Midterm Exam 2A Answer key

15110 Principles of Computing Fall 2014

November 10, 2014

Name: _____

Andrew ID: _____ Lab section: _____

Instructions

- Answer each question neatly in the space provided.
- There are 6 questions totaling 28 subproblems on 12 pages. Not all problems are the same size or difficulty.
- Please read each question carefully. You have 50 minutes for this exam. No electronic devices allowed. Good luck!

	Max	Score
1	6	
2	19	
3	21	
4	24	
5	22	
6	8	
Total:	100	

Questions

1. (6 points) This question deals with recursion and recursive algorithms.

   Consider the following Python function f, defined for nonnegative integers a and b recursively as follows:

```
def f(a, b) :
    if a == 0 :
        return b
    else :
        return f(a//2, b) + f(b//2, a)
```

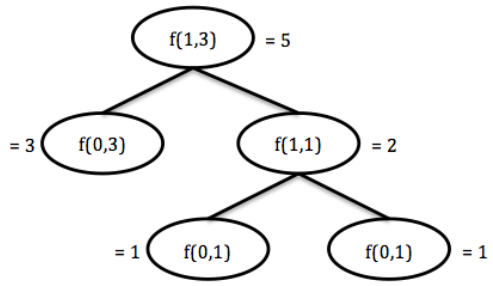
   Compute $f(1,3)$ by drawing a recursion tree showing all of the recursive calls and then use your tree to compute the final answer.

   $f(1,3) =$

Solution: 5

Recursion Tree:

Solution:



2. This problem focuses on the representation of data in a computer. The following table may be helpful:

2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1024	512	256	128	64	32	16	8	4	2	1

(a) (3 points) What is the largest number that can be represented as an unsigned integer using k bits?

Solution: $2^k - 1$,

(b) (3 points) Compute the decimal value of the byte 10100011 if it is interpreted as a signed 2s complement integer. Show your work.

Solution: -93.
Explanation: the sign bit is set, so it is negative. One's complement: 01011100, plus 1, gives 01011101. So the magnitude is $64 + 16 + 8 + 5 = 93$ or $5 \times 16 + 13 = 93$.

(c) (2 points) The ASCII character "!" is represented in binary using 7 bits as 010 0001. The character is to be sent over an unreliable communication link using even parity. What extra (leftmost) bit is sent along with this byte: 1 or 0?

Solution: 0

(d) (2 points) Suppose that the rightmost bit is corrupted during transmission of the eight bits from part (c) and is flipped from 1 to 0. Which of the following is true? Circle the appropriate letter:

Solution: B

A. The receiver cannot detect the error.
B. The receiver can detect the error but cannot determine which bit is wrong.
C. The receiver can detect the error and can correct the bit that is wrong.

(e) (2 points) Suppose we're sending ASCII codes with a parity bit, using even parity on an unreliable communication channel. With a reliability of 90%, our data source has a certain measurable entropy (information content). If the reliability of the communication channel goes up to 95%, will the entropy increase or decrease?

Solution: decrease

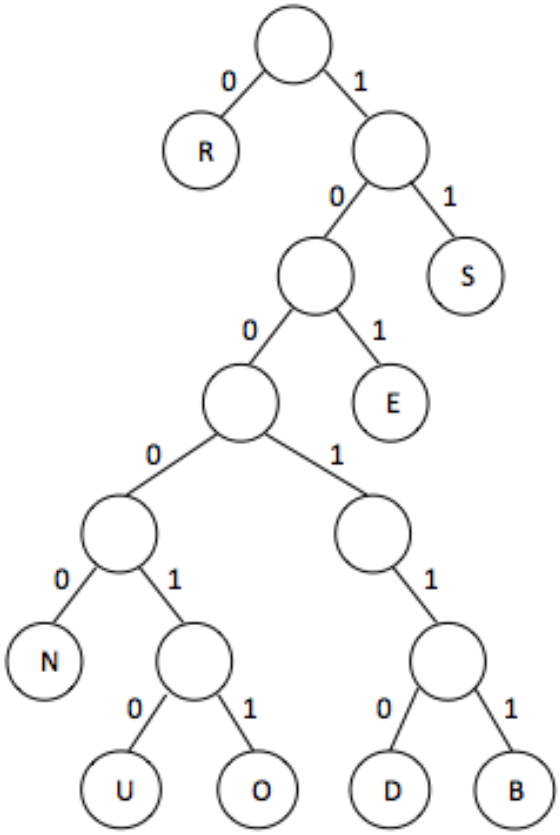
(f) (1 point) Is Huffman encoding of text files a lossless or lossy compression algorithm?

Solution: Huffman is lossless

(g) (1 point) Is MP3 encoding of audio files a lossless or lossy compression algorithm?

Solution: MP3 is lossy

(h) (5 points) Consider the following Huffman tree:



What word is represented by the following binary string, based on the Huffman tree (we have inserted spaces purely for readability): 111 000 111 000 101 000 010 011 010 10

Solution: SOUNDER

3. This question is concerned with generating and using pseudorandom numbers.

Recall that the Python function `randint(n, m)` returns a random integer between n and m, inclusive. Using the `randint` function, show how to compute the following using **one** Python expression:

(a) (3 points) A random integer between 1 and 28 that is a multiple of 5. Write one expression using `randint`.

Solution: `randint(1, 5) * 5` or something equivalent

(b) (3 points) A randomly chosen string from the list `names` below. You are not allowed to use variables other than `names` or any function other than `randint` or `len`.

```
names = ["Yuto", "Rachel", "Oliver", "Niki", "Matthew"]
```

Solution: `names[randint(0, len(names) - 1)]` or `names[randint(0, 4)]`

(c) (8 points) Write a function called `loaded_die()` that simulates a loaded die that rolls a one 40% of the time; six 20% of the time; and two, three, four, or five 10% of the time each. Your function should return an integer.

Solution:

```
def loaded_die() :
    r = randint(0,99)
    if r < 40 :
        return 1
    elif r < 60 :
        return 6
    elif r < 70 :
        return 2
    elif r < 80 :
        return 3
    elif r < 90 :
        return 4
    else:
        return 5
```

(d) (2 points) Describe in one or two sentences what method you could use to determine whether your code for part c behaves the way it should.

Solution: Use a Monte Carlo method: call roll() a large number of times and count the number of times each result is obtained.

Recall the linear congruential generator (LCG) formula:

$$x_{(i+1)} = (a \times x_i + c) \mod m \qquad \text{(for positive integers } a, c, \text{ and } m)$$

As described in class, there are certain conditions that should be obeyed for the LCG to have its maximum period:

1. c and m must be relatively prime;
2. $a - 1$ must be divisible by every prime factor of m; and
3. if m is divisible by 4, then $a - 1$ must be divisible by 4.

(e) (3 points) Consider *only* values of a that are less than m. If $c = 7$ and $m = 16$, what are the possible values of a that yield the maximum period? You may list them or describe them more succinctly.

Solution: a must be odd and $a - 1$ must be divisible by 4: 1, 5, 9, 13

(f) (2 points) What is the period obtained with $a = 1$ and $c = 7$ and $m = 16$?

Solution: The period is m, thus 16

4. This question deals with data organization.

(a) (3 points) Consider a function f that takes as input a list of words and returns a new list that contains the words in the reverse order. For example, given the list ["A", "*loves*", "B"], f returns ["B", "*loves*", "A"] by following the algorithm below:

1. Input the list *words*
2. Set *newlist* to [ ]
3. For each w in *words*
   - Store w in a **data structure**
4. For each w in the **data structure**
   - Get w from the **data structure**
   - Append to the end of *newlist*
5. Return *newlist*

Which data structure would be a more suitable choice? Stack or queue? Justify your answer with a brief explanation.

Solution: Stack because it is a FILO structure, which matches the description of the problem.

(b) (3 points) Suppose that Algorithm A stores a sequence of data elements using an array (the ith element in the sequence is found at index i) while Algorithm B stores the same sequence using a linked list (the ith element in the sequence is found at node i). Consider the operation of inserting an element in the first position in the sequence. Which algorithm would perform this operation more efficiently? Justify your answer with a brief explanation.

Solution: Algorithm B. Algorithm A would have to move all the elements to make room for the new element at the beginning. Algorithm B only needs to manipulate a few pointers.

(c) (3 points) Suppose that Algorithms A and B are two algorithms that can be used to search for a given word in an unsorted collection of n words. Algorithm A operates on a hash table with $n/3$ buckets where the words have been uniformly distributed by a hash function, and Algorithm B operates on a list of size n where each list position holds a single word. Which algorithm would be faster in searching for a word that does not exist in the collection? Justify your answer.

Solution: Algorithm A because it would have to look at 3 items in one bucket whereas B would have to look at all n items.

(d) (3 points) If the hash function used to distribute the words across the hash table were replaced with one that puts all the words into one bucket, how do Algorithms A and B compare, in terms of their performance when searching for a word that does not exist in the collection?

Solution: They would be the same because when all items fall into the same bucket the hash table degenerates into a list requiring the algorithm to inspect all the items as in the list case.

(e) (4 points) What does the following function return?

```
def mystery(wordlist):
    d = {}
    for w in wordlist:
        if w in d:
            d[w] = d[w] + 1
        else:
            d[w] = 1
    return d
```

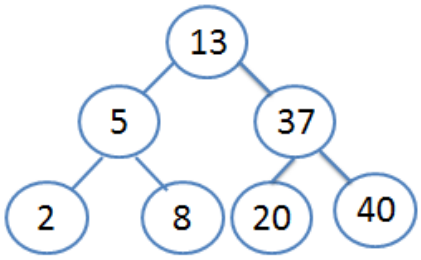
It returns a dictionary that maps

Solution: words in the word list

to

Solution: their frequencies in the word list.

(f) (6 points) One can construct several binary search trees where nodes contain integer keys drawn from the set {2, 5, 8, 13, 20, 37, 40} . Draw one that contains all the keys and has the **minimum** number of levels possible.



Solution:

(g) (2 points) In general for a binary tree with n nodes, what is the minimum number of levels the tree can have in terms of n?

Solution: $log_2(n) + 1$

5. This question deals with Boolean logic, gates and computer organization.

(a) (6 points) Draw the truth table for the Boolean expression $\neg A \wedge \neg B$, where A and B are Boolean variables. We started the table for you below. You should figure out how many rows you need.

A	B	$\neg A$	$\neg B$	$\neg A \wedge \neg B$

Solution:

A	B	$\neg A$	$\neg B$	$\neg A \wedge \neg B$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

(b) (3 points) The truth table you have drawn above can be implemented using a single gate with the inputs A and B. Which of the following gates is it? AND, OR, NOT, NAND, NOR, XOR.

Solution: NOR

(c) (5 points) Draw a combinational circuit that implements the Boolean expression $\neg(A \wedge B) \vee C$ using only OR and NOT gates. Note that AND and OR gates are 2-input gates. Hint: Recall De Morgan's Law: $\neg A \vee \neg B = \neg(A \wedge B)$

Solution: Transform to $(\neg A \vee \neg B) \vee C$ and draw the corresponding circuit.

(d) (4 points) Describe how a program is executed in a typical computer that conforms to the von Neumann architecture. Hint: Remember the name for the sequence of steps that are performed repeatedly and the purpose of each step.

Solution: Load the program counter to point to the first instruction in memory. Then, repeatedly

- **Fetch** the instruction indicated by the program countre from memory into the instruction register.
- **Decode** instruction to a control signal and get any data it needs (possibly from memory).
- **Execute** instruction with data in ALU and store results (possibly into memory, incrementing the program counter at the end

(e) (4 points) Circle the statements that are true.

A. A register is a small storage area in the CPU used to store intermediate data and instructions.

B. In the von Neumann architecture the memory unit holds data but not the instructions.

C. A machine language program consists of binary-coded instructions.

D. Gates are devices that are used to build transistors.

Solution: A and C

6. This question deals with simulation.

(a) (2 points) Describe the well-known problem we covered as an example for continuous simulation in class. Why is simulation a preferred method to tackle that problem as opposed to solving it analytically? Limit your answer to one sentence.

Solution: N-body problem: The problem of describing the motion of a collection of N bodies.

Mathematicians were able to prove that it would be impossible to derive simple solveable equations that would predict exactly where any of the N bodies would be at any arbitrary point in time in the general case. The simulation is based on the idea of estimating where a set of bodies will be by repeatedly calculating the gravitational attractions between pairs of bodies and computing their new positions a short time later. This type of operation is referred to as numeric integration.

(b) (6 points) Consider a random walk by a drunk person that begins in the middle of a square surrounded by walls. At each step of the simulation, he attempts to take a step north, south, west, or east, where each direction is equally likely. If he reaches the edge of the square and attempts to walk in the direction where the wall is, he hits his head but cannot change his position. We will assume that the layout of the square is a grid of tiles with 15 rows (row 0, row 1 etc.) and 15 columns (column 0, column 1 etc), and the drunk person initially stands at row 7 and column 7. Row 0 is the northernmost row, row 14 the southernmost row, column 0 the westernmost column, and column 14 the easternmost column. Every successful step changes the row or the column number, but not both. Every unsuccesful step leaves the position unchanged. For example, if our hero is in row 0 and attempts to go north he hits his head but remains where he is. Below is a Python function that simulates the walk for n steps, and returns the number of the times the drunk person has hit his head. Fill in the blanks.

```
from random import randrange
def walk(n):
    NORTH = 0
    SOUTH = 1
    EAST = 2
    WEST = 3
    row = 7
    col = 7
    hit = 0
    for steps in range(0,n):
        direction = ___________________
        if direction == NORTH:
            if row > 0:
                row = row - 1
            else:
                hit = hit + 1
        if direction == SOUTH:
            if _________________________:
                row = row + 1
            else:
                hit = hit + 1
        if direction == WEST:
            if _________________________:
               col = ______________________
            else:
               hit = hit + 1
        if direction == EAST:
            if ____________________________:
                col = ________________________
            else:
                hit = hit + 1
    return hit
```

Solution:

```
from random import randrange
def walk(n):
    NORTH = 0
    SOUTH = 1
    EAST = 2
    WEST = 3
    row = 7
    col = 7
    hit = 0
    for steps in range(1,n):
        direction = randrange(4)
        if direction == NORTH:
           if row > 0:
               row = row - 1
           else:
               hit = hit + 1
        if direction == SOUTH:
            if row < 14:
                row = row + 1
            else:
                hit = hit + 1
        if direction == WEST:
           if col > 0:
               col = col - 1
           else:
               hit = hit + 1
        if direction == EAST:
            if col < 14:
                col = col + 1
            else:
                hit = hit + 1
    return hit
```