CST 370 Design and Analysis of Algorithms
Summer A 2020
Midterm

Name: _______Andrew Bell____________________

Four-digits ID: ___1138________________________________

- Test time is **2 hours and 30 minutes**.
- Note that there are **13 problems in the midterm exam**.
- This is a **closed book exam**. You **can't use a calculator** during the exam. However, as a reference during the exam, you can prepare "two pages (= total of 4 sides)" **cheat sheet**. The cheat sheet can be typed or hand-written.
- **Blank paper can be used for your calculation** when solving problems.
- If possible, enter your answers directly into the Word file to increase the readability of your answers. However, if it is too difficult or time consuming to type in a Word file, write down your answer on paper. Then, take a picture and insert the picture into the Word file.
- For the exam, you must join **Zoom** (https://csumb.zoom.us/j/4457737239) and **turn on the video**. We will record the video. However, **turn off the audio** on your computer.
- **If you have a question** during the exam, **use "Chat"** in Zoom. I will answer to your question using "Chat".
- When you finish the exam, submit your PDF file (and optional Word file) on the iLearn. But keep the file(s) well in case we need it.
- Use your time wisely—make sure to answer the questions you know first.
- Read the questions carefully.

1. (2 points) Consider the following algorithm.

```
Algorithm DoSomething (A[0.. n – 1])
1.  num1 ← A[0];
2.  num2 ← A[0]
3. i ← 1
4.  while (i < n) do
5.     if A[i] < num1
6.          num1 ← A[i];
7.     if A[i] > num2
8.          num2 ← A[i];
9.     i ← i + 1
10.  return (num2 – num1);
```

(a) Present the **basic operation** of the algorithm. In your answer, you should **also present the line number** of the basic operation clearly.

The < on line 4

(b) Present the **time complexity** of the algorithm using the Θ **(= theta) notation**. If you can't represent it using Θ notation, indicate it clearly.

Θ (n)

2. (2 points) (a) Assume that there is an **undirected graph G with five vertices**. Write the **maximum** possible number of edge(s) in the graph.

10

(b) Assume that there is a **directed graph G with five vertices**. Write the **minimum** possible number of edge(s) in the graph.

0

3. (2 points) (a) Assume that Dr. Byun assigned a programming project which requires the time complexity of O(n). If your program's basic operation runs (2*n*logn + 25) times, can you say that your program meets the project requirement? (Yes/**No**).
No

(b) Assume a general tree T that is represented by <V, E>. If the number of vertices is represented by |V|, what would be the maximum number of edges possible in this tree?
Max E = V-1

4. (3 points) Consider the following algorithm:

```
// In the algorithm, assume that there are two arrays named B[0..n–1]
// and C[0..n–1] in addition to the input array A[0..n-1].
Algorithm DoSomething (A[0..n–1])
    i ← 0
    while (i < n) do
        B[i] ← 0
        i ← i + 1
    i ← 0
    while (i < n–1) do
        j ← i + 1
        while (j < n) do
            if A[i] <= A[j] // 2 5
                B[j] ← B[j] + 1
            else
                B[i] ← B[i] + 1
            j ← j + 1
        i ← i + 1
    i ← 0
    while (i < n) do
        C[B[i]] ← A[i];
        i ← i + 1
    return;
```

o n + n^2 + n

(a) Present the time complexity of the algorithm using the **O (= Big Oh)** notation. If you can't represent it using the **O** notation, indicate it clearly.

O(n^2)

(b) Assume that the input array **A** has the values **7, 2, 8, 2, 5**. Present the final results of the arrays **B[ ]** and **C[ ]**

B[3,0,4,1,2]
C[2,2,5,7,8]

5. (2 points) Based on the definitions of O, Θ, and Ω, determine whether the following assertions are true or false.

(a) $2n^3 + 5n + 1 \in O(n^2)$ (true/**false**) false

(b) $n * n * (n + 1) + 7 * n * n \in O(n^2)$ (true/**false**) false

(c) $2 * n * (n – 1) \in \Omega(n*\log n)$ (**true**/false) true

(d) $4 * (n * n) + 3 * n \in \Theta(n^2)$ (**true**/false) true

6. (2 points) (a) Assume that your program calculates the average of all numbers in an array. The time efficiencies of best case and worst case of the program are the same. (**true**/false).
True, always have to sum all of the numbers, unless we knew something ahead of time like that they were all the same number

(b) Assume that your program conducts the binary search for a value in a sorted array. The time efficiencies of best case and worst case of the program are the same. (true/**false**).
False, could find number in O(1) or O(logn)

7. (2 points) Consider the following recursive algorithm.

```
Algorithm Q(n)
if n = 1
    return 1
else
    return Q(n - 1) + 2 * n – 1
```

(a) Present the return value for Q(1).

1

(b) Present the return value for Q(2).

4

(c) Present the return value for Q(3).

9

(d) Based on the results of the above questions and other input number *n*, describe what this algorithm computes.

Computes n^2

8. (2 points) Consider the following master theorem:

$T(n) = aT(n/b) + f(n)$ where $f(n) \in \Theta(n^d)$, $d \geq 0$

Master Theorem: If $a < b^d$, $T(n) \in \Theta(n^d)$

If $a = b^d$, $T(n) \in \Theta(n^d \log n)$

If $a > b^d$, $T(n) \in \Theta(n^{\log_b a})$

Based on the theorem, select the correct time efficiency for each T(n). You have to **select and write your answer among 1, 2, 3, 4, and 5 clearly**.

(a) $T(n) = 2 * T(n/4) + 4n + 7$

1. $\Theta(n^2)$
2. $\Theta(n*\log n)$
3. $\Theta(n)$
4. $\Theta(n^{\log_4 2})$
5. None of the above.

Your answer: ______$T(n) \in \Theta(n^1)$ _____

a = 2 b = 4 d = 1

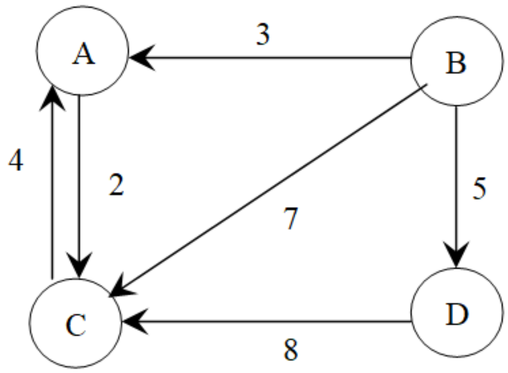
(b) $T(n) = 4 * T(n/2) + 3n^2 + 5n$

1. $\Theta(n^2)$
2. $\Theta(n*\log n)$
3. $\Theta(n)$
4. $\Theta(n^{\log_4 2})$
5. None of the above.

Your answer: _____$T(n) \in \Theta(n^2 \log n)$______

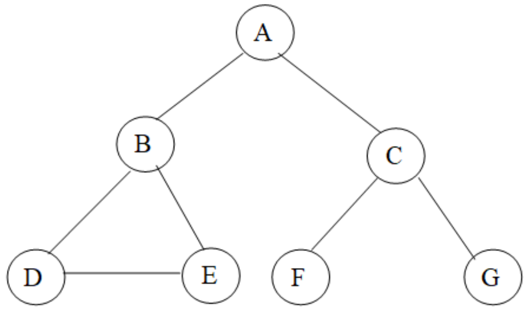
a = 4 b = 2 d = 2

9. (2 points) Represent the following weighted digraph in **an adjacency matrix** as we covered in the class. Fill out the matrix given below.



	A	B	C	D
A	**0**	inf	2	inf
B	**3**	0	7	5
C	**4**	inf	0	inf
D	**inf**	inf	8	0

10. (2 points) Consider the following **undirected graph**.



Assume that you traverse the graph by using **depth-first search** from **the vertex *a***. Fill the following **mark array** as you learned in the class. When you traverse the tree, you should follow our convention in the class (= alphabetical order).

A	**1**
B	**2**
C	**5**
D	**3**
E	**4**
F	**6**
G	**7**

11. (5 points) Solve the following recurrence relation using the **backward substitution** as we learned in the class. In the problem, you should present the **intermediate steps** clearly. Also, you should present **the time complexity** of the recurrence relation.

$M(n) = M(n - 1) + 2$ // recurrence relation

$M(0) = 5$ // initial condition

$M(n) = M(n – 1) +2$
$M(n – 2) + 4$
$M(n – 3) + 6$
$M(n – i) + 2i$

$O(n)$

12. (3 points) Suppose you have three jars, A, B, and C, in a room. Jar A has 5 large black balls, 4 large red balls, and 3 large green balls. Jar B has 5 small black balls, 4 small red balls, and 2 small green balls. Jar C is empty. Thus, there are **total 23 balls**. Now, you will pick a few balls from the jar A in the dark and place them in the jar C. After that, you will pick a few balls from the jar B in the dark and place them in the jar C. Note that the color of the selected balls at the jars A and B can not be confirmed because the surroundings are dark. Also, the numbers of balls selected from the jars A and B need not always be the same. Once you're done, you can turn on the lights in the room and see the balls in the jar C.

(a) Assuming the **worst case occurs**, what is the minimum number of balls you have to choose to **get a matching pair**? Here, a matching pair means that there must be one large ball and one small ball of the same color in the jar C. But the **color** itself of the pair **is not important**. **Present just the number of balls. You don't need to explain your answer.**

5 + 4 + 3 = 12 Balls

(b) Assuming the **best case occurs**, what is the minimum number of balls you have to choose to **get three matching pairs of each color (= black, red, green)**? In other words, you should have one pair of large and small black balls, one pair of large and small red balls, and one pair of large and small green balls. **Present just the number of balls. You don't need to explain your answer.**

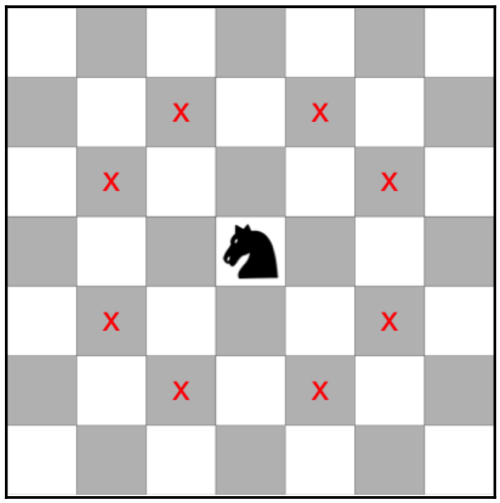
2 + 2 + 2 = 6

(c) Assuming the **worst case occurs**, what is the minimum number of balls you have to choose to **get three matching pairs of each color (= black, red, green)**? In other words, you should have one pair of large and small black balls, one pair of large and small red balls, and one pair of large and small green balls. **Present just the number of balls. You don't need to explain your answer.**

10 + 8 + 3 + 1 = 22

13. (2 points) [**Puzzle**] What is the minimum number of moves needed for **a chess knight** to go from one corner of a 100×100 board to the diagonally opposite corner? For this problem, **you don't need to explain your answer. Write just the minimum number.**

Note that the knight's moves are L-shaped: It can move two squares horizontally and one square vertically, or two squares vertically and one square horizontally. For example, let's assume that a chess knight is in the middle square of the following board. Then, it can move to 8 different squares as the diagram indicates.



35