. Explicar la diferencia clave entre dispositivos seriales y paralelos, e identificar las condiciones en las cuales cada uno es apropiada. 2. Identificar las relaciones entre el hardware físico y los dispositivos virtuales mantenidos por el sistema operativo. 3. Explicar el buffering y describir las estrategias de como implementarlo. 4. Diferenciar los mecanismos usados en interfaces con diversos dispositivos (incluyendo dispositivos handheld, redes, Multimedia) en una computadora y explicar las implicatorias de esto para el diseño de un sistema

operativo. 5. Describir las ventajas y desventajas del acceso directo a memoria y discutir las circunstancias en las cuales su uso es garantizado. 6. Identificar los requerimientos para la recuperación de o fallas.

4.8 Seguridad y Protección.

Tópicos I: Visión panorámica de la seguridad del sistema. II: Política y mecanismos de separación. III: Métodos de seguridad y dispositivos. IV: Protección, acceso y autenticación. V: Modelos de protección. VI: Protección de memoria. VII: Encriptación. VIII: Administración de recuperación.

Objetivos. 1. Defender las necesidades de protección y seguridad, y el rol de las consideraciones éticas en el uso de la computadora. 2. Listar las características y limitaciones de un sistema operativo usado para brindar protección y seguridad. 3. Comparar y contrastar los métodos e actuales para implementar la seguridad. 4. Comparar y contrastar las fortalezas y debilidades de dos o más sistemas operativos populares con respecto a la seguridad.

4.9 Sistema de Archivos.

Tópicos I: Archivos: datos, meta-datos, operaciones, organización, buffering, secuenciales, no secuenciales. II: Directorios: contenidos y estructura. III: Sistemas de archivos: particionamiento, montaje/desmontaje, sistemas de archivos virtuales. IV: Técnicas de implementación estándares V: Archivos mapeados en memoria. VI: Sistemas de archivos de propósito especial. VII: Nombrado, búsqueda, acceso, copias de respaldo.

Objetivos. 1. Listar la completa gama de consideraciones que soporten los sistemas de archivos. 2. Comparar y contrastar los diferentes abordajes de la organización de archivos reconociendo o las fortalezas y debilidades de cada uno.

4.10 Sistemas Dedicados y de Tiempo Real.

Tópicos I: Planeamiento de tareas y procesos. II: Requerimientos de la administración memoria/disco en un ambiente de tiempo real. III: Fallos, riesgos, y recuperación. IV: Preocupaciones especiales en sistemas de tiempo real. Objetivos. 1. Describir que hace a un sistema ser un sistema en tiempo real. 2. Explicar la presencia y describir las características de latencia en sistemas de tiempo real.

4.11 Scripting.

Tópicos I: Scripting y el rol de los lenguajes scripts. II: Comandos básicos del sistema. III: Creación de scripts, paso de parámetros IV: Ejecución de un script. V: Influencias del scripting en la programación.

Objetivos. 1. Resumir un conjunto de comandos típicos proveídos por el sistema operativo. 2. Demostrar la Funcionalidad típica de un lenguaje script e interpretar las implicaciones para la programación. 3. Demostrar los mecanismos de implementación de scripts y el rol de los scripts en la implementación del sistema y su integración.

4.12 Administración de sistemas Linux/UNIX

Tópicos I: Introducción. II: Línea de comando en UNIX. III: Procesamiento de streams de texto usando filtros. IV: Sistema de archivos. V: Uso de UNIX streams, pipes, y re-direcciones VI: Búsqueda de archivo de textos usando expresiones regulares. VII: Job control. VIII: Creación de procesos, monitoreo de procesos, y eliminación de procesos. IX: Modificación de prioridades de ejecución de los procesos. X: Uso de shells. XI: Conceptos del sistema de archivos. XII: Creación y cambio de links simbólicos. XIII: Permisos de pertenencia de los archivos. Usos de permisos para controlar el acceso a los archivos.

Objetivos: Presenta la administración del sistema Linux como primer paso para trabajar en software abierto

4.13 Herramientas para administración de sistemas Linux/UNIX

Tópicos I: Creación de particiones y sistemas de archivos. II: Manejo y control para montar y desmontar un sistema de archivos. III: Manutención de integridad de los archivos. IV: Búsqueda de sistemas de archivos y el movimiento de sistemas de archivos. V: Monitorear el uso de los discos. VI: Iniciar el sistema. VII: Cambio de los niveles de ejecución, apagado del sistema, re-iniciamiento del sistema. VIII: Uso y manejo de documentación del sistema. IX: Búsqueda de documentación de Linux en el sistema como en la Internet. X: Modificación del environment y manejo de variables de environment de sistema. XI: Configuración y uso de log files del sistema. X: Automatización y despachador de tareas administrativas. XI: Backup. Herramientas en software abierto.

Objetivos: Presenta algunas de las herramientas de software abierto para que eventualmente sea parte de un proyecto de software abierto.

BIBLIOGRAFIA

- Mateu, L. (1999). Apuntes de Sistemas Operativos. Universidad de Chile.
- Stallings, W. (2005). Operating Systems: Internals and Design Principles, 5/E. Prentice Hall.
- Tanenbaum, A. S. (2001). Modern Operating Systems, 2/E. Prentice Hall.
- Tanenbaum, A. S. (2006). Operating Systems Design and Implementation, 3/E. Prentice Hall.
- <http://svnbook.red-bean.com/nightly/en/svn-book.html>
- http://sources.redhat.com/autobook/autobook/autobook_toc.html
- <http://tldp.org/LDP/lame/LAME/linux-admin-made-easy/>
- <http://www.yolinux.com/TUTORIALS/LinuxTutorialSysAdmin.html>
- Linux Administration: A Beginner's Guide, Wale Soyinka, McGraw Hill, Fifth Edition.
- Linux Administration Handbook, Evi Nemeth & Garth Snyder & Trent R. Hein, Prentice Hall, 1st edition.
- Linux in a Nutshell, Fourth Edition , A Desktop Quick Reference, Ellen Siever, Stephen Figgins, Aaron Weber, O'Reilly Media
- LPI Linux Certification in a Nutshell, Second Edition, Steven Pritchard, Bruno Gomes Pessanha, Nicolai Langfeldt, James Stanger, Jeff Dean, et al., O'Reilly Media

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Lenguaje Paralelo Distribuido Orientado a Objetos
CODIGO	CC302
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC201, CC301
CARACTER	obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

Demuestra que existe lenguajes orientado a objetos que son distribuidos, paralelizados, y que son accesibles con Java (X10)

SUMILLA

El curso enseña a programar en X10. El lenguaje es diseñado para alto rendimiento, alta productividad, siendo capaz de ejecutar 100000 hilos en forma concurrente.

CONTENIDO ANALITICO

CC302 Lenguaje de Paralelo Distribuido Orientado a Objetos	
<ul style="list-style-type: none">▪ Introducción▪ X10 – Overview▪ Estructura Lexical▪ Tipos▪ Variables▪ Objetos▪ Paquetes y reglas de nombres▪ Convenciones de nombres▪ Interfaces▪ Interfaces con propiedades▪ Clases▪ Declaración de variables▪ Statements	<ul style="list-style-type: none">▪ Expresiones▪ Lugares▪ Actividades▪ Reloj▪ Operaciones de reloj▪ Arrays▪ Plugin compilers▪ Linking con código nativo

BIBLIOGRAFIA

- <http://dist.codehaus.org/x10/documentation/languagespec/x10-170.pdf>
- David Bacon. Kava: A Java dialect with a uniform object model for lightweight classes. Concurrency – Practice and Experience, 15:185–206, 2003.
- Joseph A. Bank, Barbara Liskov, and Andrew C. Myers. Parameterized types and Java. In Proceedings of the 24th Annual ACM Symposium on Principles of Programming Languages (POPL'97), pages 132–145, 1997.

- J. Gosling, W. Joy, G. Steele, and G. Bracha. The Java Language Specification. Addison Wesley, 2000.
- Jose E. Moreira, Samuel P. Midkiff, Manish Gupta, Pedro V. Artigas, Marc Snir, and Richard D. Lawrence. Java programming for high-performance numerical computing. IBM Systems Journal, 39(1):21–, 2000.
- A. Skjellum, E. Lusk, and W. Gropp. Using MPI: Portable Parallel Programming with the Message Passing Interface. MIT Press, 1999.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Computación Gráfica
CODIGO	CC332
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC301
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

Introducir un entendimiento general de los sistemas gráficos 2D y 3D y una experiencia en profundidad con un tópico relacionado al área de gráficos por computadora.

DESCRIPCION

Las tecnologías de visualización, animación y simulación tridimensional son de uso común en la sociedad actual. Entender las técnicas y métodos que fundamentan estas tecnologías es la meta de este curso.

CONTENIDO ANALÍTICO

1. Introducción. DESCRIPCION del curso, definición y motivación para estudiar y desarrollar software gráfico, áreas de aplicación y mercado.
2. Sistemas Gráficos. Dispositivos de salida. Tecnologías: CRTs, LCDs, DMD/DLP, OLED, Display Walls, Plasma, Estéreo
3. Conceptos Básicos de OpenGL. Conceptos Básicos de OpenGL. Convenciones. Sistema de ventanas - GLUT. Sistemas de coordenadas. Modelo de Cámara. Primitivas básicas. Interacción con el mouse y teclado.
4. Primitivas geométricas. Puntos y vectores. Operadores: producto interno, producto cruz y normas. Pipeline gráfico: mover modelos, trasladar, iluminar.
5. Imágenes 2D. Texturas. Mapas procedimentales Mapas UVW. Mapas de textura. Mapas de reflexión. Bump map. Light map. Mip map.
6. Geometría. Coordenadas homogéneas. Geometría Euclidiana, afín y proyectiva.
7. Interfaces de usuario. Modelos WIMP, consola y manipulación directa. Toolkits: GLUI, FLTK, Qt
8. Transformaciones geométricas. Transformaciones geométricas 2D/3D. Sistemas de coordenadas. Representaciones matriciales. Composición de matrices.

9. Proyecciones. Escenas 3D, GL_MODELVIEW y GL_PROJECTION. Pila de matrices. Sistema local y global y proyecciones. Pipeline gráfico
10. Curvas. Continuidad. Interpolación y aproximación. Algoritmo de Casteljaou. Curvas de Bezier
11. Modelamiento: Sistemas de modelamiento. Representación y especificación de superficies. Técnicas de representación. Representación por mallas de polígonos. Técnicas de modelamiento.
12. Poligonización. Operaciones básicas: muestreo y estructuración. Algoritmo Marching Cubes. Métodos de Poligonización
13. Mallas. Mallas de polígonos, 2-manifold y no manifold. Mallas orientables con/sin borde. Operaciones en mallas. Estructuras de datos para mallas
14. Iluminación: Z-buffer. Algoritmo de eliminación de faces o Culling. Iluminación. Modelo de tonalización. Modelo de iluminación. Fuentes de luz.

BIBLIOGRAFIA

- HEARN, Donald D.; BAKER, M. Pauline . 2003 Computer Graphics with OpenGL. 3 edition. Pearson Education.
- SHREINER, Dave; WOO, Mason; NEIDER, Jackie; DAVIS, Tom . 2007 OpenGL Programming Guide: The Official Guide to Learning OpenGL. 6 edition. Addison-Wesley Professional.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Teoría de la Computación
CODIGO	CC342
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC301, CC361
CARÁCTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Demuestra que la Computación es una ciencia, en particular una rama de la matemática que centra su interés en el estudio y definición formal de los computadores.

SUMILLA

El curso muestra que la Computación es la obtención de una solución o resultado a partir de ciertos datos o entradas utilizando para ello un proceso o algoritmo.

CONTENIDO ANALITICO

CC342 Teoría de la Computación	
<ul style="list-style-type: none">▪ Introducción teoría de la Computación<ol style="list-style-type: none">1. Autómatas, Computabilidad y Complejidad2. Nociones Matemáticas de Teoría de la Computación• Conjuntos TC• Funciones y Relaciones TC• Cadenas y Lenguajes<ol style="list-style-type: none">3. Inducción Matemática▪ Lenguajes Regulares<ol style="list-style-type: none">1. Autómatas Finitos• Autómatas Finitos Determinísticos• Autómatas Finitos no Determinísticos<ol style="list-style-type: none">2. Expresiones Regulares3. Lenguajes no Regulares▪ Lenguajes libres de contexto<ol style="list-style-type: none">1. Gramáticas Libres de Contexto2. Árboles de Derivación3. Formas Normales de Chomsky4. Formas Normales de Greibach5. Eliminación de Factores Comunes izquierdos6. Eliminación Recursividad Izquierda7. Eliminación de ambigüedad	<ul style="list-style-type: none">▪ Máquina de Turing<ol style="list-style-type: none">1. Definición formal de máquina de Turing2. Construcción Modular máquina de Turing3. Lenguajes aceptados por máquina de Turing4. Variantes máquina de Turing5. Problemas de Hilbert▪ Decibilidad<ol style="list-style-type: none">1. Lenguajes Decibles2. Los problemas de Halting3. Teorías Lógicas de decibilidad▪ Reducibilidad<ol style="list-style-type: none">1. Problemas Insolubles en la teoría de lenguajes2. Un problema simple que es insoluble3. Funciones Computables4. Reducibilidad de Turing

8. Autómatas PushDown 9. Lenguaje no Regulares	
---	--

BIBLIOGRAFIA

- Hopcroft, J.E.; Motwani, R. i Ullman, J.D. *Introducción to Automata Theory, Languages, and Computation* Addison-Wesley, 2001 (2a ed.).
- Kelley, Dean *Automata and Formal Languages* (Hi ha trad. a l'espanyol de la mateixa editorial), Prentice Hall, 1995.
- Sipser. *"Introducción to the Theory of Computation"*. PWS, 1997.
- Brookshear J "Teoría de la Computación," Addison Wesley, 1993.
- J. Glenn Brookshear *Teoría de la Computación : lenguajes formales, autómatas y complejidad*, Addison-Wesley Iberoamericana, 1993.
- Joaquim Gabarró Valles *Informatica classica : automatas gramatiques, indecidibilidad, paralelismo masivo*, 1995.
- Jozef Gruska *Foundations of computing*, International Thomson Computer Press, 1997.
- Efim Kinber, Carl Smith *Theory of computing : a gentle Introduction*, Prentice-Hall, 2001.
- Dexter C. Kozen. *Automata and computability*, Springer-Verlag,, 1997.
- Harry R. Lewis, Christos H. Papadimitriou *Elements of the theory of computation*, Prentice-Hall,, 1998.
- Dan A. Simovici, Richard L. Tenney *Theory of formal languages with applications*, World Scientific, 1999.
- Joan Vancells i Flotats *Teoría d'autòmats i llenguatges formals I*, Universitat Oberta de Catalunya, 2000.
- Derik Wood *Theory of computation*, Harper & Row, 1987.
- Rafel Cases, Lluís Màrquez *Llenguatges, gramàtiques i autòmats : curs bàsic*, Edicions UPC, 2003.
- John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman *Introduction to automata theory, languages, and computation*, Pearson Education International, 2003.
- Dean Kelley *Automata and formal languages : an Introduction*, Prentice Hall, 1995.
- Maria Serna [et al.] *Els Límits de la Computación : indecidibilitat i NP-completesa*, Edicions UPC, 2001.
- Michael Sipser *Introducción to the theory of computation*, Thomson Course Technology, 2006.
- <http://www.uade.edu.ar/PlanesYProgramas/VerContenido.aspx?Id=54&CodMateria=104567>
- http://www.ort.edu.uy/index.php?cookie_setted=true&id=AAAHAB&cdpto=62
- <http://biblioteca2.icesi.edu.co/cgi-olib?session=99143311&infile=details.glu&loid=7178&rs=204013&hitno=8>
- <http://www.fib.upc.edu/es/infoAca/estudis/assignatures/TC.html>
- http://ceidis.ula.ve/cursos/pgcomp/teo_comp/cont1.html
- http://www.ehu.es/p200-content/es/contenidos/assignatura/fut_16337infor302_226_2_x/es_16337/es_fileasig_16337.html

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Computación Centrada en Redes
CODIGO	CC362
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC361
CARACTER	obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

Introduce en la estructura, implementación, y fundamentos teóricos de la red de computadoras y las aplicaciones que se han habilitado por esta tecnología.

CONTENIDO ANALITICO

- Comunicación y trabajo en red: Estándares de red y organismos de normalización; el modelo de referencia ISO de 7 capas, en general, y sus instancias en TCP/IP; conmutación de circuitos y conmutación de paquetes; arroyos y datagramas; conceptos de capa física de red; conceptos de capa de enlace de datos; Interconexión y encaminamiento; servicios de la capa de transporte
- La web como un ejemplo de Computación cliente-servidor: Tecnologías web; características de servidores web; rol del computador cliente; naturaleza del relacionamiento cliente-servidor; protocolos web; herramientas de soporte para creación de web-site y administración web; desarrollando servidores de información de Internet; publicación de la información y aplicaciones.
- Creación de aplicaciones Web: Protocolos de la capa de aplicación; principios de ingeniería web; sitios web con base de datos; llamadas a procedimientos remotos; pequeños objetos distribuidos; el papel del middleware; herramientas de apoyo; los problemas de seguridad en los sistemas de objetos distribuidos; aplicaciones basadas en web en toda la empresa.
- Gestión de redes: Revisión de los asuntos de gestión de la red; temas para los proveedores de servicios de Internet; las cuestiones de seguridad y cortafuegos; la calidad de los problemas de servicio
- Compresión y descompresión: Revisión de la compresión de datos básicos; compresión y descompresión de audio; compresión y descompresión de imágenes; compresión y descompresión de vídeo; los problemas de rendimiento
- Tecnologías de datos Multimedia: Revisión de las tecnologías Multimedia; estándares Multimedia; capacidad de planificación y problemas de rendimiento; dispositivos de entrada y salida; teclados MIDI, sintetizadores; estándares de almacenamiento; servidores Multimedia y sistemas de archivos; herramientas para apoyar el desarrollo Multimedia
- Computación inalámbrica y móvil: Visión general de la historia, evolución, y compatibilidad de los estándares inalámbricos; los problemas especiales de Computación inalámbrica y móvil; redes inalámbricas de área local y

redes por satélite; bucles locales inalámbricos; el protocolo de Internet móvil; adaptación consciente de móviles; extendiendo el modelo cliente-servidor para dar cabida a la movilidad; acceso de datos móviles; los paquetes de software para Computación móvil e inalámbrica; el papel de las herramientas de middleware y de apoyo; problemas de rendimiento; las nuevas tecnologías

REFERENCIAS

1. Academia cisco Networking, C. P. A. (2004). GUIA DEL PRIMER ANO CCNA 1 Y 2. PEARSON EDUCACION.
2. Gast, M. (2008). 802.11 Wireless Networks: The Definitive Guide. O'reilly, second edition edition.
3. McNa, C. (2008). Network Security Assessment, Know Your Network. Oreilly, second edition edition.
4. Stalling, W. (2005). Comunicaciones y Redes de Computadoras. Prentice Hall, 7ma edition. 8420541109.
5. Stevens, R. (2005). TCP/IP and Protocols Implementation. Addison-Wesley, 2005.
6. Stevens, R. (2006). TCP/IP and Protocols. Addison-Wesley, 2006.
7. Tanenbaum, A. S. (2005). Redes de Computadoras. Prentice-Hall, 7th edition.
8. Wasserman, M. (2008). Engineer Task Force. <http://edu.ietf.org/>.

UNIVERSIDAD NACIONAL DE INGENIERIA
 FACULTAD DE CIENCIAS
 ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Programación de Aplicaciones en Redes
CODIGO	CC401
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC302, CC362
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03), LABORATORIO (03)

OBJETIVO DEL CURSO

- Demuestra que una red de computadores es un sistema para la comunicación entre ellos
- Se describe los protocolos IP, TCP, y UDP y el diseño e implementación de algunos de los protocolos para tener una visión de sus cualidades y desventajas.

SUMILLA

La aparición de nuevos sistemas en los cuales existen múltiples núcleos en un solo chip, requiere la construcción de una nueva arquitectura para su uso.

CONTENIDO ANALITICO

CC401 Programación de Aplicaciones de Redes	
▪ Redes de computadoras	– Myrinet
▪ Categorización	– QsNet
– Relación Funcional	– RS-232
– Topología de Redes	– SPX
– Especialización por	– System Network Architecture
Funcionalidad	– Token Ring
▪ Pilas de protocolos	– TCP
– ARCNET	– USB
– AppleTalk	– UDP
– ATM	– X.25
– Bluetooth	▪ Tópicos Especiales
– DECnet	– Internet
– Ethernet	▪ Backbone
– FDDI	▪ SITA y eQuant
– Frame relay	▪ Transit
– HIPPI	▪ Stub
– IEEE 1394 aka FireWire, iLink	– Capas
– IEEE 802.11	▪ Transmisión de Datos
– IEEE-488	▪ Transmisión por cables
– IP	▪ Transmisión inalámbrica
– IPX	– Monitoreo de Redes

BIBLIOGRAFIA

- UNIX Network Programming: Networking APIs: Sockets and XTI; por Richard Stevens Vol I Editorial: Upper Saddle River, NJ : Prentice Hall PTR, 1998.
- http://en.wikipedia.org/wiki/Computer_networking
- http://en.wikipedia.org/wiki/Protocol_stack
- Adventures in UNIX Network Applications Programming; Por Bill Rieken y Lyle Weiman. Publicado John Wiley & Sons, Inc.
- Network Application Programming Interface over Session Initiation Protocol, a novel approach to the global roaming environment; Por Tzu-Chi Huang , Ce-Kuen Shieh , Bo-Yang Lai y Yu-Ben Miao. Publicado Elsevier Science Publishers B. V
- Computer Networks; Por Andrew S. Tanenbaum. Publicado por Prentice Hall

SÍLABO

I. DESCRIPCION DEL CURSO

CURSO	Inteligencia Artificial
CODIGO	CC441
CREDITOS	04 (cuatro)
PRE-REQUISITO	CM274, CC301
CARÁCTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

II. OBJETIVO DEL CURSO

Este curso cubre tópicos fundamentales en sistemas inteligentes:

Representación del conocimiento y razonamiento, búsqueda avanzada del conocimiento y razonamiento. Evaluar las posibilidades de simulación de la inteligencia, para lo cual se estudiarán las técnicas de modelización del conocimiento.

Construir una noción de inteligencia que soporte después las tareas de su simulación.

III. SUMILLA

La investigación en Inteligencia Artificial ha conducido al desarrollo de numerosas técnicas relevantes, dirigidas a la automatización de la inteligencia humana, dando una visión panorámica de diferentes algoritmos que simulan los diferentes aspectos del comportamiento y la inteligencia del ser humano.

El curso usará el lenguaje Lisp para el estudio e implementación de los algoritmos que demostrarán el aprendizaje a través del tiempo, con la llegada de nuevos datos, y las modificaciones que se tiene que hacer al algoritmo (creación o adición de nuevas reglas).

IV. CONTENIDO ANALITICO

4.1 Tópicos Fundamentales en Sistemas Inteligentes

Tópicos. I: Historia de la inteligencia artificial II: Cuestiones filosóficas. III: La prueba de Turing. IV: Experimento de pensamiento del “Cuarto Chino” de Searle. V: Temas típicos en IA. VI: Definiciones fundamentales. VII: Comportamiento óptimo vs. comportamiento actuando como humano. VIII: Razonamiento óptimo vs. razonamiento actuando como humano. IX: Preguntas filosóficas. X: Modelando el mundo. XI: El rol de la heurística.

Objetivos. 1. Describir la prueba de Turing y el experimento de pensamiento del “Cuarto Chino”. 2. Diferenciar los conceptos de razonamiento óptimo y razonamiento actuando como humano. 3. Diferenciar los conceptos de comportamiento óptimo y comportamiento actuando como humano. 4. Lista de ejemplos de sistemas inteligentes que dependen del modelo del mundo. 5. Describir el rol de la heurística y la necesidad de un punto medio entre la optimización y la eficiencia

4.2. Búsqueda y Satisfacción de la Restricción.

Tópicos. I: Problemas de espacio. II: Búsqueda de fuerza bruta (respiro primero, profundidad primero, profundidad primero con profundización iterativa). III: Búsqueda del mejor primero (mejor primero genérico, algoritmo de Dijkstra, A*, admisibilidad de A*). IV: Juegos de dos jugadores (búsqueda mínima, poda alfa-beta). V: Satisfacción de la restricción (backtracking o métodos de búsqueda local y seguimiento).

Objetivos. 1. Formular un eficiente problema expresado en el idioma español, caracterizando este problema en términos de estados, operadores, estado inicial, y una descripción del estado final. 2. Describir el problema de explosión combinatoria y sus consecuencias. 3. Seleccionar un algoritmo apropiado de búsqueda de fuerza bruta

para un problema, implementarlo y caracterizar sus complejidades de tiempo y espacio. 4. Seleccionar un algoritmo de búsqueda heurística para un problema, implementarlo por medio del diseño de la función de evaluación heurística necesaria. 5. Describir bajo que condiciones los algoritmos de heurística garantizan una solución óptima. 6. Implementar la búsqueda mínima con poda alfa-beta para juegos de dos. 7. Formular un problema en español utilizando un algoritmo de backtracking cronológico

4.3. Representación del Conocimiento y Razonamiento.

Tópicos. I: Inferencia no monotónica. II: Razonamiento probabilístico. III: Teorema de Bayes. Objetivos. 1. Explicar la operación de la técnica de resolución para probar teoremas. 2. Explicar la diferencia entre inferencia monotónica y no monotónica. 3. Discutir las ventajas y defectos del razonamiento probabilístico. 4. Aplicar el teorema de Bayes para determinar probabilidades condicionales.

4.4. Búsqueda Avanzada.

Tópicos. I: Algoritmos genéticos II: Simulated annealing. III: Búsqueda local.

Objetivos. 1. Explicar que son los algoritmos genéticos y contrastar su efectividad con las soluciones e de problemas clásicos y técnicas de búsqueda clásicas. 2. Explicar como simulated annealing puede ser usado para reducir la complejidad y contrastar su operación con técnicas de búsqueda clásica. 3. Aplicar técnicas de búsqueda local a un dominio clásico.

4.5. Representación Avanzada del Conocimiento y Razonamiento.

Tópicos. I: Incerteza. a) Razonamiento probabilístico. b) Redes Bayesianas. c) Conjuntos difusos y teoría de la posibilidad. d) Teoría de la decisión.

Objetivos. 1. Describir y contrastar las técnicas básicas para representar incerteza.

4.6. Agentes.

Tópicos. I: Definición de agentes. II: Arquitectura de agentes. a) Agentes reactivos simples. b) Planeadores reactivos. c) Arquitecturas de capas. d) Ejemplos de arquitecturas y aplicaciones. III: Teoría de agentes. a) Acuerdos. b) Intenciones. c) Agentes de decisión teórica. d) Procesos de decisión de Markov (PDM). IV: Agentes que aprenden. V: Sistemas multiagente. a) Sistemas multiagente inspirados económicamente. b) Agentes colaborativos. c) Equipos de agentes. d) Modelando agentes. e) Aprendizaje multiagente.

Objetivos. 1. Explicar en que difiere un agente de otras categorías de sistemas inteligentes. 2. Caracterizar y contrastar las arquitecturas estándar de agentes. 3. Describir la distinción entre agentes que aprenden y no aprenden. 4. Demostrar usando ejemplos apropiados como los sistemas multiagente soportan interacción de agentes.

4.7. Procesamiento de Lenguaje Natural.

Tópicos. I: Gramáticas determinísticas y estocásticas. II: Algoritmos de parsing. III: Métodos basados en corpus. IV: Recuperación de información. V: Traslación de lenguaje. VI: Reconocimiento del habla.

Objetivos. 1. Definir y contrastar gramáticas determinísticas y estocásticas, proveyendo ejemplos a para mostrar la adecuación de cada una. 2. Identificar algoritmos de parsing clásicos para parsing o a de lenguaje natural. 3. Defender la necesidad de un corpus establecido. 4. Dar ejemplos de catálogos y procedimientos de búsqueda en un método basado en corpus. 5. Articular la distinción entre técnicas para recuperación de información, traducción del lenguaje, y reconocimiento de voz.

4.8. Aprendizaje de Máquina y Redes Neuronales.

Tópicos. I: Definición y ejemplos de aprendizaje de máquina. II: Aprendizaje supervisado. III: Aro boles de aprendizaje por decisión. IV: Redes neuronales de aprendizaje. V: Redes de aprendizaje por creencia. VI: Algoritmo del vecino mas cercano. VII: Teoría de aprendizaje. VIII: El problema del sobreajuste. IX: Aprendizaje no supervisado. X: Aprendizaje por refuerzo.

Objetivos. 1. Explicar las diferencias entre tres principales estilos de aprendizaje: supervisado, no supervisado y por refuerzo. 2. Implementar algoritmos simples para aprendizaje supervisado, aprendizaje por refuerzo y aprendizaje no supervisado. 3. Determinar cuales de los tres estilos de aprendizaje es apropiado para un dominio de problema en particular. 4. Comparar y contrastar cada una de las siguientes técnicas, proveer ejemplos de cuando cada estrategia es superior: árboles de decisión, redes neuronales, y redes de creencia.. 5. Implementar de manera apropiada o un sistema de aprendizaje simple, usando arboles de decisión, redes neuronales y/o redes de creencia. 6. Caracterizar el estado del arte en teoría del aprendizaje, incluyendo logros y defectos. 7. Explicar el

algoritmo del vecino mas cercano y su lugar dentro de la teoría del aprendizaje. 8. Explicar el problema de sobreajuste, a través de técnicas para detectar y manejar el problema.

4.9. Robotica

Tópicos. I: Visión general. a) Estado del arte de sistemas de robot. b) Planeamiento vs. Control reactivo. c) Incerteza en control. d) Sentido. e) Modelos del mundo. II: Espacios de configuración. III: Planeamiento. IV: Sentido. V: Programación de robots. VI: Navegación y control.

Objetivos. 1. Sintetizar el potencial y limitaciones del estado del arte de los sistemas de robot de hoy. 2. Implementar los algoritmos de configuración de espacio para un robot 2D y polígonos complejos. 3. Implementar algoritmos de planeamiento de movimientos simples. 4. Explicar las incertezas asociadas con sensores y como tratar con estas incertezas. 5. Diseñar una arquitectura de control simple. 6. Describir varias estrategias para navegación en ambientes desconocidos, incluyendo las fortalezas y defectos de cada una. 7. Describir varias estrategias para navegación con la ayuda de hitos, incluyendo las fortalezas y defectos de cada una.

V. BIBLIOGRAFIA

- Goldberg, D. (1989). Genetic Algorithms in Search, Optimization and Machine learning. Addison Wesley .
- Haykin, S. (1999). Neural networks: A comprehensive Foundation. Prentice Hall.
- Nilsson, N. (2001). Inteligencia Artificial: Una nueva visión. McGraw-Hill.
- Russell, S. and Norvig, P. (2003). Inteligencia Artificial: Un enfoque moderno. Prentice Hall.
- Winston, P.H. y Horn, B.K. *LISP (3a. ed.)* (Addison--Wesley, 1991).
- Guy L. Steele Common Lisp: The Language, 2nd edición. Editorial: Burlington, MA : Digital Press, ©1984.
- Artificial Intelligence: A Modern Approach de Stuart Russell y Peter Norvig: Editorial: Upper Saddle River, N.J. : Prentice Hall/Pearson Education, ©2003.
- [Machine Learning](#) de T. Mitchell, (McGraw Hill, 1997).
- A Modern Approach, 2nd Edition, Russell and Norvig, Prentice Hall, 2003. ISBN 0-13-790395-2. (<http://aima.cs.berkeley.edu/>).
- Inteligencia Artificial. Una nueva síntesis. Nilsson, Nils J. Primera Edición. 2001. McGraw Hill Interamericana de España, S.A.U.
- Inteligencia artificial: modelos, técnicas y áreas de aplicación. Escolano, Francisco; Cazorla, Miguel Ángel; Alfonso, María Isabel. Primera Edición. 2003. Thomson Editores.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Compiladores
CODIGO	CC461
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC302, CC342
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

DOCENTE

FUNDAMENTACION DEL CURSO

Que el alumno conozca y comprenda los conceptos y principios fundamentales de la teoría de compilación para realizar la construcción de un compilador

SUMILLA

1. PL/Vision General de los Lenguajes de Programacion. 2. PL/Introduccion a la Traducccion de Lenguajes. 3. PL/Sistemas de Traducccion del Lenguaje. 4. Paralelismo a nivel de instruccion 5. Optimizacion para el paralelismo y la localidad

OBJETIVOS GENERALES

- Conocer las técnicas básicas empleadas durante el proceso de generación intermedio, optimización y generación de código.
- Aprender a implementar pequeños compiladores.

CONTRIBUCION A LOS RESULTADOS

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

a) Nivel Bloom 3

Aplicar conocimientos de computación y de matemáticas apropiadas para la disciplina.

b) Nivel Bloom 4

Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución.

j) Nivel Bloom 4

Aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida.

CONTENIDO ANALITICO

1 PL/Visión General de los Lenguajes de Programación. (8 horas) [Nivel Bloom 4]

Tópicos. 1. Historia de los lenguajes de programación. 2. Breve revisión de los paradigmas de programación. 3. Lenguajes procedurales. 4. Lenguajes orientados a objetos. 5. Lenguajes funcionales. 6. Lenguajes declarativos y no algorítmicos. 7. Lenguajes de *scripts*. 8. Los efectos de la escalabilidad en las metodologías de programación.

Objetivos. 1. Listar la evolución de los lenguajes de programación identificando como es que su historia nos ha conducido a los paradigmas actuales. 2. Identificar al menos una característica distintiva para cada uno de los paradigmas de programación cubiertos en esta unidad. 3. Evaluar las ventajas y desventajas entre los diferentes paradigmas, considerando temas tales como: eficiencia de espacio, eficiencia en el tiempo (para ambas partes computadora y programador), seguridad y el poder de las expresiones. 4. Distinguir entre la programación a menor y mayor escala.

Bibliografía: [Louden, 2004b, Pratt and V.Zelkowitz, 1998]

2 PL/Introducción a la Traducción de Lenguajes. (12 horas) [Nivel Bloom 3]

Tópicos. 1. Comparación entre intérpretes y compiladores. 2. Fases de traducción del lenguaje (análisis léxico, análisis sintáctico, generación de código, optimización). 3. Aspectos de traducción dependientes e independientes de la máquina.

Objetivos. 1. Comparar y contrastar modelos de ejecución interpretados y compilados, resaltando los méritos de cada uno. 2. Describir las fases de la traducción de programas desde el código fuente hasta llegar al código ejecutable y los archivos producidos por estas fases. 3. Explicar las diferencias entre la traducción dependiente e independiente de máquina y donde estas diferencias son evidentes en el proceso de traducción.

Bibliografía: [Aho et al., 2008, Aho, 1990, Teufel and Schmidt, 1998, Louden, 2004a, Appel, 2002]

3 PL/Sistemas de Traducción del Lenguaje. (24 horas) [Nivel Bloom 2]

Tópicos. 1. Aplicación de las expresiones regulares en analizadores léxicos. 2. Análisis sintáctico (sintaxis concreta y abstracta, árboles de sintaxis abstracta). 3. Aplicación de las gramáticas libres de contexto en un parseo dirigido por tablas o recursivo descendente. 4. Administración de tablas de símbolos. 5. Generación de código por seguimiento de un árbol. 6. Operaciones específicas de la arquitectura: selección de instrucciones y alocación de registros. 7. Técnicas de optimización. 8. El uso de herramientas como soporte en el proceso de traducción y las ventajas de éste. 9. Librerías de programas y compilación separada. 10. Construcción de herramientas dirigidas por la sintaxis.

Objetivos. 1. Describir los pasos y algoritmos usados por traductores lenguajes. 2. Reconocer los modelos formales subyacentes tales como los autómatas finitos, autómatas de pila y su conexión con la definición del lenguaje a través de expresiones regulares y gramáticas. 3. Discutir la efectividad de la optimización. 4. Explicar el impacto de la facilidad de la compilación separada y la existencia de librerías de programas en el proceso de compilación.

Bibliografía: [Aho et al., 2008, Aho, 1990, Teufel and Schmidt, 1998, Louden, 2004a, Appel, 2002, A. Lemone, 1996]

4 Paralelismo a nivel de instrucción (4 horas) [Nivel Bloom 2]

Tópicos. 1. Arquitectura de procesadores. 2. Restricciones de programación de código. 3. Programación de bloques básicos. 4. Programación de código global. 5. Canalización por software.

Objetivos. 1. Describir la importancia y poder de la extracción de paralelismo de las secuencias de instrucciones. 2. Explicar los conceptos de bloques básicos y código global. 3. Distinguir los conceptos entre canalización de instrucciones por software.

Bibliografía: [Aho et al., 2008]

5 Optimización para el paralelismo y la localidad (4 horas) [Nivel Bloom 2]

Tópicos. 1. Conceptos básicos. 2. Multiplicación de matrices. 3. Espacios de iteraciones. 4. Índices de arreglos afines. 5. Análisis de dependencias de datos de arreglos. 6. Búsqueda del paralelismo sin sincronización. 7. Sincronización entre ciclos paralelos.

Objetivos. 1. Diseñar, codificar programas para cálculos paralelos. 2. Identificar las propiedades básicas del paralelismo. 3. Aplicar los fundamentos del paralelismo en la programación.

Bibliografía: [Aho et al., 2008]

BIBLIOGRAFÍA

[Aho, 1990] Aho, A. (1990). *Compiladores Principios, tecnicas y herramientas*. Addison Wesley.

[Aho et al., 2008] Aho, A., Lam, M., Sethi, R., and Ullman, J. D. (2008). *Compiladores. Principios, tecnicas y herramientas*. Addison Wesley, 2nd edition. ISBN:10-970-26-1133-4.

[A.Lemone, 1996] A.Lemone, K. (1996). *Fundamentos de Compiladores*. CECSA-Mexico.

[Appel, 2002] Appel, A. W. (2002). *Modern compiler implementation in Java*. Cambridge University Press, 2.a edicion.

[Louden, 2004a] Louden, K. C. (2004a). *Construccion de Compiladores Principios y Practica*. Thomson.

[Louden, 2004b] Louden, K. C. (2004b). *Lenguajes de Programacion*. Thomson.

[Pratt and V.Zelkowitz, 1998] Pratt, T. W. and V.Zelkowitz, M. (1998). *Lenguajes de Programacion Diseno e Implementacion*. Prentice-Hall Hispanoamericana S.A.

[Teufel and Schmidt, 1998] Teufel, B. and Schmidt, S. (1998). *Fundamentos de Compiladores*. Addison Wesley Iberoamericana.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Biología Computacional
CODIGO	CC471
CREDITOS	04 (cuatro)
PRE-REQUISITO	CH061, CC302
CARÁCTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (02) PRACTICA (02) LABORATORIO (02)

El uso de métodos computacionales en las ciencias biológicas se ha convertido en una de las herramientas claves para el campo de la biología molecular, y estas actualmente son usadas como parte crítica en sus investigaciones. Existen diversas aplicaciones en biología molecular relativas tanto al ADN como al análisis de proteínas. La construcción del genoma humano, por ejemplo, depende fundamentalmente de la biología molecular computacional. Muchos de los problemas de esta área son realmente complejos y con conjuntos enormes de datos

OBJETIVO DEL CURSO

Interpretar problemas biológicos haciendo uso de técnicas computacionales.
Analizar e implementar algoritmos y estructuras aplicables al campo de la biología.

SUMILLA

1. Conceptos Introdutorios 2. Alineamiento de Secuencias 3. Clustering 4. Arboles Filogenéticos 5. Mapeo de Secuencias 6. Introducción a la Estructura de las Proteínas

CONTRIBUCION A LA FORMACION PROFESIONAL Y FORMACION GENERAL

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- a) Aplicar conocimientos de Computación y de matemática apropiadas para la disciplina.
- g) Analizar el impacto local y global de la Computación sobre los individuos, organizaciones y sociedad.
- h) Incorporarse a un proceso de aprendizaje profesional continuo.
- i) Utilizar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- j) Aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida

CONTENIDO ANALITICO

UNIDAD 1: Conceptos Introdutorios

OBJETIVO:

Identificación de los conceptos básicos en Biología Molecular

Reconocimiento de problemas clásicos en Biología Molecular y su representación en el campo computacional

Aprendizaje de las herramientas de software e Internet clásicas para el campo de Bioinformática

Introducción a los conceptos necesarios en manejo de Cadenas, Grafos y su representación algorítmica a fin de transformar problemas biológicos al tipo computacional

CONTENIDO:

Introducción a la Historia de la Genética

Conceptos Básicos de Biología Molecular

Problemas clásicos en Bioinformática

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Herramientas de recolección y almacenamiento de secuencias en laboratorio
Recursos de Software, introducción a BLAST, CLUSTALW
Cadenas, Grafos y Algoritmos

UNIDAD 2: Alineamiento de Secuencias

OBJETIVO:

Reconocimiento de las técnicas básicas usadas en el alineamiento de secuencias
Implementación de los diversos algoritmos de comparación de secuencias
Introducción a la programación dinámica
Introducción y comparativa entre métodos heurísticos y exactos
Métodos probabilísticos: PAM

CONTENIDO:

Introducción al alineamiento de secuencias
Comparación de pares de secuencias
Alineamiento de Secuencias Global
Alineamiento de Secuencias Múltiples
Cadenas ocultas de Markov
Métodos exactos, aproximados y heurísticos del alineamiento de secuencias
Problemas derivados del alineamiento de secuencias

UNIDAD 3: Clustering

OBJETIVO:

Identificar métodos de distancia aplicables a grafos del tipo arboles
Conocer la transformación de Matrices en estructuras de grafos
Reconocer a los métodos de Clustering como útiles para la identificación de funciones en genes no conocidos a partir de genes similares
Identificar la importancia del Clustering en el reconocimiento de patrones de enfermedades

CONTENIDO:

El problema del Clustering
Clustering Jerárquico
Algoritmo de Neighbour Joining
Algoritmo del Average linkage
Clustering no jerárquico o K-means
EST clustering

UNIDAD 4: Arboles Filogenéticos

OBJETIVO:

Reconocer algoritmos de mediciones de distancias
Analizar la complejidad computacional de cada uno de los algoritmos estudiados
Reconocer la importancia de la filogenia en casos de evolución de epidemias como el HIV
Utilización de herramientas de software de libre uso Implementación de los algoritmos estudiados

CONTENIDO:

Introducción a la Filogenia
Algoritmos comunes
Aplicaciones biológicas
Algoritmos Exactos
Algoritmos Probabilísticos

UNIDAD 5: Mapeo de Secuencias

OBJETIVO:

Identificación de problemas NP-Complejos
Aplicación e implementación de técnicas diversas a fin de dar solución a estos problemas biológicos
Introducción a los métodos de tipo goloso
Reconocimiento de tópicos avanzados en teoría de Grafos

CONTENIDO:

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Problema del Double Digest y Partial Digest

Técnicas utilizadas en el mapeo de secuencias

Mapeo con Non-Unique Probes

Mapeo con Unique Probes

Grafos de Intervalos

Mapeo con Señales de Frecuencias de Restricción

UNIDAD 6: Introducción a la Estructura de las Proteínas

OBJETIVO:

Examina algunos tópicos de reconocimiento visual en Computación Gráfica

Implementación de algunas estructuras simples como el folding 2D

CONTENIDO:

Fundamentos biológicos de las proteínas

Motivación para la predicción de las estructuras de las proteínas

Alineamiento rígido de Proteínas

Técnica del alineamiento por Hashing Geométrico

Predicción de la estructuras de las proteínas

REFERENCIAS

Clote, P. and Backofen, R. (2000). Computational Molecular Biology, An Introduction. Wiley.

Jones, N. A. and Pevzner, P. A. (2004). An Introduction to Bioinformatics Algorithms. The MIT Press.

Mount, D. W. (2001). Bioinformatics: Sequence and Genome Analysis. Cold Spring Harbor Laboratory Press.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Administración de Redes
CODIGO	CC481
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC362
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Presenta la administración de redes en Linux para completar el conocimiento del uso de Linux desde el punto de vista del usuario.
- Presenta detalles de cómo funcionan y configuran routers y switches dentro de una red WAN.

SUMILLA

Este curso cubre conceptos básicos de redes, particularmente las basadas en TCP/IP. Configuraciones de nivel bajo (NAT) y nivel alto (rlogin, NFS, NIS, ...etc). Seguidamente se cuenta cómo configurar una red WAN compuesta de routers, switches, y computadores.

CONTENIDO ANALITICO

CC481 Administración de Redes	
<ul style="list-style-type: none">▪ Introducción al Trabajo en Redes▪ Cuestiones sobre redes TCP/IP▪ Configuración del hardware de red▪ Configuración del hardware serie▪ Configuración del Protocolo TCP/IP▪ El servicio de nombres y su configuración▪ SLIP: IP por línea serie▪ El protocolo punto a punto▪ Cortafuegos de TCP/IP▪ Contabilidad IP▪ Enmascaramiento IP y Traducción de Direcciones de Red▪ Características Importantes de Redes	<ul style="list-style-type: none">▪ El Sistema de Información de Red (NIS)▪ El Sistema de Ficheros de Red▪ IPX y el Sistema de Ficheros NCP▪ Administración de Taylor UUCP▪ Correo electrónico▪ Sendmail▪ Noticias▪ NNTP y el Demonio nntpd▪ Noticias de Internet▪ Configuración del lector de noticias▪ Configuraciones de cableado útiles▪ WAN, Routers y Switches

BIBLIOGRAFIA

- <http://www.ibiblio.org/pub/linux/docs/LuCaS/Manuales-LuCAS/GARL2/garl2/index.html>
- TCP/IP network administration, por Craig Hunt, Edition: 3, Publicado por O'Reilly, 2002
- Linux network administrator's guide, por Tony Bautts, Terry Dawson, Gregor N. Purdy. Edition: 3, Publicado por O'Reilly, 2005

SÍLABO

I. DESCRIPCION DEL CURSO

CURSO	Sistemas Concurrentes y Distribuidos
CODIGO	CC462
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC401
CARACTER	obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

Los Sistemas Distribuidos, son aquellos cuyos componentes hardware y software, que están en ordenadores conectados en red, se comunican y coordinan sus acciones mediante el paso de mensajes, para el logro de un objetivo. Se establece la comunicación mediante un protocolo prefijado por un esquema cliente-servidor

El propósito de un Sistema Distribuido es proveer un ambiente en que el usuario puede ejecutar sus aplicaciones distribuida de forma transparente.

Ademas el curso contempla actividades practicas en donde se resolverán problemas de concurrencia y se modificara el funcionamiento de un pseudo Sistema Distribuido.

II. OBJETIVO DEL CURSO

Conocer los elementos básicos del diseno de Sistemas Distribuidos
Aprender a instalar y usar aplicaciones en Sistemas Distribuidos

III. SUMILLA

Procesamiento distribuido. Sistema de administración de bases de datos distribuido. Arquitectura de los sistemas distribuidos. Arquitectura Cliente/Servidor. Descomposición Funcional. Lógica de Presentación. Lógica de Datos. Lógica de Negocios. Modelos de componentes distribuidos.

IV. CONTENIDO ANALITICO

1. Introducción a los Sistemas Distribuidos

- 1.1 Objetivos
- 1.2 Conceptos de Hardware y Software
- 1.3 Características
- 1.4 Aspectos de Diseño

2. Comunicación de los Sistemas Distribuidos

- 2.1 Stacks de Comunicaciones (Protocolos con capas)
- 2.2 El modelo Cliente/Servidor

3. El modelo Cliente/Servidor

- 3.1 Arquitectura
- 3.2 Clientes y Servidores
- 3.3 Plataformas
- 3.4 Modelos 2-Tier, 3-Tier, Multi-Tier

4. Soporte del Sistema Operativo

4.1 Procesos e Hilos

4.2 Modelos

5. Middleware

5.1 Llamada a un procedimiento remoto (RPC)

5.2 Middleware Orientado a Mensajes (MOM)

5.3 Peer-to-Peer

5.4 Servicio de directorio

5.5 Seguridad

6. Sistemas Distribuidos de Archivos

6.1 Diseño

6.2 Implementación

6.3 Tendencias

7. Transacciones Distribuidas y Control de Concurrencia

7.1 Sincronización

7.2 Exclusión Mutua

7.3 Transacciones Atómicas

7.4 Bloqueos en Sistemas Distribuidos

8. Objetos Distribuidos - Modelos de Componentes

8.1 Objetos y Componentes

8.2 Beneficios

8.3 Modelos de Componentes

9. CORBA

9.1 Arquitectura

9.2 Metadata y Servicios

9.3 ORB e IDL

9.4 CORBA IIOP

9.5 Implementaciones

10. COM

10.1 Historia

10.2 Arquitectura

10.3 Servicios

10.4 Documentos Compuestos y OCX/ActiveX

10.5 Integración COM-CORBA

10.6 Implementaciones

11. Enterprise Java Beans (EJB)

11.1 Arquitectura

11.2 Servicios

11.3 Componentes

11.4 EJB y RMI

11.5 Integración EJB-CORBA

11.6 Integración EJB-COM

11.7 Implementaciones

12. Web Services

12.1 Arquitectura

12.2 Servicios

12.3 XML, UDDI, SOAP

12.4 Implementaciones .NET y J2EE

13. Comparación entre Modelos de Componentes

13.1 Comparación CORBA, COM, EJB y Web Services

13.2 Integración de las arquitecturas con Web Services (Microsoft.NET, Sun One, IBM)

Bibliografía

1. Sistemas Operativos Distribuidos Andrew S. Tanenbaum - 1996 - Prentice-Hall
2. Client/Server Survival Guide Robert Orfali - Dan Harkey - 1994 - Van Nostrand Reinhold
3. Distributed Systems: Concepts and Design George Coulouris, Jean Dollimore and Tim Kindberg - Addison-Wesley
4. Lan Times - Guía de Interoperabilidad Tom Sheldon - 1995 - Osborne/McGraw-Hill
5. Lan Times - Enciclopedia de Redes – Networking Tom Sheldon - 1994 - McGraw-Hill
6. Redes de Ordenadores Andrew S. Tanenbaum - 1991 - Prentice-Hall
7. Diccionario técnico en línea: www.webopedia.com
8. Corba 3 developers guide - Reaz Hoque - 1998 - IDG Books
9. Corba Masterminds Object Management - Warren Keuffel - revista DBMS online marzo 1997 - <http://www.dbmsmag.com/9703d13.html>
10. The Next Generation of Corba - Michael Guttman and Rob Appelbaum - revista Object development & engineering - agosto 1998
11. OMA overview - OMG - www.omg.org
12. <http://www.microsoft.com>
13. Microsoft Component Services. Server Operating System. A Technology Overview. Microsoft Corporation, August 1998.
14. The Component Object Model: Technical Overview. This paper is adapted from an article in Dr. Dobbs Journal, December 1994. Copy Right 1996, Dr. Dobbs Journal and Microsoft Corporation. All rights reserved.
15. The Component Object Model Specification. Draft Version 0.9, October 24, 1995 Microsoft Corporation and Digital Equipment Corporation
16. Microsoft Windows Architecture for Developers. Student Workbook. Course Number: 794A. Part Number: 4635A. Release: 11/10/97
17. <http://www.cool.sterling.com>
18. Component Based Development. A Management Overview. Changing the information system delivery process.
19. Component Bases Development and Object Modeling. Keith Short. Sterling Software. Febrery 1997. Ver. 1.05
20. Enterprise Java Beans http://www.java.sun.com/products/ejb/white_paper.html
<http://www.java.sun.com:8081/beans/faq.html>
<http://www.viamall.com/softpro/languages-java-javabeans.html>
<http://www.javasoft.com/beans/docs/index.html>
<http://www.splash.javasoft.com/beans/spec.html>
<http://www.smoke.sdsmt.edu/docs/jdk1.1.1-doc/guide/beans/index.html>
<http://www.valto.com/>
<http://www.boci.com/>
<http://www.ibm.com/java/education/intro-javabeans/sitemap/index.html>

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Ingeniería de software I
CODIGO	CC472
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC302
CARACTER	obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

FUNDAMENTACION

La tarea de desarrollar software, excepto para aplicaciones sumamente simples, exige la ejecución de un proceso de desarrollo bien definido. Los profesionales de esta área requieren un alto grado de conocimiento de los diferentes modelos de proceso de desarrollo, para que sean capaces de elegir el más idóneo para cada proyecto de desarrollo. Por otro lado, el desarrollo de sistemas de mediana y gran o escala requiere del uso de bibliotecas de patrones y componentes y del dominio de técnicas relacionadas e al diseño basado en componentes.

OBJETIVO DEL CURSO

- Familiarizar al alumno con los procesos de software que se presentan en el desarrollo del ciclo de vida del software.
- Presentar a los alumnos los diferentes modelos de evaluación de procesos y las métricas del proceso de software
- Los alumnos deben identificar los requerimientos Funcionales y no Funcionales de la construcción o de un software
- Orientar a los alumnos en el modelamiento de análisis de requerimientos y prototipos de un a software.
- Los alumnos debe ser capaces de seleccionar y aplicar patrones de diseño apropiados en la construcción de una aplicación de software
- Aplicar el diseño de componentes y el diseño de reuso en las aplicaciones presentadas por los alumnos.

CONTRIBUCION A LOS RESULTADOS

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- Habilidad para analizar un problema e identificar y definir los requerimientos computacionales apropiados para su solución.
- Habilidad para diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas.
- Habilidad para trabajar efectivamente en equipos para cumplir con un objetivo común.
- Habilidad para comunicarse efectivamente con audiencias diversas.
- Habilidad para usar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- Habilidad para aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que o n demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida.

- Habilidad para aplicar los principios de desarrollo y diseño en la construcción de sistemas de n o software de complejidad variable.

CONTENIDO ANALITICO

1 Procesos de Software.

Tópicos. I: Ciclo de vida del software y modelos de procesos. II: Modelos de evaluación de procesos. III: Métricas del proceso de software.

Objetivos. 1. Explicar el ciclo de vida del software y sus fases incluyendo las entregas que son producidas. 2. Seleccionar con justificación los modelos de desarrollo de software más apropiados para el desarrollo y mantenimiento de un diverso rango de productos de software. 3. Explicar el rol del proceso de modelos de madurez. 4. Comparar el modelo tradicional cascada con el modelo incremental, el modelo orientado a objetos, y otros modelos apropiados. 5. Para cada uno de los diferentes escenarios de proyectos de software, describir la posición del proyecto en el ciclo de vida del software, identificar las tareas particulares que deben ser desarrolladas seguidamente e identificar las métricas apropiadas para estas tareas.

2 Representación Avanzada del Conocimiento y Razonamiento.

Tópicos. I: Elicitación de requerimientos. II: Técnicas de modelamiento del análisis de requerimientos. III: Requerimientos Funcionales y no Funcionales. IV: Prototipo. V: Conceptos básicos de técnicas de especificación formal.

Objetivos. 1. Aplicar elementos clave y métodos comunes para la elicitación y análisis, para producir un conjunto de requerimientos de software para un sistema de tamaño medio. 2. Discutir los retos de mantener software heredado. 3. Usar un método común, no formal para modelar y e u especificar (en la forma de un documento de especificación de requerimientos) los requerimientos para un sistema de software de tamaño medio. 4. Conducir una visión general de un documento de requerimientos de software, usando las mejores prácticas para determinar la calidad del documento. 5. Traducir en lenguaje natural una especificación de requerimientos de software escrita o en un lenguaje de especificación formal comúnmente usado.

3 Tópicos Fundamentales en Sistemas Inteligentes.

Tópicos. I: Conceptos fundamentales de diseño y principios. II: Patrones de diseño. III: Arquitectura de software. IV: Diseño estructural. V: Análisis y diseño orientado a objetos. VI: Diseño a nivel componente. VII: Diseño para reuso.

Objetivos. 1. Discutir las propiedades del buen diseño de software. 2. Evaluar la calidad de múltiples diseños de software basados en principios de diseño, claves y conceptos. 3. Seleccionar y aplicar patrones de diseño apropiados en la construcción de una aplicación de software. 4. Conducir una síntesis de diseño de software usando la guía apropiada. 5. Comparar y contrastar el análisis orientado a objetos con el análisis y diseño estructural. 6. Crear y especificar el diseño de software para un producto de software de tamaño medio usando una especificación de requerimientos de software, una metodología de diseño de programas aceptado (ejemplo orientado a objetos o estructurado), y una notación de diseño apropiada. 7. Evaluar un diseño de software a nivel componente.. 8. Evaluar un diseño de software a nivel componente desde la perspectiva de reuso.

BIBLIOGRAFIA

- Blum, B. I. (1992). Software Engineering: A Holistic View. Oxford University Press US, 7th edition.
- Pressman, R. S. (2004). Software Engineering: A Practitioner's Approach. McGrawHill, 6th edition.
- Schach, S. R. (2004). Object-Oriented and Classical Software Engineering. McGrawHill.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Núcleo y Redes para Computación Paralela
CODIGO	CC482
CREDITOS	02 (dos)
PRE-REQUISITO	CC461, CC481
carácter	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (01) LABORATORIO (03)

OBJETIVO DEL CURSO

Construir un sistema operativo y las redes de comunicaciones que permitan la utilización de múltiples núcleos.

SUMILLA

El curso desarrolla el sistema operativo partiendo de la infraestructura de BSD para poder exportar los elementos mas básicos como memoria física con seguridad básica que puede ser incrementada por las capas superiores.

CONTENIDO ANALITICO

CC482 Núcleo y Redes para Computación Paralela	
<ul style="list-style-type: none">▪ Arquitectura, diseño e implementación del ExoKernel.▪ Arquitectura, diseño e implementación de librerías que pueden formar parte del sistema operativo (libases).▪ Definición del Control y la Seguridad en las librerías de sistema operativo (libOS)▪ Multiplexación de redes▪ Protección de discos▪ Protocolos: Arquitectura, diseño, e implementación ▪ Diseño e implementación del protocolo HTTP	<ul style="list-style-type: none">▪ Diseño e implementación de la arquitectura del Web Server▪ Sistema de archivos de Redes▪ Arquitectura, diseño, e implementación del Router▪ Arquitectura, diseño, e implementación del Switch▪ Sistema de archivos con cambios garantizados (Journal file system)▪ Sistema Global de archivos (Global File system)▪ Motor de búsqueda ▪ Base de Datos

BIBLIOGRAFIA

- <http://pdos.csail.mit.edu/exo.html>
- <http://en.wikipedia.org/wiki/Exokernel>
- <http://es.wikipedia.org/wiki/Exon%C3%BAcleo>
- <http://pdos.csail.mit.edu/exo/distrib.html>
- <http://www.cs.berkeley.edu/~brewer/cs262b-2004/Lec-Exokernel.pdf>
- <http://www.cs.utexas.edu/users/dahlin/Courses/UGOS/reading/engler95exokernel.pdf>

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- <http://pdos.csail.mit.edu/PDOS-papers.html>
- P. J. Hatcher y Michael J. Quinn, Data-Parallel Programming on MIMD Computers, MIT Press, 1991
- C. Xavier y S.S. Ivengar, Introducción to parallel algoritmos, Wiley-Interscience, 1998.
- J. Reinders, Intel Threading Building Blocks: Outfitting C++ for Multi-core Processor Parallelism, O'Reilly, 2007
- Shameem Akhter y Jason Roberts, Multi-Core Programming Increasing Performance through Software Multi-threading, Intel Corporation, 2006.
- Andrew S. Tanenbaum y Maarten van Oteem, Distributed Systems: Principles and paradigms, Segunda edición, Prentice Hall, 2006.
- John L. Hennessy y David A. Patterson, Computer Architecture: A Quantitative Approach. Morgan Kaufmann, cuarta edición, 2006.
- Maurice J. Bach ,Design of the UNIX Operating System. Prentice Hall PTR, 1986
- Kaare Christian y Susan Richter, The UNIX Operating System. Wiley Professional Computing, 1993.
- Stephen W. Keckler, Kunle Olukotun y H. Peter Hofstee, Multicore Processors and Systems (Integrated Circuits and Systems). Springer .2009

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Introducción a la Computación Grid y Cloud
CODIGO	CC581
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC462
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Introducción al lenguaje Java que nos permite el acceso al ambiente de Globus.
- Aprender a usar la programación paralela en la Infraestructura de Globus que es la implementación de la infraestructura Grid.
- Introducción al Cloud Computing desde una perspectiva tecnologica y de negocios.

SUMILLA

El curso permite usar los servicios de Globus. La tecnología Grid muestra el nuevo paradigma de Computación distribuida propuesto por Ian Foster y Carl Kesselman a mediados de los 90. Se basa fundamentalmente en el acceso remoto a recursos computacionales, y su tecnología estándar es el Globus Toolkit.

En la segunda parte del curso se presentará una visión general de Cloud Computing. En concreto, tratara sobre las plataformas, las abstracciones, el nexa entre una aplicación eficiente nube y la virtualización. Plataformas Cloud populares serán discutidas, y los factores a considerar mientras que la aplicación de una iniciativa Cloud también se presentará.

CONTENIDO ANALITICO

Computación de malla (Grid computing)

- Introducción a Java
- Configuración del GRID
- Como lograr acceso al GRID (Organizaciones Virtuales)
- Obtener los permisos de seguridad (Certificados de usuarios y certificados de maquina)
- Herramientas de Globus
- Como usar los servicios de Globus
- Como escribir un Servicio con los siguientes métodos: Add x, Subtract y, getValue método. (The WS-Resource Framework)
- Crear la estructura del folder
- Implementar la Interface de Grid Service
- Implementación del Grid Service
- Configuración del Grid Service
- Distribución del Grid Service

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- Verificación del Grid service
- Recuperación del Grid service
- Servicio del Data Grid Services
- Definición de un elemento de Data Service (SDE)
- Notificaciones en Grid Service
- Grid: ejecución, terminación y monitoreo
 1. Gestión de la asignación de recursos de Globus (GRAM)
 2. Especificación de los recursos de lenguaje (RSL)
 3. Acceso global al almacenamiento secundario de GLOBUS (GASS) y transferencia de archivos (GridFT)
- Interface de mapeo de Java CogKi a Globus Toolkit
- Java Paralelo
- Tecnología del sistema Condor. High Throughput Computing (HTC)

Computación en nube (Cloud Computing)

Introducción a la Computación en nube

- Definiendo Computación en nube
 - Componentes de una Computación en nube
 - Tipos de nube: publica, privada, híbrida
- Entregando servicios de la nube
 - Tipos de servicio por categorías
 - Vendedores de productos de la nube: Amazon, Google, Microsoft y otros

Adoptando la nube

- Los factores clave de soluciones de Cloud Computing
 - Aprovisionamiento instantáneo de recursos informáticos
 - Manipulación de cargas variadas con elasticidad y escalabilidad sin problemas
 - El aprovechamiento de una capacidad de almacenamiento infinita
 - Modelos de facturación por costo efectivo (pay-as-you-use)
- Barreras a la Computación en nube
 - Manejando datos sensibles
 - Aspectos de seguridad de la nube
 - Evaluación de soluciones de gobierno

Software como un servicio (SaaS)

- Caracterizando SaaS
 - Minimizando la necesidad por hardware y software local
 - Simplificación de la administración con instalación centralizada y actualizaciones
 - Optimizando costos y rendimiento con la habilidad de escalar de acuerdo a la demanda
- Escenarios de servicio
 - Mejorando la colaboración con herramientas de productividad de negocios
 - Simplificando la creación de procesos de negocios por integración de componentes existentes
- Tecnologías SaaS
 - Implementación de aplicaciones Web
 - Implementación de servicios Web: SOAP, REST
 - Escogiendo una plataforma de desarrollo

Plataforma como un Servicio (PaaS)

- Fundamentos técnicos para el PaaS
 - Especificando las componentes del PaaS
 - Analizando disposiciones del vendedor PaaS
 - Seleccionando una implementación apropiada

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- Construyendo servicios con pilas de solución
 - Evaluando la arquitectura de la plataforma específica del vendedor
 - Herramientas de la plataforma de servicio
 - Utilizando el poder del middleware escalable
- Administrando almacenamiento en la nube
 - Controlando datos no estructurados en la nube
 - Implementando bases de datos relacionales en la nube
 - Mejorando la disponibilidad de los datos
- Empleando servicios de soporte
 - Prueba en la nube
 - Monitoreando servicios basados en la nube
 - Analizando portabilidad a través de plataformas

La Infraestructura como un Servicio (IaaS)

- Permitiendo tecnologías
 - Clústeres escalables de servidores
 - Logro de transparencia con la virtualización de la plataforma
 - Dispositivos elásticos de almacenamiento
- Accediendo al IaaS
 - Suministro de servidores de acuerdo a la demanda
 - Manejando direcciones IP dinámicas y estáticas
 - Herramientas y soporte para administración y monitoreamiento

Construyendo un Caso de Negocio

- Implicaciones financieras
 - Requerimientos de cómputo actuales y futuros
 - Comparando facilidades locales con las de la nube
 - Estimación de factores económicos downstream
- Preservando la continuidad del negocio
 - Selección apropiada de acuerdos de nivel de servicio
 - Protección del acceso a los activos de la nube
 - Seguridad, disponibilidad y estrategias de recuperación en caso de desastre, en la nube

Migrando a la Nube

- Consideraciones técnicas
 - Rearquitectando aplicaciones para la nube
 - Integrando la nube con aplicaciones existentes
 - Evitando la dependencia del proveedor
- Planeando la migración
 - Solución incremental vs un paso
 - Seleccionando el vendedor
 - Estableciendo las necesidades de habilidades del personal

BIBLIOGRAFIA

- Foster, I. and Kesselman, C. (eds) (1999) The Grid: Blueprint for a New Computing Infrastructure. San Francisco, CA: Morgan Kaufmann.
- <http://www.globus.org>
- <http://www.eu-datagrid.org>
- Grid computing: making the global infrastructure a reality, por Fran Berman, Geoffrey Fox, Anthony J. G. Hey, Fran Berman, Geoffrey Fox. Publicado por John Wiley and Sons, 2003.
- Grid computing: software environments and tools, por José Cardoso Cunha, Omer Rana. Publicado por Birkhäuser, 2006.
- Grid computing: experiment management, tool integration, and scientific workflows, por Radu Prodan, Thomas Fahringer. Publicado por Springer, 2007.
- Lessons in Grid Computing : The System Is a Mirror ; Por Stuart Robins. Publicado por John Wiley & Sons
- Grid Computing for Developers , Por Vladimir Silva . Publicado Charles River Media
- The Evolution of Grid Computing ; Por Juergen Hirtenfelder. Publicado VDM Verlag Dr. Mueller e.K.
- Introduction to Cloud Computing: Business and Technology. Por Timothy Chou. Publicado por

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Sistemas Operativos Avanzados
CODIGO	CC571
CREDITOS	02 (dos)
PRE-REQUISITO	CC482, CC401
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (01), LABORATORIO (03)

OBJETIVO DEL CURSO

- Demuestra las componentes de un sistema operativo y como trabajar en el, en forma real.
- Enseñar la implementación del Kernel de Linux. Se recompilará el Kernel y se instalará como parte del proceso de manejo de software abierto.

SUMILLA

Se estudia las componentes del sistema operativo y se usa el software del Kernel de Linux para estudiar como este es construido. Las componentes más importantes del kernel son estudiados en detalle para buscar la solución a un problema reportado.

CONTENIDO ANALITICO

CC571 Sistemas Operativos Avanzados	
<ul style="list-style-type: none">▪ Introducción▪ Estructura del kernel○ Núcleo○ Inicialización○ Arquitectura y control de Memoria○ Comunicación y control entre los Procesos○ Redes○ Sistema de Archivos○ Código que depende del tipo de plataforma▪ Estructura para la programación del código Kernel ▪ Estructura del sistema de archivos	<ul style="list-style-type: none">▪ Archivos principales▪ Mount – Montar sistema de archivos y los programas de manejo del dispositivo▪ Clases de dispositivos▪ Espacios asignados al kernel y al Usuario▪ Dispositivo para el manejo de caracteres▪ Ejemplo de un programa para manejo de un dispositivo▪ Medición de tiempo en el kernel (jiffies) y en el espacio del usuario▪ Gestión de la memoria▪ Puertos de entrada y salida▪ Dispositivos por bloques▪ Depuración ▪ Portabilidad

BIBLIOGRAFIA

- Understanding the Linux Kernel By Daniel P. Bovet & Marco Cesati Editorial: Beijing ; Cambridge, Mass. : O'Reilly, 2001.
- Kernel Projects for Linux By Gary Nutt Editorial: Boston : Addison Wesley Longman, 2001.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- Linux Device Drivers, 2nd Edition by Alessandro Rubini & Jonathan Corbet Editorial: Sebastopol : O'Reilly & Associates, ©2005.
- Linux File Systems By Moshe Bar Editorial: New York : Osborne/McGraw-Hill, ©2001.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

I. DESCRIPCION DEL CURSO

CURSO	Seminario de Tesis I
CODIGO	CC541
CREDITOS	04 (cuatro)
PRE-REQUISITO	Aprobar 8 ciclos
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (02)

II. OBJETIVO DEL CURSO

- Supervisar las etapas iniciales del desarrollo de tesis de cada uno de los alumnos registrados en el seminario; incluyendo las etapas de búsqueda bibliográfica, definición de los objetivos, metodología, productos a entregar y calendario del trabajo de investigación propuesto.
- Registro formal del tema de tesis (Protocolo) de acuerdo a los lineamientos propuestos por la Facultad de Ciencias en el programa de la Ciencia de la Computación.

III. SUMILLA

Tiene por finalidad orientar al alumno en la preparación de la tesis profesional. Se sustenta en el conocimiento previo, el cual se organiza y actualiza en torno a un plan de investigación que además de ser riguroso debe ser realizable.

IV. CONTENIDO ANALITICO

CC541 Seminario de Tesis I	
<ul style="list-style-type: none">▪ Qué es una investigación▪ Tesis<ol style="list-style-type: none">a. Introducciónb. Qué es lo que se sabe hasta ahora brevementec. Presentar las ideas mas relevantes que han sido trabajadas por otrosd. Problema que se va resolvere. DESCRIPCION de cómo resolvió el problemaf. Conclusióng. Referenciash. Apéndices▪ Cómo escoger un profesor como guía de tesis	<ul style="list-style-type: none">▪ Cómo se hace investigación en MIT

V. BIBLIOGRAFIA

- <http://delta.cs.cinvestav.mx/~francisco/semtesis/semtesis07.html>
- <http://www.cs.umd.edu/~oleary/gradstudy/gradstudy.html>
- <http://www.cs.indiana.edu/mit.research.how.to.html>

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Seminario de Tesis II
CODIGO	CC542
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC541
CARACTER	obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (02)

OBJETIVO DEL CURSO

- Supervisar la parte final de la tesis. En el primer mes del semestre el alumno sustentará los resultados antes de terminar de escribir la tesis.
- Los resultados del trabajo se publicará en eventos dentro y fuera del país.

SUMILLA

Tiene por finalidad orientar al alumno para que escriba los resultados en su tesis como para presentar los resultados en congresos y revistas indexadas.

CONTENIDO ANALITICO

CC542 Seminario de Tesis II	
<ul style="list-style-type: none">▪ Qué es una investigación▪ Cómo escribir una tesis▪ Cómo escribir un trabajo para ser presentado en el congreso▪ Cómo escribir un trabajo para ser presentado en una revista indexada▪ Cómo postular para conseguir un trabajo▪ Cómo postular para seguir estudios de postgrado dentro y fuera del país▪ Cómo continuar haciendo investigación▪ Cómo conseguir referencias para continuar un postgrado▪ Cómo conseguir referencias para obtener un trabajo	<ul style="list-style-type: none">▪ Cómo hacer trabajos de investigación en grupo▪ Cómo hacer investigación con personas fuera de la Universidad

BIBLIOGRAFIA

- <http://delta.cs.cinvestav.mx/~francisco/semtesis/semtesis07.html>
- <http://www.cs.umd.edu/~oleary/gradstudy/gradstudy.html>
- http://www.cs.indiana.edu/mit_research.how_to.html

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Simulación y Modelamiento
CODIGO	CC562
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC201, CC441, CM334
CARÁCTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

1. Explicar los beneficios de la simulación y el modelado en una serie de importantes áreas de aplicación.
2. Demostrar la capacidad de aplicar las técnicas de modelado y simulación a una serie de áreas problemáticas.
3. Evaluar una simulación, destacando las ventajas y los inconvenientes.

SUMILLA

Temas:

- Definición de la simulación y el modelado, la relación entre la simulación y modelado
 - Objetivo incluidos los beneficios y limitaciones: el papel - la consideración de rendimiento, optimización, el apoyo a la toma de decisiones, la previsión, las consideraciones de seguridad, para la formación y la educación
 - Importantes áreas de aplicación: la asistencia sanitaria (incluida la asistencia con el diagnóstico), economía y finanzas; aula del futuro, la formación y la educación; juegos; ciudad y simulaciones urbanas; simulación en la ciencia y la ingeniería de simulación militar
 - Diferentes tipos de simulaciones - físico, humano en el lazo, la interacción, ordenador, realidad virtual
 - El proceso de simulación - base sólida, la identificación de las características clave o comportamientos, la simplificación de las hipótesis, validación de los resultados. La construcción de modelos: el uso de la fórmula matemática o ecuación, gráficos, restricciones. metodológicas y técnicas. Uso del tiempo que camina de sistemas dinámicos
 - Consideraciones teóricas; Métodos Monte Carlo, procesos estocásticos, teoría de colas
 - Tecnologías en apoyo de la simulación y modelado: los procesadores gráficos; dispositivos Haptic feedback.
- Consideraciones de interacción humano-computador.
- La evaluación y la evaluación de simulaciones en una variedad de contextos.
 - Software en apoyo de la simulación y el modelado, los paquetes, los idiomas

CONTENIDO ANALITICO

UNIDAD 1:

Sistema de definición y componentes, actividades estocástico, continua y sistemas discretos, el modelado del sistema, los tipos de modelos, modelos físicos estáticos y dinámicos, modelos matemáticos estáticos y dinámicos, el modelo social completa, los tipos de sistemas de estudio.

UNIDAD 2:

Simulación del sistema, Necesidad de la simulación, Naturaleza básica de la simulación, las técnicas de simulación, comparación de la simulación y métodos analíticos, los tipos de sistema de simulación, simulación en

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

tiempo real, simulación híbrida, la simulación de problemas búsqueda, el sistema de un solo servidor de cola y un problema de inventario, simulación Monte-Carlo, modelo distribuido de Lag, modelo Telaraña.

UNIDAD 3:

Simulación de sistemas continuos, simulación analógica vs digital, simulación de sistema de reservorio de agua, la simulación de un sistema servo, la simulación de un piloto automático. Simulación de un sistema discreto, modelo de tiempo de paso fijo vs evento-a-evento, generación de números aleatorios, la prueba de aleatoriedad, Computación Monte-Carlo vs simulación estocástica.

UNIDAD 4:

Sistemas dinámicos, modelos de crecimiento exponencial, modelos de decaimiento exponencial, curvas logísticas, diagramas de sistemas dinámicos, modelo mundial.

UNIDAD 5:

Simulación de redes PERT, el cálculo de la ruta crítica, incertidumbres en la duración de la actividad, asignación de recursos y consideración, lenguajes de simulación, simulación orientada a objetos.

Reference Books:

- 1) Geoffrey Gordon, “System Simulation”, PHI
- 2) Narsingh Deo, “System Simulation with digital computer”, PHI.
- 3) Averill M. Law, W. David Kelton, “Simulation Modelling and Analysis”,TMH.

SÍLABO

I. DESCRIPCION DEL CURSO

CURSO	Estructura e Interpretación de Programas
CODIGO	CC011
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC201, CC342
CARACTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

II. OBJETIVO DEL CURSO

- Introducir conceptos de programación sin tener en cuenta el lenguaje y las reglas de syntax de este.
- Demostrar que cualquier lenguaje debe definir data y procedure en la forma más simple (primitivos). Además debe tener métodos para manipular data y procedures en forma abstracta.

III. SUMILLA

El curso muestra que un proceso computacional como entidades que residen en el computador y que los datos son entes abstractos usados por el proceso, durante la evolución de este. La evolución de un proceso es dirigido por un conjunto de reglas llamado programa. Se escribe programas para dirigir procesos.

IV. CONTENIDO ANALITICO

CC011 Estructura e Interpretación de Programas	
<ul style="list-style-type: none">▪ Creación de abstracciones con Procedures<ol style="list-style-type: none">a. Elementos de Programaciónb. Los procesos que los Procedures generanc. Formulación de abstracciones con Procedures de alto orden▪ Creación de abstracciones con datos<ol style="list-style-type: none">a. Introducción a la abstracción de Datosb. Jerarquía de los Datos y propiedad de Clausurac. Tipos de Datosd. Diferentes representaciones de las abstracciones de los datose. Sistemas con operaciones genéricas▪ Objetos, modularidad, y estado<ol style="list-style-type: none">a. Asignamiento y estado localb. Modelo de evaluaciónc. Modelos con datos mutablesd. Concurrenciae. Streams	<ul style="list-style-type: none">▪ Abstracción de Metalingüística<ol style="list-style-type: none">a. El evaluador meta curricularb. Variaciones sobre un esquema – Evaluación Perezosac. Variaciones sobre un esquema – Computación no indeterminada▪ Programación lógica<ol style="list-style-type: none">a. Computación con máquinas con registrosb. Diseño con máquinas que tienen registrosc. Simulador de una máquina con registrosd. Alocación de almacenamiento y garbage collectione. Control evaluador explícitof. Compilación▪ Lenguajes<ol style="list-style-type: none">a. Schemab. Otros lenguajes vistos

V. BIBLIOGRAFIA

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- <http://mitpress.mit.edu/sicp/full-text/book/book.html>
- Structure and Interpretation of Computer Programs 2nd edition_Harold Abelson and Gerald Jay Sussman with Julie Sussman_1996 McGraw-Hill Book Company

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Herramientas para el uso de software abierto
CODIGO	CC012
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC102, CC361
CARACTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Demuestra la cultura y metodología de software abierto. Nuevos desarrolladores de software aprenden a trabajar a través de las herramientas del software abierto.
- Se demuestra que las herramientas son necesarias para el éxito de cualquier proyecto. al igual que la automatización es un elemento fundamental en el proceso de desarrollo de software abierto de alta calidad.

SUMILLA

El curso cubre todas las herramientas necesarias para trabajar con el software abierto. Describe también su cultura; para que el alumno esté al tanto de cómo debe trabajar con otros desarrolladores de software que nunca ha conocido y posiblemente no tendrá la oportunidad de conocerlos.

CONTENIDO ANALITICO

CC012 Herramientas para el uso del software abierto	
<ul style="list-style-type: none">▪ Cómo escribir buenos programas▪ GNU Emacs▪ Automake y Autoconf▪ Libtool▪ Autotoolset▪ Cómo usar C y C++ en una forma más eficiente▪ Internacionalización del código▪ Mantenimiento de la Documentación▪ Programación portable de la shell▪ Escritura de macros con Autoconf▪ RCS y CVS▪ Bourne Shell, AWK, Sed y herramientas similares de secuencias de comandos▪ Herramientas para depurar el lenguaje C: gdb	<ul style="list-style-type: none">▪ Bugzilla▪ Buenas prácticas para mejoras (arreglar el sistema), nombres de proyectos y archivos, licencias y derecho de autor, desarrollo, distribución, comunicación y gestión de proyectos▪ Diagnóstico: Verificación del error de código▪ Mejora: Solución del error, pruebas y comunicación a la comunidad de que existe para el problema xxx (error de código identificado con xxx)▪ Distribución de la mejora.▪ Instalación de la mejora▪ Cultura de software abierto

BIBLIOGRAFIA

- <http://www.ibiblio.org/pub/linux/docs/LuCaS/Manuales-LuCAS/GARL2/garl2/index.html>
- http://en.wikipedia.org/wiki/Open_source_software_development
- <http://www.tldp.org/HOWTO/Software-Release-Practice-HOWTO/>
- <http://www.catb.org/~esr/faqs/hacker-howto.html>
- <http://samizdat.mines.edu/howto/HowToBeAProgrammer.html>
- GNU/Linux Application Programming, por M. Tim Jones, ebrary, Inc. Publicado por Cengage Learning, 2005. The definitive guide to GCC, por William Von Hagen. Publicado por Apress, 2006.
- Free Software, Free Society: Selected Essays of Richard M. Stallman, por Richard M. Stallman, Lawrence Lessig, Joshua Gay, Free Software Foundation (Cambridge, Mass.), Colaborador Lawrence Lessig, Joshua Gay. Publicado por Free Software Foundation, 2002.
- Open source: technology and policy, por Fadi P. Deek, James A. McHugh. Publicado por Cambridge University Press, 2007.
- The cathedral and the bazaar: musings on Linux and Open Source by an accidental revolutionary, por Eric S. Raymond. Publicado por O'Reilly, 2001.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Tópicos Especiales de Software
CODIGO	CC021
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC302, CC322, CC331
CARÁCTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Demuestra una metodología para desarrollar redes de voz y datos
- Muestra los beneficios y deficiencias de la implementación de voz sobre IP

SUMILLA

El curso nos muestra la telefonía actual. Se desarrolla la parte conceptual de los paquetes de voz en tiempo real sobre las redes de datos. Cubre el descubrimiento, análisis, planificación y diseño de un proyecto de integración de voz y datos.

CONTENIDO ANALITICO

CC021 Tópicos Especiales de Software	
<ul style="list-style-type: none">▪ Estado de las comunicaciones por voz▪ La señalización utilizada en telefonía empresarial▪ Sistema de señalización▪ Enrutamiento de llamadas y planificación de conexiones telefónicas▪ Definición y medida de la calidad de la voz▪ Digitalización y codificación de la voz▪ Paquetes de voz y redes de datos▪ Protocolos WAN para servicios de voz y datos integrados▪ Consideraciones de diseño para los protocolos WAN▪ Revisión de las funciones IP para la integración de voz y datos	<ul style="list-style-type: none">▪ Protocolos de señalización y transporte VOIP▪ Integración de voz y datos: metodología, planificación y diseño▪ Presupuestos de retraso y planes de pérdida▪ Establecimiento de un plan integrado de servicio telefónico▪ Habilidad de la calidad de servicio de red extensa▪ Establecimiento de la conectividad del router PBX▪ Establecimiento de capacidad de llamada de área extensa▪ Resolución de cuestiones relacionadas con la calidad de la voz

BIBLIOGRAFIA

- UNIX Network Programming: Networking APIs: Sockets and XTI; Vol I por W Richard Stevens Editorial: Upper Saddle River, NJ : Prentice Hall PTR, 1998.
- Integración de Redes de Voz Datos por Scott keagy Editorial: Madrid : Cisco Systems, ©2001.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Introducción a Lenguajes Formales
CODIGO	CC023
CREDITOS	02 (dos)
PRE-REQUISITO	CC262, CM094
CHARACTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (01), LABORATORIO (02), SEMINARIO (01)

OBJETIVO DEL CURSO

- Desarrollar su capacidad de abstracción, utilizar un lenguaje de programación Funcional.
- Introducir y desarrollar estructuras y algoritmos.

SUMILLA

Este es uno de los primeros cursos del tronco de conocimientos de la carrera de Ciencia de la Computación. Los lenguajes formales son usados para definición precisa de los formatos de datos y de la sintaxis de los lenguajes de programación. Se muestra su importancia en el desarrollo de compiladores. La Teoría de Autómatas vera las maquinas abstractas y los problemas que son capaces de resolver.

CONTENIDO ANALITICO

- 1 Autómata: Los métodos y la locura
 - 1.1 Porque estudiamos Teoría de Autómatas?
 - 1.1.1 Introducción al Autómata Finito
 - 1.1.2 Representaciones Estructurales
 - 1.1.3 Autómata y Complejidad
 - 1.2 Introducción a la Prueba Formal
 - 1.2.1 Pruebas Deductivas
 - 1.2.2 Reducción a Definiciones
 - 1.2.3 Otros Teoremas Formales
 - 1.2.4 Teoremas que parecen no ser instrucciones If-then
 - 1.3 Formas adicionales de prueba
 - 1.3.1 Probando equivalencias sobre conjuntos
 - 1.3.2 La contrapositiva
 - 1.3.3 Prueba por contradicción
 - 1.3.4 Contra ejemplos
 - 1.4 Pruebas inductivas
 - 1.4.1 Inducciones en enteros
 - 1.4.2 Formas mas generales de inducciones en enteros
 - 1.4.3 Inducciones estructurales
 - 1.4.4 Inducciones mutuas
 - 1.5 Los conceptos centrales de la Teoría de Autómatas
 - 1.5.1 Alfabetos

- 1.5.2 Cadenas
- 1.5.3 Lenguajes
- 1.5.4 Problemas
- 2 Autómata finito
 - 2.1 Una representación informal de Autómata Finito
 - 2.1.1 Las reglas de base
 - 2.1.2 El protocolo
 - 2.1.3 Permitiendo al autómata ignorar acciones
 - 2.1.4 El sistema entero como un Automaton
 - 2.1.5 Usando el producto Automaton para validar el Protocolo
 - 2.2 Autómata Finito Determinístico (DFA)
 - 2.2.1 Definición de un Autómata Finito Determinístico
 - 2.2.2 Como un DFA procesa las cadenas
 - 2.2.3 Notaciones simples para DFA's
 - 2.2.4 Extendiendo la función de transición a cadenas
 - 2.2.5 El Lenguaje de un DFA
 - 2.3 Autómata Finito Nondeterminístico (NFA)
 - 2.3.1 Una vista informal del NFA
 - 2.3.2 Definición del NFA
 - 2.3.3 La función de transición extendida
 - 2.3.4 El lenguaje de un NFA
 - 2.3.5 Equivalencia de DFA y NFA
 - 2.3.6 Un caso malo para la construcción de subconjuntos
 - 2.4 Una aplicación: Búsqueda de Texto
 - 2.4.1 Hallando cadenas en Texto
 - 2.4.2 NFA para búsqueda de Texto
 - 2.4.3 Un DFA para reconocer un conjunto de palabras llave
 - 2.5 Autómata Finito con Transiciones Epsilon
 - 2.5.1 Uso de Transiciones-epsilon
 - 2.5.2 La notación formal para un NFA-epsilon
 - 2.5.3 Cierres-Epsilon
 - 2.5.4 Transiciones extendidas y Lenguajes para NFA-epsilon
 - 2.5.5 Eliminando Transiciones-epsilon
- 3 Expresiones Regulares y Lenguajes
 - 3.1 Expresiones Regulares
 - 3.1.1 Los operadores de Expresiones Regulares
 - 3.1.2 Construyendo Expresiones Regulares
 - 3.1.3 Precedencia de operadores de Expresiones Regulares
 - 3.2 Autómata Finito y Expresiones Regulares
 - 3.2.1 De los DFA's a Expresiones Regulares
 - 3.2.2 Convirtiendo DFA's a Expresiones Regulares por Eliminación de Estados
 - 3.2.3 Convirtiendo Expresiones Regulares a Autómata
 - 3.3 Aplicaciones de Expresiones Regulares
 - 3.3.1 Expresiones Regulares en UNIX
 - 3.3.2 Análisis del Léxico
 - 3.3.3 Hallando patrones en Texto
 - 3.4 Leyes algebraicas para Expresiones Regulares
 - 3.4.1 Asociatividad y Conmutatividad
 - 3.4.2 Identidades y Aniquiladores
 - 3.4.3 Leyes Distributivas
 - 3.4.4 La ley de Idempotencia
 - 3.4.5 Leyes envolviendo cierres
 - 3.4.6 Descubriendo leyes para Expresiones Regulares
 - 3.4.7 El test para una ley algebraica para Expresiones Regulares
- 4 Propiedades de los Lenguajes Regulares
 - 4.1 Probando si un lenguaje no es Regular

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- 4.1.1 El lema de bombeo para lenguajes regulares
- 4.1.2 Aplicaciones del lema de bombeo
- 4.2 Propiedades de cierre de Lenguajes Regulares
 - 4.2.1 Cierre de Lenguajes Regulares bajo Operaciones Booleanas
 - 4.2.2 Reversa
 - 4.2.3 Homomorfismos
 - 4.2.4 Homomorfismos inversos
- 4.3 Propiedades de decisión de lenguajes Regulares
 - 4.3.1 Convirtiendo entre Representaciones
 - 4.3.2 Probando el vacío de Lenguajes Regulares
 - 4.3.3 Probando sociedad en un Lenguaje Regular
- 4.4 Equivalencia y Minimización de Autómata
 - 4.4.1 Probando Equivalencia of States
 - 4.4.2 Probando Equivalencia of Lenguajes Regulares
 - 4.4.3 Minimización de DFA's
 - 4.4.4 Porque el DFA Minimizado no puede ser vencido
- 5 Gramáticas libre de contexto y Lenguajes
 - 5.1 Gramáticas libre de contexto (CFG)
 - 5.1.1 Un ejemplo informal
 - 5.1.2 Definición de Gramáticas libre de contexto
 - 5.1.3 Derivaciones usando una Gramática
 - 5.1.4 Derivaciones lo mas izquierdo y lo mas derecho
 - 5.1.5 El lenguaje de una Gramática
 - 5.1.6 Formas de sentencia
 - 5.2 Arboles ordenados o de análisis (**Concrete Syntax Tree o Parse Tree**)
 - 5.2.1 Construyendo arboles de análisis
 - 5.2.2 El rendimiento de un árbol de análisis
 - 5.2.3 Inferencia, Derivaciones, y Arboles de Análisis
 - 5.2.4 De Inferencias a Arboles
 - 5.2.5 De arboles a Derivaciones
 - 5.2.6 De Derivaciones a Inferencias Recursivas
 - 5.3 Aplicaciones de Gramáticas libre de contexto
 - 5.3.1 Analizadores
 - 5.3.2 El Generador-Analizador YACC
 - 5.3.3 Lenguajes de Marca
 - 5.3.4 XML y definiciones de tipo de documento
 - 5.4 Ambigüedad en Gramáticas y Lenguajes
 - 5.4.1 Gramáticas ambiguas
 - 5.4.2 Removiendo ambigüedad de las Gramáticas
 - 5.4.3 Derivaciones de lo mas izquierdo como una forma de expresar ambigüedad
 - 5.4.4 Ambigüedad Inherente
- 6 Autómata de empuje hacia abajo (Pushdown Automata - PDA)
 - 6.1 Definición del PDA
 - 6.1.1 Introducción informal
 - 6.1.2 La definición formal de PDA
 - 6.1.3 Una notación gráfica para PDA's
 - 6.1.4 Descripciones instantáneas de un PDA
 - 6.2 Los lenguajes de un PDA
 - 6.2.1 Aceptancia para Estado Final
 - 6.2.2 Aceptancia para pila vacía
 - 6.2.3 De pila vacía a estado final
 - 6.2.4 De estado final a pila vacía
 - 6.3 Equivalencia de PDA's y CFG's
 - 6.3.1 De CFGs a PDAs
 - 6.3.2 De PDA's a CFGs
 - 6.4 Autómata Pushdown Determinístico

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- 6.4.1 Definición de un PDA Determinístico
- 6.4.2 Lenguajes Regulares y PDAs Determinísticos
- 6.4.3 DPDA's y Lenguajes Libres de Contexto (CFL)
- 6.4.4 DPDA's y Gramáticas Ambiguas

7 Propiedades de CFLs

- 7.1 Formas normales para CFGs
 - 7.1.1 Eliminando símbolos inútiles
 - 7.1.2 Calculando la generación y símbolos accesibles
 - 7.1.3 Eliminando Producciones-epsilon
 - 7.1.4 Eliminando producciones unitarias
 - 7.1.5 Forma normal de Chomsky
- 7.2 El lema de bombeo para CFLs
 - 7.2.1 El tamaño de árboles de análisis
 - 7.2.2 Declaración del lema de bombeo
 - 7.2.3 Aplicaciones del lema de bombeo para CFL's
- 7.3 Propiedades de cierre de los CFLs
 - 7.3.1 Substituciones
 - 7.3.2 Aplicaciones del Teorema de Substitución
 - 7.3.3 Reversa
 - 7.3.4 Intersección con un Lenguaje Regular
 - 7.3.5 Homomorfismo inverso
- 7.4 Propiedades de decisión de los CFL's
 - 7.4.1 Complejidad de conversión entre CFG's y PDA's
 - 7.4.2 Tiempo de duración de la conversión a la forma normal de Chomsky
 - 7.4.3 Probando Vacío de CFL's
 - 7.4.4 Probando pertenencia en un CFL
 - 7.4.5 Revisión de los problemas indecidibles CFL

8 Introducción a las Maquinas de Turing

- 8.1 Problemas que los computadores no pueden resolver
 - 8.1.1 Programas que imprimen "Hello, World"
 - 8.1.2 El hipotético probador "Hello, World"
 - 8.1.3 Reduciendo un problema a otro
- 8.2 La maquina de Turing (TM)
 - 8.2.1 La búsqueda para decidir todas las preguntas matemáticas
 - 8.2.2 Notación para la Maquina de Turing
 - 8.2.3 Descripciones instantáneas para Maquinas de Turing
 - 8.2.4 Diagramas de transition para Maquinas de Turing
 - 8.2.5 El Lenguaje de una Maquina de Turing
 - 8.2.6 Maquinas de Turing y la Parada
- 8.3 Técnicas de Programación para Maquinas de Turing
 - 8.3.1 Almacenamiento en el estado
 - 8.3.2 Múltiples Temas
 - 8.3.3 Subrutinas
- 8.4 Extensiones a la maquina de Turing básica
 - 8.4.1 Maquinas de Turing multicinta
 - 8.4.2 Equivalencia TM's multicinta y de una cinta
 - 8.4.3 Tiempo de duración y construcción de multicinta a una cinta
 - 8.4.4 maquinas de Turing no-deterministas
- 8.5 Maquinas de Turing restringidas
 - 8.5.1 TMs con cintas semi-infinitas
 - 8.5.2 Maquinas multi-pila
 - 8.5.3 Maquinas contadoras
 - 8.5.4 El poder de las maquinas contadoras
- 8.6 Maquinas de Turing y computadores
 - 8.6.1 Simulando una maquina de Turing por un Computador
 - 8.6.2 Simulando un Computador por una Maquina de Turing

8.6.3 Comparando los tiempos de duración de Computadores y Maquinas de Turing

9 Indecidibilidad

- 9.1 Un lenguaje que no es recursivamente enumerable (RE)
 - 9.1.1 Enumerando las cadenas binarias
 - 9.1.2 Códigos para Maquinas de Turing
 - 9.1.3 El Lenguaje de Diagonalización (L_d)
 - 9.1.4 Prueba de que el L_d no es recursivamente enumerable
- 9.2 Un problema indecidible que es RE
 - 9.2.1 Lenguajes Recursivos
 - 9.2.2 Complementos de lenguajes Recursivos y RE
 - 9.2.3 El Lenguaje Universal
 - 9.2.4 Indecidibilidad del Lenguaje Universal
- 9.3 Problemas de Indecidibilidad sobre Maquinas de Turing
 - 9.3.1 Reducciones
 - 9.3.2 Maquinas de Turing que aceptan Lenguaje Vacío
 - 9.3.3 Teorema de Rice y Propiedades de los lenguajes RE
 - 9.3.4 Problemas acerca especificaciones de la Maquina de Turing
- 9.4 el Problema de la correspondencia del cartero (PCP)
 - 9.4.1 Definición del PCP
 - 9.4.2 El PCP “modificado”
 - 9.4.3 Finalización de la prueba de indecidibilidad del PCP
- 9.5 Otros problemas indecidibles
 - 9.5.1 Problemas acerca Programas
 - 9.5.2 Indecidibilidad de la ambigüedad para CFG's
 - 9.5.3 El complemento de una lista de lenguajes

10 Problemas intratables

- 10.1 Las clases P y NP
 - 10.1.1 Problemas Solubles en tiempo polinomial (P)
 - 10.1.2 Un ejemplo: Algoritmo de Kruskal
 - 10.1.3 Tiempo polinomial Nondeterministico (NP)
 - 10.1.4 Un ejemplo NP: El problema del vendedor viajero (TSP)
 - 10.1.5 Reducciones de tiempo polinomial
 - 10.1.6 Problemas NP-Completo
- 10.2 Un problema NP-Completo
 - 10.2.1 El problema de Satisfiabilidad (SAT)
 - 10.2.2 Representando instancias SAT
 - 10.2.3 NP-Compleitud del problema SAT
- 10.3 Un problema de Satisfiability Restringida
 - 10.3.1 Formas Normales (NF) para Expresiones Booleanas
 - 10.3.2 Convirtiendo Expresiones a CNF
 - 10.3.3 NP-Compleitud de CSAT
 - 10.3.4 NP-Compleitud de 3SAT
- 10.4 Problemas adicionales NP-Completo
 - 10.4.1 Describiendo problemas NP-completos
 - 10.4.2 El Problema de conjuntos Independientes
 - 10.4.3 El problema de Node-Cover
 - 10.4.4 El problema directo del Circuito-Hamilton
 - 10.4.5 Circuitos Hamilton no dirigidos y el TSP

11 Clases adicionales de Problemas

- 11.1 Complementos de Lenguajes en NP
 - 11.1.1 La Clase de Lenguajes Co-NP
 - 11.1.2 Problemas NP-Completo y Co-NP
- 11.2 Problemas Solubles en espacio polinomial
 - 11.2.1 Maquinas de Turing de Espacio-Polinomial
 - 11.2.2 Relación de PS y NPS a Clases definidas previamente
 - 11.2.3 Espacio polinomial Deterministico y Nondeterministico

- 11.3 Un problema que es Completo para PS
 - 11.3.1 PS-Complejidad
 - 11.3.2 Formulas Booleanas cuantificadas (QBF)
 - 11.3.3 Evaluando QBFs
 - 11.3.4 PS-Complejidad del problema QBF
- 11.4 Clases de Lenguaje basados en Randomización
 - 11.4.1 Quicksort: un ejemplo de algoritmo Randomizado
 - 11.4.2 Un modelo de Maquina de Turing usando Randomización
 - 11.4.3 El lenguaje de una Maquina de Turing Randomizada
 - 11.4.4 La clase RP (tiempo polinomial randomizado)
 - 11.4.5 Reconociendo Lenguajes en RP
 - 11.4.6 The Class ZPP (tiempo polinomial probabilístico de error cero)
 - 11.4.7 Relaciones entre RP y ZPP
 - 11.4.8 Relaciones a las Clases P y NP
- 11.5 La complejidad del Test de Primalidad
 - 11.5.1 La importancia del Test de Primalidad
 - 11.5.2 Introducción a la Aritmética Modular
 - 11.5.3 La complejidad de Aritmética-Modular
 - 11.5.4 Test de Primalidad Random-Polinomial
 - 11.5.5 Test de Primalidad Nondeterministico

BIBLIOGRAFÍA

- Teoría de autómatas y lenguajes formales. Dean Kelley, 1998
- Autómatas y Lenguajes. Ramón F. Breña. Editor Tecnológico de Monterrey. ISBN9709448404, 9789709448405
- <http://infolab.stanford.edu/~ullman>

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Interacción Humano Computador
CODIGO	CC032
CREDITOS	03 (tres)
PRE-REQUISITO	CC472 Ingeniería de Software I
CARACTER	Obligatorio
MODALIDAD	Semestral
HORAS	TEORIA (01) PRACTICA (02) LABORATORIO (02)

DOCENTE

FUNDAMENTACION DEL CURSO

La eficacia del Interfaz Usuario-Computador se ha hecho cada vez más importante debido a que los sistemas de computadora se han hecho instrumentos útiles para personas no entrenadas en ciencias de la computación. De hecho, la interfaz es a menudo el factor más importante en el éxito o el fracaso de cualquier sistema computacional.

Las transacciones, con las cuestiones numerosas interrelacionadas de manera sutil y consideraciones técnicas, comportamentales y estéticas, consumen una parte grande y creciente del tiempo de desarrollo y un porcentaje correspondiente del código total para cualquier aplicación dada. Una revisión de uno de los libros más acertados en la Interacción Usuario-Computador, esta información da a los estudiantes, investigadores y practicantes una descripción de los conceptos significativos y una guía completa de la literatura de investigación de esta área.

OBJETIVOS GENERALES

- Cómo desarrollar técnicas para compartir datos y conocimiento entre los distintos usuarios de una aplicación y cómo apoyar los procesos y dinámicas de interacción grupal que permiten lograr un objetivo común.
- Desarrollar en el alumno la tendencia de que la tecnología se adapte a las personas y no las personas a la tecnología, independiente de las características que tengan, ya sean físicas, ideológicas u otras.

CONTRIBUCION A LOS RESULTADOS

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

b) Nivel Bloom 3

Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución.

c) Nivel Bloom 3

Diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas.

d) Nivel Bloom 3

Trabajar efectivamente en equipos para cumplir con un objetivo común.

e) Nivel Bloom 4

Entender correctamente las implicancias profesionales, éticas, legales, de seguridad y sociales de la profesión.

g) Nivel Bloom 3

Analizar el impacto local y global de la computación sobre los individuos, organizaciones y sociedad.

o) Nivel Bloom 4

Mejorar las condiciones de la sociedad poniendo la tecnología al servicio del ser humano.

CONTENIDO ANALITICO

1 PF/Programación Orientada a Eventos. (1 horas) [Nivel Bloom 3]

Tópicos. 1. Métodos para la manipulación de eventos. 2. Propagación de eventos. 3. Manejo de excepciones.

Objetivos. 1. Explicar la diferencia entre programación orientada a eventos y programación por línea de comandos. 2. Diseñar, codificar, probar y depurar programas de manejo de eventos simples que respondan a eventos del usuario. 3. Desarrollar código que responda a las condiciones de excepción lanzadas durante la ejecución.

Bibliografía: [Baecker et al., 2000]

2 HC/Fundamentos de la Interacción Hombre-Computador (HCI) (4 horas) [Nivel Bloom 3]

Tópicos. 1. Motivación: ¿Por qué el estudio de la interacción entre las personas y la tecnología es vital para el desarrollo de sistemas más usables y aceptables? 2. Contexto para HCI (dispositivos móviles, dispositivos consumidores, aplicaciones de negocios, web, sistemas colaborativos, juegos, etc.). 3. Proceso de desarrollo centrado en el usuario: foco temprano en los usuarios, pruebas empíricas, diseño iterativo. 4. Distintas medidas de evaluación: utilidad, eficiencia, facilidad de aprendizaje, satisfacción del usuario. 5. Modelos para medir el diseño de la Interacción Humano-Computador: atención, percepción y reconocimiento, movimiento y cognitivo. 6. Aspectos sociales que influyen en el diseño y en el uso de la Interacción Humano-Computador: cultura, comunicación y organizaciones. 7. Adaptación a la diversidad humana, incluyendo diseño y accesibilidad universal, diseño para múltiples contextos culturales y lingüísticos. 8. Los errores más frecuentes del diseño de interfaces. 9. Interfaces de usuario estándares.

Objetivos. 1. Discutir las razones por las cuales es importante el desarrollo de software centrado en el usuario. 2. Explicar porque los modelos humanos individuales y los modelos sociales son importantes a la hora de diseñar la Interacción Humano-Computador. 3. Definir un proceso centrado en el usuario que explícitamente haga notar que el desarrollador y su conocimiento previo son muy distintos del usuario. 4. Describir formas en las cuales un diseño centrado en el usuario puede fallar incluyendo ejemplos. 5. Definir los diferentes procesos para definición de interfaces para diferentes contextos. 6. Diferencias entre el rol de la hipótesis y los resultados experimentales vs. correlaciones. 7. Escoger entre métodos de evaluación cualitativos y cuantitativos para una evaluación dada. 8. Usar vocabulario para analizar la interacción humano con el software: costo percibida, modelo conceptual, modelo mental, metáforas, diseño de la interacción, retroalimentación, etc. 9. Proveer ejemplos de como un determinado ícono, símbolo, palabra o color puede tener diferentes interpretaciones entre a) dos diferentes culturas humanas y b) una cultura y una de sus subculturas. 10. Estar preparado para describir al menos un estándar nacional o internacional de diseño estándar de interfaces.

Bibliografía: [Baecker et al., 2000]

3 HC/Construcción de Interfaces Gráfica de Usuario. (6 horas) [Nivel Bloom 4]

Tópicos. 1. Principios de las interfaces gráficas de usuario (GUIs). 2. Acción-objeto vs objeto-acción. 3. Eventos de la interfase de usuario. 4. Construcción de una interfase de usuario para correr localmente y una para la web.

Objetivos. 1. Identificar los varios principios fundamentales para el diseño de interfaces de usuario tales como facilidad de aprendizaje, flexibilidad y robustez. 2. Describir ejemplos de mala navegación, malos diseños de pantalla e diseños de interfaces incomprensibles. 3. Crear una aplicación simple que soporte interfase gráfica de usuario para correr localmente o en la web. 4. Observar un intento de usar una aplicación por parte de un usuario y obtener su crítica a la aplicación. 5. Explicar como una cuidadosa evaluación va más allá de la observación de un único usuario.

Bibliografía: [Baecker et al., 2000]

4 HC/Evaluación de Software Centrado en el usuario. (4 horas) [Nivel Bloom 4]

Tópicos. 1. Evaluación sin usuarios típicos: recorridos, *Keystroke Level Model* (KLM), análisis basado en expertos, heurísticas, lineamientos y estándares. 2. Evaluación con usuarios típicos: observación, pensar en voz alta, entrevista, examen, experimentos. 3. Desafíos de una evaluación efectiva: muestreo, evaluación. 4. Reporte de resultados de evaluación.

Objetivos. 1. Discutir el criterio de evaluación: tiempo de las tareas y completitud, tiempo de aprendizaje, retención, errores y satisfacción del usuario. 2. Conducir un ensayo y un análisis del modelo de nivel Keystroke (KLM). 3. Comparar una interfase de un usuario dado con un conjunto de lineamientos o estándares para identificar insuficiencias. 4. Conducir una prueba de usabilidad con más de un usuario, recolectar resultados con al menos dos métodos. 5. Comparar una prueba de laboratorio con una prueba de campo. 6. Explicar un problema de usabilidad soportado por resultados de una prueba de usabilidad. Recomendar una solución al problema de usabilidad. 7. Criticar una evaluación de usuario, resaltar las amenazas de validación. 8. Dado un contexto de evaluación (por ejemplo: tiempo, disponibilidad de usuarios de prueba, lugar en el proceso de diseño, objetivos de evaluación), recomendar y justificar un método de evaluación.

Bibliografía: [Baecker et al., 2000]

5 HC/Desarrollo de Software Centrado en el Humano. (5 horas) [Nivel Bloom 4]

Tópicos. 1. Enfoques, características y listado de procesos. 2. Funcionalidad y usabilidad: análisis de tareas, entrevistas, encuestas. 3. Especificación de interacción y presentación. 4. Técnicas de prototipado y herramientas. a) Secuencia de pantallas en papel. b) Herencia y despacho dinámico. c) Lenguajes de prototipado y constructores de GUI.

Objetivos. 1. Explicar los tipos básicos y características del desarrollo orientado al usuario. 2. Comparar el desarrollo centrado al humano con los métodos tradicionales de ingeniería del software. 3. Establecer tres requerimientos funcionales y tres requerimientos de usabilidad. 4. Especificar un objeto interactivo con redes de transición, diseño OO o descripción de escenarios. 5. Discutir las ventajas y desventajas del desarrollo con prototipos de software y en papel.

Bibliografía: [Baecker et al., 2000]

6 HC/Diseño de la Interfaz Gráfica de Usuario. (4 horas) [Nivel Bloom 3]

Tópicos. 1. Selección de estilos de interacción (línea de comandos, menú, voz, gestos, WIMP (window, icon, menu, pointing device)) y técnicas de interacción. 2. Escoger el widgets adecuado para usuarios y tareas. 3. Aspectos de tipo HCI en el diseño de pantallas: esquemas, color, fuentes y etiquetado. 4. Manejo de fallas humanas y de sistema. 5. Más allá del diseño de ventanas simples: visualización, representación y metáforas. 6. Interacción multimodal: gráficos, sonidos y haptics. 7. Interacción 3D y realidad virtual. 8. Diseño para dispositivos pequeños como celulares. 9. Interacción y comunicación multi cultural.

Objetivos. 1. Listar los estilos comunes de interacción. 2. Explicar los principios del buen diseño de cada uno de los siguientes tópicos: widgets comunes, presentación de pantallas secuenciadas, diálogos de captura de errores, manuales de usuario. 3. Diseñar, prototipar y evaluar una GUI 2D simple ilustrando el conocimiento de los conceptos enseñados en las unidades: HC/Evaluación de Software Centrado en el usuario. y HC/Desarrollo de Software Centrado en el Humano. 4. Discutir los retos que existen en la interacción del movimiento de interacciones 2D a 3D. 5. Identificar los objetivos que existen en llevar una aplicación de una pantalla tradicional a un dispositivo móvil.

Bibliografía: [Baecker et al., 2000]

7 HC/Programación de Interfaces Gráficas de Usuario. (4 horas) [Nivel Bloom 3]

Tópicos. 1. UIMS, independencia de diálogos y análisis de niveles, el modelo Seeheim. 2. Clases Widgets y librerías. 3. Administración de eventos e interacción de usuario. 4. Diseño web vs. diseño de aplicaciones nativas. 5. Administración de geometría. 6. Constructores de GUI y entornos de programación UI. 7. Diseño para múltiples plataformas. 8. Diseño para dispositivos móviles.

Objetivos. 1. Diferenciar entre las responsabilidades de la UIMS y la aplicación. 2. Diferenciar entre los modelos basados en kernel y cliente-servidor para la UI. 3. Comparar los paradigmas orientados a eventos con los procedimientos tradicionales para la UI. 4. Describir la agregación de widgets y la administración de la geometría basada en restricciones. 5. Explicar los callbacks y su rol en los constructores de GUI. 6. Identificar al menos tres diferencias comunes en el diseño de una interfase de usuario multi-plataforma (por ejemplo, para escritorio, web y teléfono celular). 7. Identificar las características comunes que se puedan encontrar en interfase de usuario multi-plataforma.

Bibliografía: [Baecker et al., 2000]

8 HC/Aspectos de Sistemas de Multimedia. (4 horas) [Nivel Bloom 3]

Tópicos. 1. Categorización y arquitecturas de información: jerarquías, mallas (*grids*), hipermedia, redes. 2. Recuperación de información y desempeño humano. 3. Búsqueda Web. 4. Usabilidad de los lenguajes de consultas a base de datos. 5. Gráficos. 6. Sonido. 7. Diseño HCI de sistemas de información con multimedia. 8. Reconocimiento de voz y procesamiento de lenguaje natural. 9. Medios de información y computación móvil. 10. Visualizaciones interactivas. 11. Diseño y navegación en información. 12. Interfases táctiles.

Objetivos. 1. Discutir en que se diferencia la recuperación de información del procesamiento de transacciones. 2. Explicar como la organización de la información soporta la recuperación. 3. Describir los principales problemas de usabilidad de los lenguajes de consultas de bases de datos. 4. Explicar en particular el estado actual de la tecnología de reconocimiento de voz y en general el estado del procesamiento de lenguaje natural. 5. Diseñar, prototipar y evaluar un sistema de información multimedia simple ilustrando el conocimiento de los conceptos mostrados en las unidades HC/Desarrollo de Software Centrado en el Humano., HC/Diseño de la Interfaz Gráfica de Usuario. y HC/Aspectos de Sistemas de Multimedia.

Bibliografía: [Baecker et al., 2000]

9 HC/Aspectos de Colaboración y Comunicación. (4 horas) [Nivel Bloom 3]

Tópicos. 1. Groupware para soporte de tareas especializadas: preparación de documentos, juegos para multi-jugadores. 2. Comunicación asincrónica en grupo: e-mail, boletines, listserv, wikis, etc. 3. Comunicación de grupo sincronizada: salas de chat, conferencias. 4. Comunidades en línea: MUDs/MOOs. 5. Agentes de software y agentes inteligentes, mundos virtuales y transformaciones. 6. Psicología social. 7. Redes sociales. 8. Computación social. 9. Técnicas de usabilidad colaborativa.

Objetivos. 1. Comparar los temas HCI en interacción individual con interacciones grupales. 2. Discutir los diferentes tópicos relacionados a aspectos sociales del software colaborativo. 3. Discutir tópicos HCI en software que incluye la intención humana. 4. Describir las diferencias entre comunicación síncrona y asíncrona. 5. Diseñar, prototipar y evaluar un groupware simple o aplicaciones de comunicación de grupos ilustrando los conocimientos de los conceptos enseñados en las unidades HC/Desarrollo de Software Centrado en el Humano., HC/Diseño de la Interfaz Gráfica de Usuario. y HC/Aspectos de Colaboración y Comunicación. 6. Participar en un equipo de proyectos para el cual algunas interacciones seas cara a cara y otras mediante entornos de software. 7. Describir las similitudes y diferencias entre la colaboración cara a cara y a través de un software colaborativo.

Bibliografía: [Baecker et al., 2000]

10 Diseño de interacción para nuevos ambientes. (4 horas) [Nivel Bloom 3]

Tópicos. 1. Diseño de interacción orientado a experiencias recíprocas simpáticas. 2. Ambientes presenciales, telepresenciales e inmersivos. 3. Interacción y emoción afectiva. 4. Inteligencia de ambiente. 5. Computación físico e interacción personificada.

Objetivos. 1. Comparar asuntos metodológicos y filosóficos involucrados en el diseño para usabilidad y diseño atractivo. 2. Discutir varios asuntos de orden social y ético levantados a raíz de los ambientes inmersivos y altos niveles de emoción en Interacción Humano-Computador. 3. Discutir aspectos de HCI involucrados en software interactivo que personifica un nivel de inteligencia. 4. Describir la diferencia entre diseño de interacción y la HCI tradicional. 5. Diseñar, prototipar y evaluar un sistema interactivo atractivo para entretenimiento o educación. 6. Evaluar las experiencias de personas en ambientes inmersivos. 7. Describir los asuntos involucrados con interfases tangibles de usuarios, gestos y una interacción presencial. 8. Describir los asuntos involucrados con involucrar los sentidos en experiencias interactivas.

Bibliografía: [Baecker et al., 2000]

11 Factores humanos y seguridad. (4 horas) [Nivel Bloom 3]

Tópicos. 1. Psicología aplicada y políticas de seguridad. 2. Diseño pensando en usabilidad y seguridad. 3. Ingeniería social. 4. Suplantación de identidad. 5. Adquisición de información confidencial de forma fraudulenta *Phishing*.

Objetivos. 1. Explicar el concepto de *phishing* y como reconocerlo. 2. Explicar el concepto de robo de identidad y cómo dificultarlo. 3. Diseñar una interfase de usuario con mecanismos de seguridad. 4. Discutir procedimientos que ayuden a reducir un ataque de ingeniería social. 5. Analizar una política de seguridad y/o procedimientos para mostrar donde funcionan y donde fallan. Hacer consideraciones de valor práctico.

Bibliografía: [Baecker et al., 2000]

12 PL/Programación Orientada a Objetos. (1 horas) [Nivel Bloom 3]

Tópicos. 1. Diseño orientado a objetos. 2. Encapsulación y ocultamiento de la información. 3. Separación de comportamiento e implementación. 4. Clases y subclasses. 5. Herencia (sobrescritura, despacho dinámico). 6. Polimorfismo (polimorfismo de subtipo vs. herencia). 7. Jerarquías de clases. 8. Clases de tipo colección y protocolos de iteración. 9. Representaciones internas de objetos y tablas de métodos.

Objetivos. 1. Justificar la filosofía de diseño orientado a objetos y los conceptos de encapsulación, abstracción, herencia y polimorfismo. 2. Diseñar, implementar, probar y depurar programas simples en un lenguaje de programación orientado a objetos. 3. Describir como los mecanismos de clases soportan encapsulación y ocultamiento de la información. 4. Diseñar, implementar y probar la implementación de la relación es-un IsKindOf entre objetos usando jerarquía de clases y herencia. 5. Comparar y contrastar las nociones de sobrecarga y sobrescritura de métodos en un lenguaje orientado a objetos. 6. Explicar la relación entre la estructura estática de una clase y la estructura dinámica de las instancias de dicha clases. 7. Describir como los iteradores acceden a los elementos de un contenedor.

Bibliografía: [Baecker et al., 2000]

BIBLIOGRAFÍA

[Baecker et al., 2000] Baecker, R., Buxton, W., and Grudin, J. (2000). Readings in Human-Computer Interaction: Toward the Year 2000. The Morgan Kaufmann Series in Interactive Technologies. Morgan Kaufmann, 2nd edition edition.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Tópicos en Seguridad
CODIGO	CC033
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC302
CARACTER	ELECTIVO
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

Demuestra que seguridad es algo que nos da cierto nivel de certeza con respecto a los ataques de ingreso de usuarios a áreas protegidas. El curso muestra la arquitectura de seguridad de Java 2.x

SUMILLA

El curso cubre arquitectura, diseño, e implementación de la Seguridad de Java 2.x

CONTENIDO ANALITICO

CC033 Tópicos en Seguridad	
<ul style="list-style-type: none">▪ Fundamentos de seguridad del computador y redes▪ La seguridad básica para el lenguaje Java▪ La arquitectura de seguridad en JDK 2.x▪ El uso de la arquitectura de seguridad▪ Creando los permisos y las pólizas de la arquitectura de seguridad al gusto del cliente▪ Objeto de seguridad▪ Programando Criptografía ▪ Direcciones Futuras	

BIBLIOGRAFIA

- Inside java 2 Platform Security by Li Gong Editorial: Boston : Addison-Wesley, ©2003.
- Cryptography: A Primer by A.G. Konheim Editorial: New York : Wiley, ©1981.
- Cryptography in C and C++, por Michael Welschenbach, David Kramer. Ed: 2, Publicado por Apress, 2005.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Patrones de Lenguaje Paralelo
CODIGO	CC034
CREDITOS	02 (dos)
PRE-REQUISITO	CC301
CARACTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (01) LABORATORIO (03)

OBJETIVO DEL CURSO

- Demuestra que el uso de patrones o el descubrimiento de ellos servirán como una herramienta de evolución de modelos de programación.
- Los patrones de programación paralela guiara a los programadores a tomar decisiones en cada paso del desarrollo del software paralelo

SUMILLA

El curso se enfocara en definir patrones en las áreas de descomposición, algoritmos, estructuras, y metodología.

CONTENIDO ANALITICO

CC034 Patrones de Lenguaje Paralelo	
<ul style="list-style-type: none">▪ Introducción: Computación paralela y las dificultades que enfrentan los programadores de programación paralela▪ Patrones paralelos mas importantes<ol style="list-style-type: none">a. Descomposición: descomposición de data y descomposición Funcionalb. Orden: orden debe ser preservado y orden no es importante pero el efecto al resultado es lo cuenta,c. Comunicación: Local, estructura de árbol recursiva, forma geométricas complicadasd. Dependencias: separables, Funcional, inseparable▪ Búsqueda de concurrencia en un problema de diseño y la descomposición de este en tareas concurrentes▪ Manejo de data a través de tareas	<ul style="list-style-type: none">▪ Crear la estructura de un algoritmo que efectivamente explota la concurrencia que se ha identificado durante el diseño▪ Conexión de las estructuras y de los algoritmos al API requerida para implementación de ellos▪ Construcción de control específico para implementar programas paralelos▪ Como trabajar con los ambientes de programación mas reconocidos: OpenMP, MPI, y Java▪ Tabla de patrones paralelos

BIBLIOGRAFIA

- Patterns for Parallel Programming by Timothy G. Mattson, Beverly A. Sanders, Berna L. Massingill. Editorial: Boston : Addison-Wesley, 2005.
- Uzi Vishkin. Optimal parallel pattern matching in strings. Information and Control , 67:91 {113, 1985
- David A. Koufaty A. Algoritmos paralelos para búsqueda de patrones. Tesis de Maestría, Universidad Simón Bolívar, Caracas, 1991

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Arquitectura y Programación de FPGA y VHDL
CODIGO	CC044
CREDITOS	04 (cuatro)
PRE-REQUISITO	NINGUNO
CARÁCTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

Demostrar la evolución de la tecnología en lógica programable y en la alta escala de integración de los circuitos. La metodología ha evolucionado y esto ha sido expresado por los lenguajes de descripción de los circuitos.

SUMILLA

El curso explica a los alumnos una metodología de diseño. Se enseña la descripción, y se simula la implementación.

CONTENIDO ANALITICO

CC044 Arquitectura y Programación de FPGA y VHDL	
<ul style="list-style-type: none">▪ Fundamentos Teóricos▪ Concepto y Estructura de FPGA▪ Familias de FPGA▪ CAD para el diseño sobre FPGA▪ Metodología de diseño▪ Descripción del diseño▪ Introducción al Lenguaje VHDL▪ Elementos sintácticos del VHDL▪ Descripción del Flujo de Datos▪ Descripción comportamental algorítmica▪ Descripción Estructural▪ Poniendo orden: Subprogramas, Paquetes, y Bibliotecas▪ Conceptos Avanzados en VHDL▪ VHDL para la simulación▪ VHDL para síntesis▪ Utilización del Lenguaje VHDL	

BIBLIOGRAFIA

- VHDL Lenguaje para síntesis y modelado de circuitos por Fernando Pardo Carpio; José A Boluda Grau Editorial: México : Alfaomega, ©2004
- <http://www.chu.es/%7Ejtpolagi/completo.htm>
- Rapid system prototyping with FPGAs, por R. C. Cofer, Benjamin F. Harding. Publicado por Elsevier, 2005.
- Fpga-Based System Design, por Wayne Wolf. Publicado por Prentice Hall, 2004.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Introducción al Compilador Paralelo
CODIGO	CC045
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC461
CARACTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Introducir conceptos de programación sin tener en cuenta el lenguaje y las reglas de syntax de este.
- Demostrar que cualquier lenguaje debe definir data y procedure en la forma más simple (primitivos). Además debe tener métodos para manipular data y procedures en forma abstracta.

SUMILLA

El curso muestra que un proceso computacional como entidades que residen en el computador y que los datos son entes abstractos usados por el proceso, durante la evolución de este. La evolución de un proceso es dirigido por un conjunto de reglas llamado programa. Se escribe programas para dirigir procesos.

CC045 Introducción al Compilador Paralelo	
<ul style="list-style-type: none">▪ Introducción▪ Pre-definición de los registros locales▪ Introducción al análisis de Lexical▪ RE a DFA: La construcción de Thompson y la construcción de subconjunto▪ La minimización de DFA▪ Resumen del análisis de Lexical▪ Introducción al Parser▪ Parser: Implementación descendente▪ Parser: Implementación ascendente▪ Elaboración de Semántica▪ Representación Intermedia	<ul style="list-style-type: none">▪ La abstracción de un Procedure.▪ Introducción a los problemas de generación de código▪ Código para expresiones, Bolean, y control de flujo▪ Arreglos y Cadenas (Arrays y String)▪ Cronograma de las instrucciones locales.▪ Introducción al algoritmo de colores de la teoría de grafos en la pre-definición de registro locales▪ Eliminación de código muerto.

BIBLIOGRAFIA

- <http://www.linux-magazine.es/issue/12/Intel90.pdf>
- <http://www.dcc.uchile.cl/~mmarin/parallel/openMP/index.html>
- <http://www.angelfire.com/dragon/letstry/tutorials/compiler/index.html>
- <http://csis.pace.edu/~bergin/Compiler.htm>
- Engineering a Compiler by Cooper and Torczon

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- Gail A. Alverson, William G. Griswold, Calvin Lin, David Notkin, and Lawrence Snyder. Abstractions for Portable, Scalable Parallel Programming. IEEE Transactions on Parallel and Distributed Systems, 1998.
- H.E. Bal. A Comparative Study of Five Parallel Programming Languages. In EurOpen Spring Conference on Open Distributed Systems, 1991.
- Heri E. Bal, M. Frans Kaashoek, and Andrew S. Tanenbaum. Orca: A Language for Parallel Programming of Distributed Systems. IEEE Transactions on Software Engineering, 1992.
- Hesham El-Rewini and Ted G. Lewis. Distributed and Parallel Computing. Manning Publications Co., Hardbound, 1997.
- Ian Foster. Designing and Building Parallel Programs. Addison-Wesley, 1995. [http://www-unix.mcs.anl.gov/dbpp/](http://www.unix.mcs.anl.gov/dbpp/).
- A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, and V. Sunderam. PVM: Parallel Virtual Machine, A Users' Guide and Tutorial for Networked Parallel Computing. MIT Press, 1994.
- Cameron Hughes and Tracey Hughes. Parallel and Distributed Programming Using C++. Addison Wesley, Aug. 2003.
- Bruce P. Lester. The Art of Parallel Programming. Prentice-Hall, Inc., 1993.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Lenguaje Funcional Paralelo
CODIGO	CC053
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC302, CC342
CARACTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Demuestra las componentes y sintaxis del lenguaje Haskell, y Haskell paralelo (pH).
- Se estudia una extensión al lenguaje Funcional Haskell que permite describir el hardware (functional hardware programming lenguaje) en particular el diseño de los chips.

SUMILLA

El curso cubre las primeras ideas de paralelismo para entender por qué es necesario trabajar en nuevas ideas para resolver el problema de programación paralela.

CONTENIDO ANALITICO

CC053 Lenguaje Funcional Paralelo	
<ul style="list-style-type: none">▪ Paralelismo en sistemas que permiten múltiples hilos. Lenguajes y compiladores▪ Programación Funcional: Funciones y tipos▪ Cálculo de Lambda: Una base para lenguajes Funcionales▪ Cálculo de Lambda con constantes y let-blocks▪ Tipos y tipo simple de inferencia▪ Tipo de Interface▪ Overloading, tipo de clases, y tipo de data algebraicos▪ Lista de entendimientos▪ Compilación de patrones de semejanza o igualdad y lista de entendimientos▪ Arrays y I-Structuras▪ M-Estructuras: Programación con estado indeterminado	<ul style="list-style-type: none">▪ M-Estructuras: Introducción a E/S monad▪ El estado de Monad y Friends▪ Hardware para programación paralela▪ Diseños para IFFT▪ Micro-arquitectura: Modulo para la búsqueda de IP▪ Tópicos en FIFO y control de concurrencia▪ Lenguaje paralelo con acciones atómicas y módulos▪ Grafos para dataflow estático▪ Dataflow grafos con comportamiento bien entendido▪ Dataflow dinámico▪ Primitivas del despachador para Bluespec

BIBLIOGRAFIA

- Implicit Parallel Programming in pH by Rishiyur S. Nikhil and Arvind
- <http://web.mit.edu/~jmaessen/Public/phc-front-end.tar.gz> - Compiler source code

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- P. Hudak. Conception, Evolution, and Application of Functional Programming Languages. ACM Computing Surveys 21(3):359-411, 1989
- R. Plasmeijer and M. van Eekelen. Functional Programming and Parallel Graph Rewriting. Addison-Wesley, Reading, MA, 1993
- C. Reade. Elements of Functional Programming. Addison-Wesley, Reading, MA, 1993

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Motores de Búsquedas
CODIGO	CC054
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC342
CARÁCTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Demostrar cómo han sido construidos los motores de búsqueda.

SUMILLA

El curso describe la arquitectura de motores de búsqueda y sus algoritmos. Los estudiantes implementaran, y utilizaran sus propias maquinas de búsquedas.

CONTENIDO ANALITICO

CC054 Motor de Búsquedas	
<ul style="list-style-type: none">▪ Introducción▪ Modelos▪ Evaluación de los datos encontrados▪ Lenguajes de búsqueda▪ Operaciones de búsqueda▪ Propiedades y lenguajes de texto y Multimedia▪ Operaciones de texto▪ Índices e interfaces de búsquedas ▪ Información recogida de sistemas distribuidos y paralelos	<ul style="list-style-type: none">▪ Interfaces del usuario y visualización▪ Evaluación de la información de Multimedia encontrada▪ Modelos y lenguajes de información de Multimedia encontrada▪ Índices y interfaces de búsquedas de información de Multimedia▪ Interfaces de búsqueda en la Web▪ Librerías para sistemas bibliográficos ▪ Librerías digitales

BIBLIOGRAFIA

- Modern Information Retrieval (Baeza-Yates and Ribeiro-Neto; Addison Wesley).1999
- Information Retrieval – Algoritmos and Heuristics (Grossman and Frieder; Springer). Editorial: Dordrecht ; [Great Britain] : Springer, ©2004.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Introducción a la Robotica
CODIGO	CC055
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC441, CF221
CARÁCTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

Provee a los estudiantes con el conocimiento de métodos de diseño para el análisis de sistemas inteligentes y robóticos.

SUMILLA

El curso cubre los elementos básicos de modelamiento, diseño, planeamiento y control de los sistemas del robot. Desarrolla las formas de pensamiento lógico y la capacidad de razonamiento, mediante el análisis de métodos heurísticos y su aplicación a problemas no algoritmizables o poco estructurados.

CONTENIDO ANALITICO

CC055 Introducción a la Robotica	
Introducción	Introducción a la Optimización
Manipuladores de los sistemas de los robots	Optimización numérica
Sensores y activadores	Evaluación de Monte Carlo y algoritmos evolucionarios
Control de bajo nivel en los robots	Máquinas de aprendizaje
Robot móviles	Clasificación de conjuntos de datos
Modelamiento de sistemas dinámicos	Redes neuronales
Sistemas continuos y discretos	Entrenamiento de redes neuronales. Aprendizaje a través del tiempo
Lógica formal y maquinas de Turing	Control optimo
Calculo de predicados: Lógica de primer orden y conjuntos difusos	Control neuronal robusto y adaptativo.
Sistemas expertos: Interfaces de Representación y conocimiento	Planeamiento de tareas y sistemas de multiagente
Teoría de la probabilidad	

BIBLIOGRAFIA

- An Introduction to Intelligent and Autonomous Control by Panos J. Antsaklis and Kevin M. Passino Editorial: Boston : Kluwer Academic, ©1993.
- Fundamentals of robotic mechanical systems, por Jorge Angeles. Publicado por Springer, 2003.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Calidad de Software
CODIGO	CC056
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC472 Ingeniería de Software I
CARACTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Introducir los conceptos, teoría y práctica del proceso de aseguramiento de la calidad del software

DESCRIPCION

Los profesionales desarrolladores de software requieren proveer productos que satisfagan completamente los requerimientos de los usuarios finales. Para que esto sea posible, se debe localizar y corregir defectos de los productos de forma eficiente y sistemática. Este curso tiene como finalidad abordar los conceptos y métodos relacionados.

CONTENIDO ANALÍTICO

1. Introducción al aseguramiento de la calidad
 - ¿Qué es?, ¿Cómo es medida? ¿Cómo es alcanzada?
2. Procesos de software
 - Modelos de proceso de software.
 - Programación extrema.
3. Pruebas
 - Pruebas sistemáticas
 - Pruebas de caja negra
 - Pruebas de caja blanca
 - Continuidad de pruebas
 - Automatización de pruebas
4. Inspección
 - Inspección sistemática
 - Inspección en el proceso de software
 - Inspección del código fuente
 - Inspección automatizada
5. Métricas
 - Métricas de software
 - Métricas de calidad del producto
 - Métricas de código y estructuras
 - Métricas del proceso
6. Confiabilidad y seguridad
 - Aspectos transversales y métodos

BIBLIOGRAFIA

- Stephen H. Kan. 2002. Metrics and Models in Software Quality Engineering (2nd ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- Richard E. Nance and James D. Arthur. 2002. *Managing Software Quality*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- G. Gordon Schulmeyer and James I. McManus (Eds.). 1998. *Handbook of Software Quality Assurance* (3rd Ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Ingeniería Web
CODIGO	CC057
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC472 Ingeniería de Software I
CARACTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

- Introducir metodologías estructuradas para la construcción de proyectos web de alta calidad

DESCRIPCION

La web ha llegado a ser la mayor plataforma de deliberación de información en el mundo. Muchos proyectos web son desarrollados usando técnicas ad-hoc, contribuyendo a la proliferación de problemas de usabilidad, mantenimiento, calidad y confiabilidad. En este curso se examinará métodos cuantificables y sistemáticos para desarrollar aplicaciones web confiables, usables y de alta calidad.

CONTENIDO ANALÍTICO

1. Introducción a la ingeniería web
2. Ingeniería de requerimientos para aplicaciones web
3. Modelamiento de aplicaciones web
4. Arquitecturas web: diseño y usabilidad
5. Tecnologías client-side
6. Administración de proyectos web
7. El proceso de desarrollo de aplicaciones web
8. Tecnologías server-side
9. Seguridad
10. Pruebas, operación y mantenimiento de aplicaciones web

BIBLIOGRAFIA

- Kappel, G., Proll, B. Reich, S. & Retschitzegger, W. (2006). Web Engineering, 1st ed. Hoboken, NJ: Wiley & Sons. ISBN: 04700-1554-3.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Física Computacional
CODIGO	CC063
CREDITOS	04 (cuatro)
PRE-REQUISITO	CF221, CC201
CARÁCTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

El uso de computadoras en Física, así como en la mayoría de otras ramas de la ciencia y la ingeniería, se ha incrementado muchísimo junto con el rápido desarrollo de hardware más rápido y barato. Este curso pretende dar al estudiante una base sólida en las principales técnicas computacionales utilizadas en la Física moderna. Es particularmente importante en este curso que los alumnos deban aprender haciendo. El curso está en consecuencia diseñado tal que una gran parte del tiempo de los estudiantes es consumido realmente programando específicos problemas físicos más que aprendiendo técnicas abstractas.

SUMILLA

El curso abordará los problemas en 4 (5) secciones generales:

Ecuaciones diferenciales ordinarias, como las de mecánica clásica.

Ecuaciones diferenciales parciales, tales como las ecuaciones de Maxwell y las ecuaciones de difusión y de Schrodinger.

Matrices, como los sistemas de ecuaciones y problemas de valores propios aplicados a la ecuación de Poisson y cálculos de estructura electrónica.

Monte Carlo y otros métodos de simulación, tales como el Algoritmo de Metrópolis y la dinámica molecular.

(Si el tiempo lo permite:) Álgebra computacional: una Introducción usando Mapple a los usos y abusos de la Computación algebraica en Física.

CONTENIDO ANALITICO

* Física Computacional

DESCRIPCION del curso

* Ecuaciones Diferenciales Ordinarias

o Tipos de ecuaciones diferenciales

o Método de Euler

+ Orden de precisión

+ Estabilidad

+ La ecuación de crecimiento

+ Aplicación a las ecuaciones diferenciales no lineales

+ Aplicación a las ecuaciones vectoriales

o El Método Leap-Frog

o El Método de Runge-Kutta

o El Método Predictor-Corrector

o El Método Intrínseco

o Resumen

o Problemas

- o Proyecto - Electrones en un campo magnético
 - + Un campo uniforme
 - # Unidades
 - # La solución analítica
 - # Elección de un algoritmo
 - + Campos eléctricos y magnéticos cruzados
 - + Campo eléctrico oscilante
 - + Su Informe
- * Ecuaciones Diferenciales Parciales
 - o Tipos de ecuaciones
 - o Ecuaciones elípticas - la ecuación de Laplace
 - o Ecuaciones hiperbólicas - ecuaciones de onda
 - + Un Algoritmo Simple
 - + Un Algoritmo mejorado - el Método Lax
 - + Ecuaciones no lineales
 - + Otros métodos para ecuaciones hiperbólicas
 - o Métodos de Euler y Lagrange
 - o Ecuaciones parabólicas - Difusión
 - + Un Método Simple
 - + El Método Dufort-Frankel
 - + Otros métodos
 - Métodos conservativos
 - + La ecuación de continuidad
 - + La ecuación de difusión
 - + Las ecuaciones de Maxwell
 - o Dispersión
 - o Problemas
 - o Proyecto - Código de fluidos de Lagrange
 - + Las ecuaciones diferenciales
 - + Condiciones de frontera
 - + Condiciones iniciales
 - + La Física
 - + Una mejora?
 - + El Informe
 - o Proyecto -- Solitones
 - + Introducción
 - + Discretización
 - + Física
- * Álgebra matricial
 - o Introducción
 - o Tipos de matrices
 - o Problemas simples de matrices
 - + Suma y Resta
 - + Multiplicación de matrices
 - o Ecuaciones elípticas - La ecuación de Poisson
 - + Una dimensión
 - + 2 o más dimensiones
 - o Sistemas de ecuaciones e inversión de la matriz
 - + Métodos exactos
 - + Métodos Iterativos
 - # El Método de Jacobi
 - # El Método de Gauss-Seidel
 - o Problemas de valores propios de Matrices
 - + La ecuación de Schrödinger
 - + Principios generales
 - + Diagonalización

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

- + El problema de valor propio generalizado
- + Diagonalización parcial
 - # Secuencia de Sturm
- Matrices dispersas y el Algoritmo de Lanczos
- o Problemas
- Proyecto -- Oscilaciones de una grúa
 - + Análisis
- Proyecto -- fonones en una cuasicristal
 - + Introducción
 - + La Red Fibonacci
 - + El Modelo
 - + La Física
- * Método Monte Carlo y Simulación
 - o Monte Carlo
 - + Generadores de número aleatorios
 - o Integración Monte-Carlo
 - o El Algoritmo de Metrópolis
 - + El modelo de Ising
 - + Promedios termodinámicos
 - o Quantum Monte-Carlo
 - o Dinámica Molecular
 - + Principios generales
 - o Problemas
 - Proyecto -- El modelo de Ising
 - + Introducción
 - + Modelo y método
 - + La Física
 - o Proyecto - Cálculo Cuántico Monte Carlo
 - + Introducción
 - + El Método
 - + La Física
- * Álgebra Computacional
 - o Introducción
 - o Principios básicos
 - o Solución de ecuaciones
 - o Ecuaciones diferenciales
 - o Otras características
 - o Ejemplo Final
 - Proyecto -- La aproximación de Thomas-Fermi
 - + Introducción
 - + Algunas ideas

Bibliografía

Lapack Numerical Library n.d. URL: <ftp://unix.hensa.ac.uk/pub/netlib/lapack/>

Metropolis N, Rosenbluth A W, Rosenbluth M N, Teller A H & Teller E 1953 J. Chem. Phys. 21, 1087.

Numerical Algoritmos Group n.d. URL: <http://www.nag.co.uk:70/>

Potter D 1973 Computational Physics Wiley Chichester.

Press W H, Flannery B P, Teukolsky S A & Vettering W T 1992 Numerical Recipes: The Art of Scientific Computing Cambridge University Press Cambridge. URL: <http://www.nr.com/>

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Wilkinson J H 1964 The Algebraic Eigenvalue Problem Clarendon Press Oxford.

Wolfram S 1991 Mathematica -- A System for Doing Mathematics by Computer Addison-Wesley Redwood City, California. ISBN 0 201 51502 4.

<http://www.cmth.ph.ic.ac.uk/angus/Lectures/compphys/node113.html>

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Química Computacional
CODIGO	CC064
CREDITOS	04 (cuatro)
PRE-REQUISITO	CQ112, CF122, CC102
CARÁCTER	electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

OBJETIVO DEL CURSO

El curso se centra en el aprendizaje de los principios de la química computacional y el diseño molecular basado en computadora. Ambos modelos mecánica molecular y mecánica cuántica son cubiertos. Los estudiantes aprenderán una variedad de técnicas de uso común, tales como optimización geométrica, localización de estados de transición, análisis conformacional y predicción de propiedades moleculares y espectroscópicas. Los estudiantes aprenderán lo básico de la aplicación de algoritmos claves, tales como minimización de Newton-Rhapson, y análisis del modo normal de movimientos vibratorios. Los estudiantes se familiarizarán con diferentes paquetes de software, incluyendo MOLDEN para la construcción de un modelo general, Gaussian, PC GAMESS, y Dalton para cálculos de química cuántica, y BOSS para simulaciones de líquidos. Se espera que los estudiantes que completen el curso sean capaces de hacer preguntas que puedan resolverse con enfoques modernos de cálculo y elijan herramientas adecuadas de cómputo para ayudarse en sus actuales o futuras investigaciones.

SUMILLA

CONTENIDO ANALITICO

Descripción del curso. Promesas de la química computacional
Mecánica molecular de vibraciones enlazadas. Métodos de minimización
Mecánica molecular: Fuerzas en moléculas poliatómicas
Laboratorio de Computación: Algoritmos de minimización
Fuerzas intermoleculares. Parametrización y pruebas de campos de fuerza
Docking
Laboratorio de Computación: Campos de fuerza y análisis conformacional
Método Monte Carlo: Principios
Método Monte Carlo: aplicaciones químicas y bioquímicas
Laboratorio de Computación: Monte Carlo
Fundamentos de la teoría MO
Teoría MO semi-empírica
Laboratorio de Computación: Métodos semi-empíricos
Teoría MO Ab initio: Bases
Teoría de Hartree-Fock: Principios y Aplicaciones
Laboratorio de Computación: la teoría de Hartree-Fock, los orbitales moleculares
Tratamiento de la correlación electrónica: MCSCF, Métodos CI
Tratamiento de la correlación electrónica: MP y Métodos CC
Laboratorio de Computación: el oscilador armónico en mecánica cuántica

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Espectroscopia vibracional y la termodinámica en fase gaseosa

Laboratorio de Computación: las vibraciones moleculares

Descripción de los estados excitados electrónicamente

Descripción de los efectos disolventes

Laboratorio de Computación: predicción de espectros ópticos

Teoría de Funcionales de la Densidad (DFT): principios

Teoría de Funcionales de la Densidad (DFT): aplicaciones en materiales

Estados de transición en el gas las reacciones de fase

Curso de proyectos

Asignaciones:

1. Algoritmos de minimización
2. Análisis conformacional Monte Carlo.
3. Aplicación de métodos semi-empíricos a reacciones químicas
4. Descripción de la reactividad química usando la teoría MO
5. Análisis vibracional
6. Espectros ópticos

Bibliografía:

Requerido: Frank Jensen, Introduction to Computational Chemistry, 2nd edition (Wiley) Hemisferio
ó

Christopher Cramer, Essentials of Computational Chemistry, 2nd edition (Wiley)

Recomendado: Andrew Leach, Molecular Modelling: Principles and Applications, 2nd Edition (Prentice Hal).

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Computación Bioinspirada
CODIGO	CC065
CREDITOS	04(cuatro)
PRE-REQUISITO	CC441 Inteligencia Artificial
CARÁCTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (02) PRACTICA(02) LABORATORIO(02)

FUNDAMENTACION DEL CURSO

La Computación Bioinspirada es el área de investigación que estudia las diferentes técnicas computacionales que tienen inspiración biológica, las cuales permiten desarrollar nuevas herramientas para la solución de problemas y pueden estar basadas en patrones naturales, en comportamiento de los seres vivos, en la estructura misma de los organismos, etc.

SUMILLA

1. Introducción a la Computación Bioinspirada 2. Conceptualización 3. IS/Búsqueda Avanzada. 4. IS/Aprendizaje de Máquina. 5. Inteligencia de enjambre 6. Sistema inmunológico artificial 7. Geometría fractal 8. Vida artificial 9. Computación basada en ADN 10. Computación cuántica

OBJETIVO DEL CURSO

Elaborar modelos teóricos inspirados biológicamente, que puedan ser implementados en las computadoras, a fin de reproducir su funcionamiento tanto cualitativa como cuantitativamente.

Estudiar los fenómenos naturales, los procesos, modelos teóricos, para construir algoritmos capaces de resolver problemas complejos.

CONTRIBUCION A LA FORMACION PROFESIONAL Y FORMACION GENERAL

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- a) Aplicar conocimientos de Computación y de matemática apropiadas para la disciplina.
- h) Incorporarse a un proceso de aprendizaje profesional continuo.
- i) Utilizar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- j) Aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida.

CONTENIDO ANALITICO

UNIDAD 1: Introducción a la Computación Bioinspirada (2 horas)

OBJETIVO:

Conocer el fundamento de la Computación Bioinspirada.

Diferenciar las diferentes ramas de la Computación naturalmente inspirada.

CONTENIDO:

Introducción

Motivación

La filosofía de la Computación natural

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Computación inspirada por la naturaleza
Simulación y emulación de la naturaleza en las computadoras
Computación con materiales naturales

UNIDAD 2: Conceptualización (4 horas)

OBJETIVO:

Conocer los conceptos básicos en los que se fundamentan la Computación Bioinspirada
Caracterizar los sistemas bioinspirados Identificar los comportamientos complejos

CONTENIDO:

Entidades Individuales y Agentes.
Procesamiento paralelo y distribuido.
Interactividad.
Adaptación.
Auto Organización.
Complejidad, emergencia y reduccionismo.
Determinismo.
Teoría del Caos.
Fractales.

UNIDAD 3: IS/Búsqueda Avanzada.(8 horas)

OBJETIVO:

Explicar que son los algoritmos genéticos y contrastar su efectividad con las soluciones de problemas clásicos y técnicas de búsqueda clásicas.
Explicar como simulated annealing puede ser usado para reducir la complejidad y contrastar su operación con técnicas de búsqueda clásica.
Aplicar técnicas de búsqueda local a un dominio clásico.

CONTENIDO:

Heurísticas.
Búsqueda local y optimización.
Subiendo a la colina Hill climbing.
Algoritmos genéticos.
Simulated annealing.
Estrategias local de recorte de caminos local beam search.
Búsquedas en el adversario para juegos

UNIDAD 4: IS/Aprendizaje de Maquina.(10 horas)

OBJETIVO:

Explicar las diferencias entre tres principales estilos de aprendizaje: supervisado, no supervisado y por refuerzo.
Implementar algoritmos simples para aprendizaje supervisado, aprendizaje por refuerzo y aprendizaje no supervisado.
Determinar cuales de los tres estilos de aprendizaje es apropiado para un dominio de problema en particular.
Comparar y contrastar cada una de las siguientes técnicas, proveer ejemplos de cuando cada estrategia es superior: arboles de decisión, redes neuronales y redes de creencia.
Implementar de manera apropiada un sistema de aprendizaje simple, usando arboles de decisión, redes neuronales y/o redes de creencia.

Caracterizar el estado del arte en teoría del aprendizaje, incluyendo logros y defectos.
Explicar el algoritmo del vecino mas cercano y su lugar dentro de la teoría del aprendizaje.
Explicar el problema de sobreajuste, a través de técnicas para detectar y manejar el problema.

CONTENIDO:

Definición y ejemplos de aprendizaje de maquina.
Aprendizaje inductivo, aprendizaje basado en estadística, aprendizaje por refuerzo.
Aprendizaje supervisado.
Árboles de aprendizaje por decisión.
Aprendizaje por redes neuronales .
Redes de aprendizaje por creencia.

Algoritmo del vecino más cercano.
Teoría de aprendizaje.
El problema del sobreajuste.
Aprendizaje no supervisado.
Aprendizaje por refuerzo.

UNIDAD 5: Inteligencia de enjambre (6 horas)

OBJETIVO:

Conocer la inteligencia de enjambre.
Implementar la colonia de hormigas.
Estudiar la optimización de enjambre de partículas

CONTENIDO:

Introducción
Colonias de hormigas: inspiración biológica.
Colonias de hormigas: algoritmo básico.
Optimización de enjambre de partículas: inspiración biológica.
Optimización de enjambre de partículas: algoritmo básico.
Aplicación de la inteligencia de enjambre.
Tendencias y problemas abiertos.

UNIDAD 6: Sistema inmunológico artificial (6 horas)

OBJETIVO:

Conocer la motivación de los sistemas inmunológicos.

CONTENIDO:

Motivación biológica.
Sistemas inmunológicos.
Sistemas inmunológicos artificiales.
Redes de sistemas inmunológicos.
Principios de diseño.
Ámbito de aplicación de los sistemas inmunológicos.
Tendencias y problemas abiertos.

UNIDAD 7: Geometría fractal (6 horas)

OBJETIVO:

Estudiar la geometría fractal.
Estudiar los autómatas celulares.
Implementar autómatas celulares

CONTENIDO:

Introducción.
Dimensión fractal.
Naturaleza de la geometría fractal.
Autómatas celulares.
Autómatas celulares y sistemas dinámicos.
sistema de Lindenmayer.
Tendencias y problemas abiertos

UNIDAD 8: Vida artificial (6 horas)

OBJETIVO:

Estudiar como generar vida artificial.
Implementar autómatas celulares para generar vida artificial

CONTENIDO:

Introducción.
La esencia de la vida.
Proyectos basados en vida artificial.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Autómatas Celulares para la creación de vida artificial.
Ámbito de aplicación de la vida artificial.
Tendencias y problemas abiertos.

UNIDAD 9: Computación basada en ADN (6 horas)

OBJETIVO:

Estudiar la Computación basada en ADN.
Estudiar de la potencia computacional de las variantes consideradas, comparada con la potencia de las máquinas de Turing

CONTENIDO:

Introducción.
Motivación biológica.
Filtrando modelos.
Modelos Formales.
Computadores de ADN universales.
Ámbito de aplicación de la vida artificial.
Tendencias y problemas abiertos.

UNIDAD 10: Computación cuántica (6 horas)

OBJETIVO:

Estudiar la Computación cuántica.
Codificar algoritmos cuánticos.
Simular y calcular la eficiencia de algoritmos cuánticos.

CONTENIDO:

Introducción.
conceptos básicos de la teoría cuántica.
Principales mecanismos de la teoría cuántica.
Algoritmos cuánticos.
Computadores cuánticos.
Ámbito de aplicación de la vida artificial.
Tendencias y problemas abiertos

BIBLIOGRAFIA

- [Baldi and Brunak, 2001] Baldi, P. and Brunak, S. (2001). Bioinformatics: the machine learning approach. The MIT Press.
- [De Castro, 2006] De Castro, L. (2006). Fundamentals of natural computing: basic concepts, algorithms, and applications. CRC Press.
- [Dorigo and Stützle, 2004] Dorigo, M. and Stützle, T. (2004). Ant colony optimization. the MIT Press.
- [Goldberg, 1989] Goldberg, D. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley.
- [Haykin, 1999] Haykin, S. (1999). Neural networks: A Comprehensive Foundation. Prentice Hall.
- [Kennedy et al., 2001] Kennedy, J., Eberhart, R., and Yuhui, S. (2001). Swarm intelligence. Morgan Kaufmann Publishers.
- [Mitchell, 1998] Mitchell, M. (1998). An introduction to genetic algorithms. The MIT press

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Programación de Dispositivos Móviles
CODIGO	CC066
CREDITOS	03(tres)
PRE-REQUISITO	CC362 Computación Centrada en Redes
CARÁCTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (01) PRACTICA(02) LABORATORIO(02)

FUNDAMENTACION DEL CURSO

El siempre creciente desarrollo de las tecnologías de comunicación y la información hace que exista una marcada tendencia a establecer mas redes de computadores que permitan una mejor gestión de la información
En este curso se brindará a los participantes una introducción a los problemas que conlleva la comunicación entre computadores, a través del estudio e implementación de protocolos de comunicación como TCP/IP y la implementación de software sobre estos protocolos

SUMILLA

1. NC/Seguridad de Red.2. NC/Organización de la Web.3. NC/Aplicaciones en redes.4. NC/Tecnología de Datos Multimedia.5. NC/Computación Móvil e Inalámbrica

OBJETIVO GENERAL

Que el alumno implemente y/o modifique un protocolo de comunicación de datos.
Que el alumno dominio las técnicas de transmisión de datos utilizadas por los protocolos de red existentes

CONTRIBUCION A LA FORMACION PROFESIONAL Y FORMACION GENERAL

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- b) Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución.
- c) Diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas.
- e) Entender correctamente las implicatorias profesionales, éticas, legales, de seguridad y sociales de la profesión.
- g) Analizar el impacto local y global de la Computación sobre los individuos, organizaciones y sociedad.
- i) Utilizar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- j) Aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida.

CONTENIDOS:

UNIDAD 1: NC/Seguridad de Red.(3 horas)

OBJETIVO:

Describir las mejoras hechas por el IPSec al IPv4.

Identificar protocolos usados para mejorar la comunicación en Internet y escoger el protocolo apropiado para un determinado caso.

Entender y detectar intrusiones.

Discutir las ideas fundamentales de criptografía de clave pública.

Describir como la criptografía de clave pública trabaja.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Distinguir entre el uso de algoritmos de clave privada y pública.

Resumir los protocolos comunes de autenticación.

Generar y distribuir un par de claves PGP y usar el paquete PGP para enviar un mensaje de correo electrónico encriptado.

Resumir las capacidades y limitaciones del significado de criptografía que se encuentran disponibles para el público en general.

Describir y discutir recientes ataques de seguridad exitosos.

Resumir las fortalezas y debilidades asociadas con diferentes abordajes de seguridad.

CONTENIDO:

Fundamentos de criptografía: a) Algoritmos de clave pública. b) Algoritmos de clave privada.

Protocolos de autenticación

Firmas digitales y ejemplos.

Tipos de ataques por red: negación de servicio (Denial of service), desborde flooding, sniffing y desvío de tráfico, ataques de integridad de mensajes, usurpación de identidad, ataques de vulnerabilidades (desborde de buffers, caballos de troya, puertas traseras), por dentro del ataque, infraestructura (secuestro de DNS, ruteo nulo- route blackholing, comportamiento inadecuado de ruteadores que descartan tráfico), etc.

Uso de contraseñas y mecanismos de control de acceso.

Herramientas y estrategias de defensa básica. a) Detección de intrusos. b) Firewalls. c) Detección de malware. d)

Kerberos. e) IPSec. f) Redes privadas virtuales (Virtual Private Networks). g) Traducción de direcciones de red.

Políticas de gerenciamiento de recursos en redes.

Auditoria y logging.

UNIDAD 2: NC/Organización de la Web.(3 horas)

OBJETIVO:

Explicar los diferentes roles y responsabilidades de los clientes y servidores para un conjunto de posibles aplicaciones.

Seleccionar un conjunto de herramientas que aseguren un método eficiente para implementar varias posibilidades cliente-servidor.

Diseñar y construir una aplicación interactiva simple basada en la web (por ejemplo, un formulario web simple que colecte información desde el cliente y almacene esto en un archivo en el servidor y un servidor que responda a los datos del formulario y produzca un resultado.)

CONTENIDO:

Tecnologías Web. a) Programas del lado del servidor. b) Scripts del lado del cliente. c) EL concepto de applet.

Características de los servidores web. a) Manejo de permisos. b) Administración de archivos. c) Capacidades de las arquitecturas comunes de servidores.

Rol de las computadoras cliente.

Naturaleza de la relación cliente-servidor.

Protocolos Web.

Herramientas de soporte para la creación y mantenimiento de sitios web.

Desarrollo de servidores de información Internet (Internet Information Servers).

Publicación de información y aplicaciones.

Grid Computing, cluster, mallas (mesh).

Servicios Web, Web 2.0, Ajax.

UNIDAD 3: NC/Aplicaciones en redes.(12 horas)

OBJETIVO:

Ilustrar como aplicaciones web interactivas cliente-servidor de tamaño medio pueden ser construidas usando diferentes tipos de tecnologías web.

Demostrar como implementar un sitio web basado en bases de datos, explicando las tecnologías relevantes involucradas en cada capa de la arquitectura y los límites de desempeño correspondientes.

Ilustrar el estado actual de la efectividad de una búsqueda Web.

Implementar una aplicación que invoque el API de una aplicación basada en la Web.

Implementar un sistema distribuido utilizando dos frameworks de objetos distribuidos y compararlos con respecto al desempeño y seguridad.

Discutir asuntos de seguridad y estrategias en una aplicación empresarial basada en web.

CONTENIDO:

Protocolos en la capa de aplicación
Interfaces Web: navegadores y APIs.
Tecnología de búsqueda en la web.
Principios de la ingeniería web.
Sitios web dirigidos a bases de datos.
Llamadas a procedimientos remotos (RPC).
Objetos ligeros distribuidos.
El rol del middleware.
Herramientas de soporte.
Tópicos de seguridad en sistemas de objetos distribuidos.
Aplicaciones empresariales basadas en web. a) Arquitecturas orientadas a servicios.

UNIDAD 4: NC/Tecnología de Datos Multimedia.(9 horas)

OBJETIVO:

Para cada uno de los varios estándares Multimedia, describir en un lenguaje no técnico lo que el estándar realiza y explicar como los aspectos de percepción humana podrían ser sensibles a las limitaciones de dicho estándar
Evaluar el potencial de un sistema de computadores para alojar una aplicación de un grupo de posibles aplicaciones Multimedia, incluyendo una evaluación de requerimientos de sistemas Multimedia en la tecnología de redes sobre la que se trabaja.

Describir las características de un sistema de computador (incluyendo identificación de herramientas de soporte y estándares apropiados) que tienen que alojar la implementación de una de varias aplicaciones Multimedia posibles.

Implementar una aplicación Multimedia de tamaño moderado.

CONTENIDO:

Sonido y audio, imágenes y gráficos, animación y vídeo
Estándares Multimedia (audio, música, imágenes, telefonía, vídeo, TV).
Planeamiento de capacidad y asuntos de desempeño.
Dispositivos de entrada/salida (scanners, cámaras digitales, pantallas de tacto, activación por voz).
Teclado MIDI, sintetizadores.
Estándares de almacenamiento (discos ópticos magnéticos, CD-ROM, DVD).
Servidores Multimedia y sistemas de archivos.
Herramientas para soporte al desarrollo Multimedia

UNIDAD 5: NC/Computación Móvil e Inalámbrica.(18 horas)

OBJETIVO:

Describir las principales características de IP móvil y explicar como difiere del IP con respecto a la administración de la movilidad, ubicación y desempeño.

Ilustrar (con agentes locales y externos) como el email u otro tipo de tráfico es ruteado usando IP móvil

Implementar una aplicación simple que se base en comunicación móvil e inalámbrica de datos.

Describir las áreas actuales y de interés emergente en computación inalámbrica y móvil así como evaluar las capacidades, limitaciones y potencial en cada uno.

CONTENIDO:

Vista general de la historia, evolución y compatibilidad de los estándares inalámbricos
Los problemas especiales de la computación inalámbrica y móvil
Redes inalámbricas de área local y redes basadas en satélites
Ciclos inalámbricos locales.
Protocolos de Internet móvil
Adaptación conciente a dispositivos móviles
Extendiendo el modelo cliente servidor para adaptarse a la movilidad.
Acceso a datos móviles: diseminación de datos en el servidor y administración del cache del cliente.
Soporte de paquetes de software para Computación inalámbrica y móvil.
El rol del middleware y herramientas de soporte.
Problemas de desempeño.
Tecnologías emergentes

Referencias

- [Harvey, 2004] Harvey, D. (2004). Redes de computadoras y Sistemas Operativos. PEARSON EDUCACION.
- [Stalling, 2005] Stalling, W. (2005). Comunicaciones y Redes de Computadoras. Prentice Hall, 7ma edition. 8420541109.
- [Stevens, 2007] Stevens, R. (2007). Unix Network Programming. Addison-Wesley, 2007.
- [Tanenbaum, 2005] Tanenbaum, A. S. (2005). Redes de Computadoras. Prentice-Hall, 7th edition.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Seguridad en Computación
CODIGO	CC067
CREDITOS	03(tres)
PRE-REQUISITO	CM094, CC262
CARÁCTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (01) PRACTICA(02) LABORATORIO(02)

FUNDAMENTACION DEL CURSO

Hoy en día la información es uno de los activos mas preciados en cualquier organización. Este cursos esta orientado a poder brindar al alumno los elementos de seguridad orientados a proteger la información de la organización y principalmente poder preveer los posibles problemas relacionados con este rubro. Esta materia involucra el desarrollo de una actitud preventiva por parte del alumno en todas las áreas relacionadas al desarrollo de software

SUMILLA

1. PF/Fundamentos de seguridad de la Información2. PF/Programación segura.3. OS/Modelos de seguridad. 4. AL/Algoritmos Cristalográficos5. NC/Seguridad de Red.6. NC/Administración de Redes.7. Factores humanos y seguridad.8. SP/Operaciones de seguridad.9. PL/Maquinas Virtuales.

OBJETIVO GENERAL

Discutir a un nivel intermedio avanzado los los fundamentos de la Seguridad Informática
Brindar los diferentes aspectos que presenta el código malicioso.
Que el alumno conozca los conceptos de criptografía y seguridad en redes de computadoras.
Discutir y analizar junto con el alumno los aspectos de la Seguridad en Internet.

CONTRIBUCION A LA FORMACION PROFESIONAL Y FORMACION GENERAL

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- Aplicar conocimientos de computación y de matemática apropiadas para la disciplina.
- Analizar problemas e identificar y definir los requerimientos computacionales apropiados para su solución.
- Diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas.
- Analizar el impacto local y global de la computación sobre los individuos, organizaciones y sociedad.
- Incorporarse a un proceso de aprendizaje profesional continuo.
- Utilizar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- Aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida.

CONTENIDOS

UNIDAD 1: PF/Fundamentos de seguridad de la Información(4 horas)

OBJETIVO

Explicar los objetivos de la seguridad de la información.
Analizar los puntos de equilibrio inherentes a la seguridad.
Explicar la importancia y las aplicaciones de la confidencialidad, integridad y disponibilidad.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Entender las categorías básicas de las amenazas a las computadoras y redes.
Discutir problemas para crear políticas de seguridad para una organización de gran tamaño.
Defender la necesidad de la protección y la seguridad y el rol de consideraciones éticas en el uso de computadores.

CONTENIDO

Rol y propósito de la seguridad en las computadoras y redes.
Objetivos de seguridad: confidencialidad, integridad y disponibilidad.
Políticas y estándares de seguridad.
Mentalidad orientada a la seguridad.
Defensa en profundidad.
Amenazas comunes: worms, virus, troyanos, bloqueo de acceso a servicios.
Estimación de riesgos y análisis de costo beneficio.
Seguridad vs usabilidad.

UNIDAD 2: PF/Programación segura.(4 horas)

OBJETIVO

Reescribir un simple programa para remover una simple vulnerabilidad.
Explicar porque es o no es posible el desborde en un lenguaje de programación de dominio del estudiante.
Explicar porque una o mas construcciones de lenguaje pueden originar problemas de seguridad como desborde.

CONTENIDO

Validaciones importantes para evitar desbordes en array y cadenas.
Construcciones en lenguajes de programación para evitar problemas de seguridad.
¿Como los atacantes usan el desborde para destruir la pila (stack) en tiempo de ejecución?

UNIDAD 3: OS/Modelos de seguridad.(4 horas)

OBJETIVO

Comparar y contrastar métodos existentes para la implementación de seguridad.
Comparar y contrastar las fortalezas y debilidades de dos o mas sistemas operativos actuales con respecto a la seguridad.
Comparar y contrastar las fortalezas y debilidades en seguridad de dos o mas sistemas operativos actuales con respecto a la gestión de la recuperación
Describir la matriz de control de accesos y como esta se relaciona la Lista de control de accesos (Access Control Lists-ACLs.) y a las listas de capacidades (C-Lists)
Aplicar el modelo de Biba para el chequeo de las entradas de un programa (contaminada y descontaminada por ejemplo).
Describir como el modelo Bell-LaPadula combina mecanismos de control de acceso obligatorios y a discreción así como explicar la formulación de lattice de Bell-LaPadula y Biba.
Comparar y contrastar dos modelos de seguridad.
Relacionar modelos de seguridad particular con los modelos del ciclo de desarrollo de software.
Aplicar modelos particulares a diferentes entornos y seleccionar el modelo que mejor captura el entorno.

CONTENIDO

Modelos de protección
Protección de memoria.
Encriptación
Gestión de la recuperación
Tipos de control de acceso: obligatorio, a discreción, controlado por el origen, basado en el rol.
Modelo de matriz de control de acceso.
El modelo Harrison-Russo-Ullman y la indecisión en temas de seguridad.
Modelos de confidencialidad tales como Bell-LaPadula.
Modelos de integridad tales como Biba y Clark-Wilson.
Modelos de conflicto de interés tales como la muralla china.

UNIDAD 4: AL/Algoritmos Cristalográficos(4 horas)

OBJETIVO

Describir algoritmos numérico-teóricos básicos eficientes, incluyendo el máximo común divisor, inversa multiplicativa mod n y elevar a potencias mod n.

Plan de estudios 2011 – Especialidad de Ciencia de la Computación

Describir al menos un cripto-sistema de llave pública, incluyendo una suposición necesaria de complejidad teórica sobre su seguridad.

Crear extensiones simples de protocolos criptográficos, usando protocolos conocidos y primitivas criptográficas.

CONTENIDO

Revisión histórica de la criptografía

criptografía de llaves privadas y el problema del intercambio de llaves.

criptografía de llaves públicas.

Firmas digitales.

Protocolos de seguridad.

Aplicaciones (pruebas de cero-conocimiento, autenticación y otros).

UNIDAD 5: NC/Seguridad de Red.(8 horas)

OBJETIVO

Describir las mejoras hechas por el IPSec al IPv4.

Identificar protocolos usados para mejorar la comunicación en Internet y escoger el protocolo apropiado para un determinado caso.

Entender y detectar intrusiones.

Discutir las ideas fundamentales de criptografía de clave pública.

Describir cómo la criptografía de clave pública trabaja.

Distinguir entre el uso de algoritmos de clave privada y pública.

Resumir los protocolos comunes de autenticación

Generar y distribuir un par de claves PGP y usar el paquete PGP para enviar un mensaje de correo electrónico encriptado.

Resumir las capacidades y limitaciones del significado de criptografía que se encuentran disponibles para el público en general.

Describir y discutir recientes ataques de seguridad exitosos.

Resumir las fortalezas y debilidades asociadas con diferentes abordajes de seguridad.

CONTENIDO

Fundamentos de criptografía: a) Algoritmos de clave pública. b) Algoritmos de clave privada.

Protocolos de autenticación

Firmas digitales y ejemplos.

Tipos de ataques por red: negación de servicio (Denial of service), desborde flooding, sniffing y desvío de tráfico, ataques de integridad de mensajes, usurpación de identidad, ataques de vulnerabilidades (desborde de buffers, caballos de troya, puertas traseras), por dentro del ataque, infraestructura (secuestro de DNS, ruteo nulo- route blackholing, comportamiento inadecuado de routers que descartan tráfico), etc.

Uso de contraseñas y mecanismos de control de acceso.

Herramientas y estrategias de defensa básica a) Detección de intrusos. b) Firewalls. c) Detección de malware. d)

Kerberos. e) IPSec. f) Redes privadas virtuales (Virtual Private Networks). g) Traducción de direcciones de red.

Políticas de gerenciamiento de recursos en redes.

Auditoría y logging

UNIDAD 6: NC/Administración de Redes.(8 horas)

OBJETIVO

Explicar los asuntos de la administración de redes resaltando amenazas de seguridad, virus, gusanos, troyanos y ataques de negación de servicios.

Desarrollar una estrategia para asegurar niveles apropiados de seguridad en un sistema diseñado para un propósito particular.

Implementar un muro de fuego (firewall) de red.

CONTENIDO

Vista general de la administración de redes.

Uso de contraseñas y mecanismos de control de acceso.

Nombres de dominio y servicios de nombre.

Proveedores de servicio de Internet (ISPs).

Seguridad y muros de fuego (firewalls).

Asuntos de calidad de servicio: desempeño, recuperación de errores.

UNIDAD 7: Factores humanos y seguridad.(2 horas)

OBJETIVO

Explicar el concepto de phishing y como reconocerlo.

Explicar el concepto de robo de identidad y como dificultarlo.

Diseñar una interfase de usuario con mecanismos de seguridad.

Discutir procedimientos que ayuden a reducir un ataque de ingeniería social.

Analizar una política de seguridad y/o procedimientos para mostrar donde funcionan y donde fallan.

Hacer consideraciones de valor practico.

CONTENIDO

Psicología aplicada y políticas de seguridad.

Diseño pensando en usabilidad y seguridad.

Ingeniería social.

Suplantación de identidad

Adquisición de información confidencial de forma fraudulenta Phishing.

UNIDAD 8: SP/Operaciones de seguridad.(8 horas)

OBJETIVO

Desarrollar un plan de recuperación de incidentes para manejar los compromisos de una organización

Analizar los procedimientos de seguridad establecidos en busca de puntos débiles que un atacante podría explotar y explicar como los mismos podrían fallar.

Proponer medidas de seguridad apropiadas para diferentes situaciones.

Explicar para una comunidad de usuarios no expertos en seguridad que medidas ellos deben seguir y porque en una situación en la que sus trabajos no sean relacionados con seguridad.

CONTENIDO

Seguridad física

Control de acceso fisico

Control de acceso de personal.

Seguridad Operativa.

Políticas de seguridad para sistemas/redes.

Recuperación y respuesta.

Manejando problemas técnicos y humanos.

UNIDAD 9: PL/Maquinas Virtuales.(3 horas)

OBJETIVO

Explicar como los programas ejecutables pueden violar la seguridad de sistema computacional accediendo a archivos de disco y memoria.

CONTENIDO

Temas de seguridad relacionados a ejecutar código sobre una máquina externa.

Referencias

[Bellovin, 1989] Bellovin, S. (1989). Security problems in the tcp/ip protocol suite. ACM Computer Communications Review, 19(2):32–48.

[Caballero, 1996] Caballero, P. (1996). Introducción a la criptografía, volume Textos Universitarios. Ra-Ma.

[Cano, 1998] Cano, J. J. (1998). Pautas y recomendaciones para elaborar políticas de seguridad informática. Technical report, Universidad de Los Andes.

[Department of Defense, 1985] Department of Defense (1985). Password Management Guideline (Green Book). Department of Defense. CSC-STD-002-85.

[FIPS PUB, 1994] FIPS PUB (1994). Guideline for the analysis of local area network security. Technical Report 191, FIPS PUB.

[Fuster et al., 1997] Fuster, A., De la Guia, D., Hernandez, L., Montoya, F., and Muñoz, J. (1997). Técnicas criptográficas de Protección de Datos. Ra-Ma.

[ICSA Inc., 1998] ICSA Inc. (1998). An introduction to intrusion detection and assessment. Technical report, ICSA Inc.

- [NCSC, 1987] NCSC (1987). A guide to understanding discretionary access control in trusted systems. Technical report, National Computer Security Center. NCSC-TG-003.
- [Neuman and Ts'o, 1994] Neuman, B. C. and Ts'o, T. (1994). Kerberos: An authentication service for computer networks. IEEE Communications Magazine, 32(9):33–38.
- [of Computer Engineering, 1995] of Computer Engineering, D. (1995). A structured approach to computer security. Technical report, Chalmers University of Technology.
- [Ramio Aguirre, 1999] Ramio Aguirre, J. (1999). Aplicaciones criptográficas Dpto. de Publicaciones EUI-UPM, segunda edición.
- [Russel and Gangemi, 1991] Russel, D. and Gangemi, G. (1991). Computer Security Basics. O'Reilly and Associates.
- [Sandhu and Samarati, 1994] Sandhu, R. S. and Samarati, P. (1994). Authentication, access control and intrusion detection. IEEE Communications, 32(9).
- [Seberry and Pieprzyk, 1989] Seberry, J. and Pieprzyk, J. (1989). Cryptography. An Introduction to Computer Security. Prentice-Hall.
- [Spafford, 1998] Spafford, E. H. (1998). The internet worm program: An analysis. Technical report, Purdue. CSD-TR-823.
- [Stallings, 1999] Stallings, W. (1999). Cryptography and Network Security. Principles and Practice. Prentice Hall International Editions, 2da edición.
- [Tinto, 1989] Tinto, M. (1989). Computer viruses: prevention, detection and treatment. Technical Report 001, National Computer Security Center.
- [Venerma, 1998] Venerma, W. (1998). Tcpwrapper: networking monitoring, access control and booby traps. Technical report, Mathematics and Computing Science, Eindhoven University of Technology.
- [William, 1995] William, S. (1995). Network and Internetwork Security, Principles and Practice. Prentice-Hall.

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Tópicos en Base de Datos
CODIGO	CC071
CREDITOS	04 (cuatro)
PRE-REQUISITO	CC331
CARACTER	Electivo
MODALIDAD	Semestral
HORAS	TEORIA (03) LABORATORIO (03)

FUNDAMENTACION

La Gestión de la Información (IM-Information Management) juega un rol principal en casi todas las áreas donde los computadores son usados. Esta área incluye la captura, digitalización, representación, organización, transformación y presentación de información; algoritmos para mejorar la eficiencia y efectividad del acceso y actualización de información almacenada, modelamiento de datos y abstracción, y técnicas de almacenamiento de archivos físicos. Este también abarca la seguridad de la información, privacidad, integridad y protección en un ambiente compartido. Los estudiantes necesitan ser capaces de desarrollar modelos de datos conceptuales y físicos, determinar que métodos de IM y técnicas son apropiados para un problema dado, y ser capaces de seleccionar e implementar una apropiada solución de IM que refleje todas las restricciones aplicables, incluyendo escalabilidad y usabilidad.

OBJETIVO DEL CURSO

- Hacer que el alumno entienda las diferentes aplicaciones que tienen las bases de datos, en las diversas áreas de conocimiento.
- Mostrar las formas adecuadas de almacenamiento de información basada en sus diversos enfoques y su posterior recuperación de información.

CONTRIBUCION A LOS RESULTADOS

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- Habilidad para aplicar conocimiento de Computación y de matemática apropiadas para la disciplina.
- Habilidad para analizar un problema e identificar y definir los requerimientos computacionales apropiados para su solución.
- Habilidad para diseñar, implementar y evaluar un sistema, proceso, componente o programa computacional para alcanzar las necesidades deseadas.
- Habilidad para trabajar efectivamente en equipos para cumplir con un objetivo común.
- Correcto entendimiento de las implicatorias profesionales, éticas, legales, de seguridad y sociales de la profesión.
- Habilidad para usar técnicas y herramientas actuales necesarias para la práctica de la Computación.
- Habilidad para aplicar la base matemática, principios de algoritmos y la teoría de la Ciencia de la Computación en el modelamiento y diseño de sistemas computacionales de tal manera que demuestre comprensión de los puntos de equilibrio involucrados en la opción escogida.

- Habilidad para aplicar los principios de desarrollo y diseño en la construcción de sistemas de software de complejidad variable.

CONTENIDO ANALITICO

1 Base de Datos Relacionales.

Tópicos. I: Mapeo del esquema conceptual al esquema relacional. II: Entidad e integridad referencial. III: Álgebra relacional y cálculo relacional.

Objetivos. 1. Preparar un esquema relacional de un modelo conceptual usando el modelo entidad-relación. 2. Explicar y demostrar los conceptos de restricciones de la integridad de la entidad o y restricciones de la integridad referencial (incluyendo la definición del concepto de llave foránea). 3. Demostrar el uso de las operaciones del álgebra relacional desde la teoría de conjuntos matemáticos (unión, intersección, diferencia y producto cartesiano) y las operaciones de álgebra relacional desarrolladas específicamente para bases de datos relacionales (select, product, join y división). 4. Demostrar consultas en el álgebra relacional. 5. Demostrar consultas en el cálculo relacional de tuplas.

2 Lenguajes de Consultas de Base de Datos.

Tópicos. I: Síntesis de los lenguajes de bases de datos. II: SQL (definición de datos, formulación de consultas, actualización del sublenguaje, restricciones e integridad).

Objetivos. 1. Crear un esquema de base de datos relacional en SQL que incorpora restricciones de integridad referencial, integridad-entidad, clave. 2. Demostrar la definición de datos en SQL y recuperar información de una base de datos usando la sentencia SQL SELECT. 3. Evaluar un conjunto de estrategias de procesamiento de consultas y seleccionar la óptima.. 4. Crear una consulta no procedimental por medio de llenado de plantillas de relaciones para construir un ejemplo del resultado de la consulta deseada. 5. Incrustar consultas orientadas a objetos en un lenguaje independiente tales como C++ o Java (ejemplo, SELECT Col. método() FROM Objeto).

3 Diseño de Bases de Datos Relacionales.

Tópicos. I: Diseño base de datos. II: Dependencia Funcional. III: Formas normales (1NF, 2NF, 3NF, BCNF). IV: Dependencia multivaluada(4NF). V: Dependencia de grupos (PJNF; 5NF). VI: Representación teórica.

Objetivos. 1. Determinar la dependencia Funcional entre dos o más atributos que son un subconjunto a de una relación. 2. Describir que significa 1NF, 2NF, 3NF y BCNF. 3. Identificar si una relación es una 1NF, 2NF, 3NF o BCNF. 4. Normalizar una 1NF en un conjunto de relaciones en 3NF (o BCNF) y desnormalizar un esquema relacional. 5. Explicar el impacto de la normalización sobre la eficiencia de las operaciones de base de datos, especialmente la utilización de consultas. 6. Describir que es una dependencia multivaluada y qué tipo de restricciones ésta especifica 7. Explicar por qué 4NF es útil en el diseño del esquema.

4 Procesamiento de Transacciones.

Tópicos. I: Transacciones.

Objetivos. 1. Crear una transacción mediante la incrustación de SQL en un programa de aplicación. 2. Explicar el concepto de compromiso implícito. 3. Describir los temas específicos para la ejecución de transacciones eficientes. 4. Explicar cuándo y por qué el rollback y como el logging asegura un rollback apropiado. 5. Explicar los efectos de los diferentes niveles de aislamiento sobre los mecanismos de control de concurrencia.

5 Bases de Datos Distribuidas.

Tópicos. I: Almacenamiento de datos distribuido. II: Procesamiento de consultas distribuidas. III: Modelo de transacción distribuido. IV: Control de concurrencia. V: Soluciones heterogéneas y homogéneas. VI: Cliente-servidor.

Objetivos. 1. Explicar las técnicas usadas para la fragmentación, replicación, alocaión de datos durante el proceso de diseño de bases de datos distribuidas. 2. Evaluar estrategias simples para ejecutar una consulta distribuida para seleccionar la estrategia que minimice la cantidad de transferencia de datos. 3. Explicar cómo el protocolo de compromiso en dos fases es usado o para tratar una transacción que accese a una base de datos almacenada en múltiples nodos. 4. Describir el control de concurrencia distribuido basado en la distinción de

técnicas de copiado y el método de elección. 5. Describir los tres niveles de software en el modelo cliente-servidor.

BIBLIOGRAFIA

- Bernstein, P. A. and Newcomer, E. (1997). Principles of Transaction Processing, First Edition. Morgan Kaufmann.
- Celko, J. (2005). Joe Celko's SQL Programming Style. Elsevier.
- Date, C. (2005). Data Mining: Practical Machine Learning Tools and Techniques, Second Edition. Elsevier.
- Dietrich, S. W. (2001). Understanding Relational Database Query Languages, First Edition. Prentice Hall.
- Elmasri, R. and Navathe, S. B. (2004). Fundamentals of Database Systems, Fourth Edition. Addison Wesley.
- Harrington, J. L. (2002). Relational Database Design Clearly Explained, Second Edition. Morgan Kaufmann.
- Ozsu, M. T. and Valduriez, P. (1999). Principles of Distributed Database Systems, Second Edition. Prentice Hall.
- Whitehorn, M. and Marklyn, B. (2001). Inside Relational Databases, Second Edition. Springer.

UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACION

SÍLABO

DESCRIPCION DEL CURSO

CURSO	Formación de Empresas de Base Tecnológica
CODIGO	CC081
CREDITOS	03 (tres)
PRE-REQUISITO	Ninguno
CARÁCTER	ELECTIVO
MODALIDAD	Semestral
HORAS	TEORIA (02) PRACTICA (02)

OBJETIVO DEL CURSO

- Elaborar un plan de negocio para dar inicio a una empresa de base tecnológica.
- Hacer un estudio que le permita detectar un nicho de mercado que haga viable la creación de la empresa.

SUMILLA

El curso esta dividido en: diagnóstico de mercado, plan tecnológico, marco legal aplicado a las TICs y análisis estratégico. Se busca aprovechar el potencial creativo e innovador en la creación de nuevas empresas.

CONTENIDO ANALITICO

CC081 Formación de Empresas de Base Tecnológica	
<ul style="list-style-type: none">▪ Proceso de investigación de mercados▪ Como solicitar estudios de mercado▪ La investigación como herramienta de análisis▪ La innovación y el ciclo de vida de la tecnología▪ La estrategia tecnológica▪ Propiedad Intelectual▪ Aspectos Tributarios▪ Aspectos Legales para la Constitución de la Empresa▪ Contratos Informáticos▪ Análisis Estratégico▪ Análisis de Entorno▪ Análisis Interno▪ Estrategia Competitiva▪ Visión y Misión de la Empresa	

BIBLIOGRAFIA

- Centro de Investigación y Desarrollo – PUCP, 2004. Plan de Negocio en Tecnologías de Información y Comunicación. Pontificia Universidad Católica del Perú.
- Congreso de la República del Perú (1996). Decreto Legislativo No823. Ley de la Propiedad Industrial. El Peruano.
- De la República del Perú, C. (1997). Ley No26887. Ley General de Sociedades. El Peruano.
- Ferre, J. (1997). Investigación de Mercados Estratégica. Gestión 2000
- Ludevid, M. (1994). Como crear su propia empresa: Factores Claves de Gestión. Marcombo Boxixareu.