CST 370 Design and Analysis of Algorithms
Spring A 2020
Final Exam

Name: ______Ivan Alejandre_____________________________

Four-digits Class ID: ______8889_________________________

- Test time is **2 hours and 30 minutes**.
- Note that there are **12 problems**
- 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.
- 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, please write down your answer on paper. Then, take a picture and insert the picture into the Word file.
- During the exam, you must sign into the Zoom session and turn on the video and turn off the audio. We will record the whole zoom session.
- If you have a question during the exam, please use "Chat" in Zoom. I will answer as soon as possible.
- When you finish the exam, submit your Word file (and an optional PDF file) on the iLearn. But keep the Word file well in case we need it.
- Use your time wisely—make sure to answer the questions you know first.
- Read the questions carefully.

1. (10 points) Choose one of the 4 options for each question.

(a) Let P be a QuickSort Program to sort numbers in ascending order using the first element as pivot. Let t1 and t2 be the number of comparisons made by P for the inputs {4, 1, 5, 3, 2} and {1, 2, 3, 4, 5} respectively. Which one of the following holds?

(A) t1 = 5
(B) t1 < t2
(C) t1 > t2
(D) t1 = t2

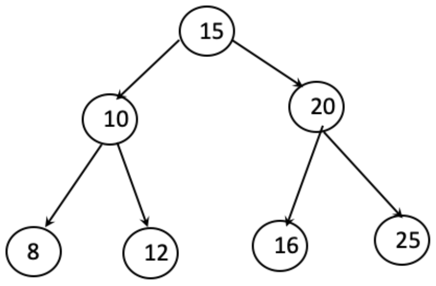
(b) Consider the following code snippet in C++. The function BSTk() receives root of a **Binary Search Tree (BST)** and a positive integer k as arguments. In the “node” data structure, “left” points to the left child and “right” points to the right child.

```
// A BST node
struct node {
    int data;
    struct node *left, *right;
};

int count = 0;

void BSTk(struct node *root, int k)
{
    if (root != NULL && count <= k)
    {
        BSTk(root->right, k);
        count++;
        if (count == k)
          printf("%d ", root->data);
       BSTk(root->left, k);
    }
}
```

What is the printed output of BSTk(root, 2) where root represents the root of the following BST?

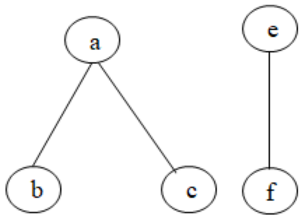


(A) 10
(B) 16

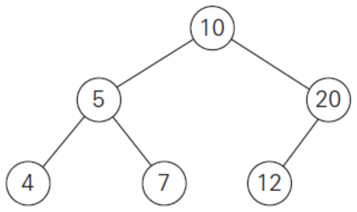
(C) 20
(D) 20 10

2. (10 points)

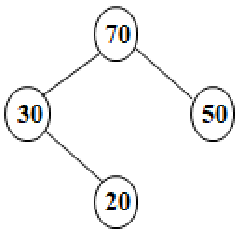
(a) Based on our class's definition, this is a graph. (**True**/False)



(b) This is an AVL tree. (**True**/False)



(c) This is a max heap. (**True**/False)



3. (5 point) 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 $a < b^d$, $T(n) \in \Theta(n^d)$ *Case 1*
If $a = b^d$, $T(n) \in \Theta(n^d \log n)$ *Case 2*
If $a > b^d$, $T(n) \in \Theta(n^{\log_b a})$ *Case 3*

Based on the theorem, determine the time efficiency of the following formula T(n).

(a) $T(n) = 2 * T(n/2) + n^4$
a=2, b=2, d=4. $\rightarrow$ 2 ? 2^4 $\rightarrow$ 2 < 16
case 1 applies. $\boldsymbol{T(n) \in \Theta(n^4)}$

(b) $T(n) = 16 * T(n/4) + n^2$
a=16, b=4, d=2 $\rightarrow$ 16 ? 4^2 $\rightarrow$ 16 = 16
case 2 applies $\boldsymbol{T(n) \in \Theta(n^2 \log n)}$

You need to indicate the values for a, b, and d. Also, indicate which case of the Master Theorem applies

4. (5 points) What is the time complexity of the following function fun()? Choose one of the 4 options and describe what the basic operation is here.

```
int fun(int n)
{
  int count = 0;
  for (int i = n; i > 0; i /= 2)
     for (int j = 0; j < i; j++)
        count += 1;
  return count;
}
```

(A) O(nLog(n))
(B) O(n)
(C) O(n^2)
(D) O(nLog(n)Log(n))

basic operation is the line "count += 1;" as it is executed each pass of the inner loop.

5. (10 points) Suppose you have three jars, A, B, and C, in a room. Jar A has 5 large black balls, 3 large red balls, and 2 large green balls. Jar B has 4 small black balls, 3 small red balls, and 2 small green balls. Jar C is empty. Thus, there are **total 19 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 the total number of balls chosen and how they are chosen (number of balls from each jar)

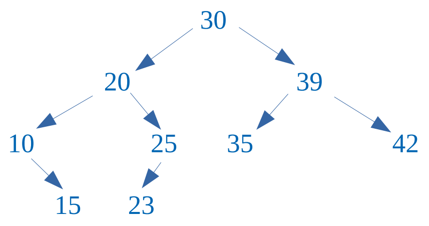
11 balls **total**. 10 balls from jar A and 1 ball from jar B.

(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 the total number of balls chosen and how they are chosen (number of balls from each jar)

6 balls **total** must be selected; 3 each from jars A and B. The best case is where from each jar, we pick the three colors as our 3 balls.

6. (5 points) The preorder traversal sequence of a binary search tree is 30, 20, 10, 15, 25, 23, 39, 35, 42. Which one of the following is the postorder traversal sequence of the same tree? Explain your answer.

a) 10, 20, 15, 23, 25, 35, 42, 39, 30
b) 15, 10, 25, 23, 20, 42, 35, 39, 30
c) 15, 20, 10, 23, 25, 42, 35, 39, 30
d) 15, 10, 23, 25, 20, 35, 42, 39, 30

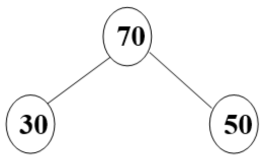


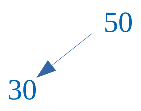
In preorder traversal, we display the root, then go to the left child, then the right child. Each child traversal is a recursive call so once we reach a leaf, we ascend the tree back to the first root where we went left without visiting the right.

In postorder traversal, we descend the tree recursively down to the leftmost child leaf and display its data. Then we ascend one node and visit its right child and display the leaf data. Only when both children of a node have been displayed do we display the parent node's data.

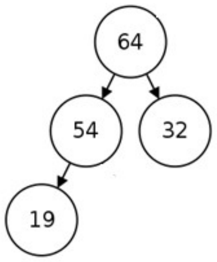
7. (10 points)

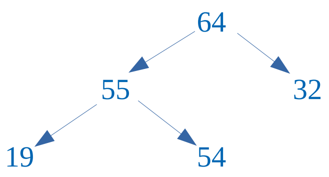
(a) Remove the max value from the following heap. Perform necessary operations to make it a heap again (heapify). Only need to present the final tree which should be a heap.





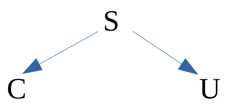
(b) Add 55 to the following heap. Perform necessary operations to make it a heap again (heapify). Only need to present the final tree which should be a heap.





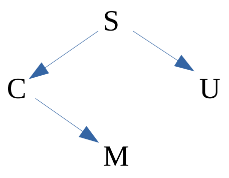
8. (5 points) Construct an AVL tree for the list **C, S, U, M, B, G, O**. Insert each letter successively starting with the empty tree. Use alphabetical order when inserting a letter. Your answer should present the rotation operations (if necessary) for each letter insertion.

Insert C, S, U:

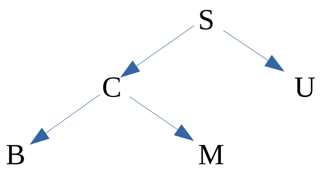


Involved one rotate L to arrive here.

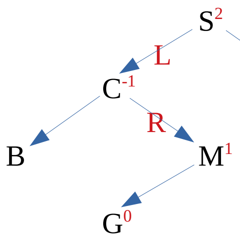
Insert M:



Insert B:

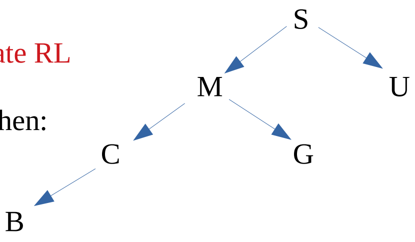


Insert G:

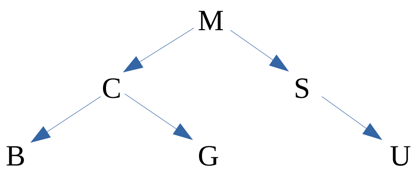


rotate RL

then:

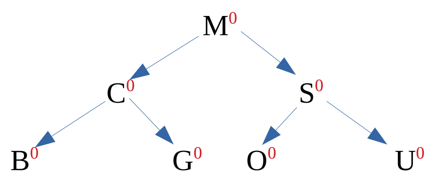


final:

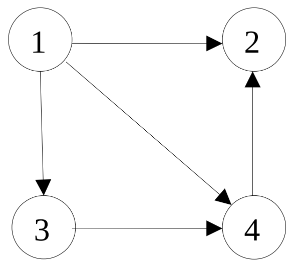


last step on next page

Insert O:



9. (10 points) Apply the Warshall's algorithm to get the transitive closure of the digraph defined by the following graph. Present $R^{(0)}$, $R^{(1)}$, $R^{(2)}$, $R^{(3)}$, and $R^{(4)}$ as we discussed in the class.



R(0)

	1	2	3	4
1	0	1	1	1
2	0	0	0	0
3	0	0	0	1
4	0	1	0	0

R(1)

	1	2	3	4
1	0	1	1	1
2	0	0	0	0
3	0	0	0	1
4	0	1	0	0

R(2)

	1	2	3	4
1	0	1	1	1
2	0	0	0	0
3	0	0	0	1
4	0	1	0	0

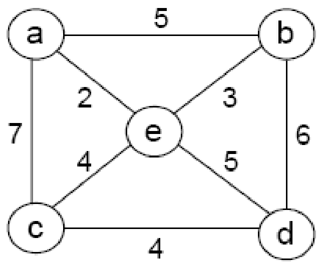
R(3)

	1	2	3	4
1	0	1	1	1
2	0	0	0	0
3	0	0	0	1
4	0	1	0	0

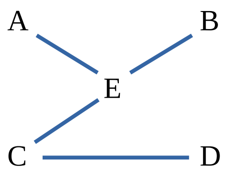
R(4)

	1	2	3	4
1	0	1	1	1
2	0	0	0	0
3	0	1	0	1
4	0	1	0	0

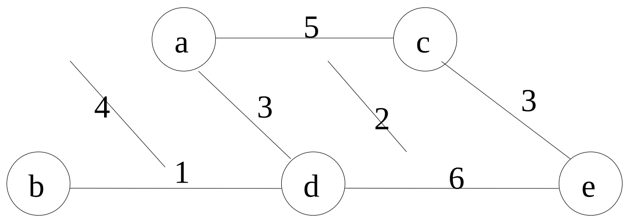
10. (10 points) Assume that you are going to solve the **MST** (Minimum Spanning Tree) problem using the **Prim's algorithm** for the following graph. Draw the **final MST**. For the problem, you have to start from the vertex **a**. You must also provide the **sequence of vertices** to be added to the "visited" set as we covered in the class.



Visited: A, E, B, C, D



11. (10 points) Assume that you are going to solve the single-source shortest-paths problem using the **Dijkstra's algorithm** for the following graph. For the problem, you should start from the vertex **a**. Fill out the table as you learned in the class.



V	a	b	c	d	e
a	0_a	4_a	5_a	3_a	∞
d		4_a	5_a	3_a	9_d
b		4_a	5_a		9_d
c			5_a		8_c
e					8_c

12. (10 points) Assume that you want to calculate 2^n where n is a positive integer using a **decrease-and-conquer** algorithm. For example, if the algorithm receives an input value 3 for n, it should return 8 (= 2^3).

(a) Describe the basic idea of your decrease-and-conquer algorithm **in English**.

One approach would to use the decrease by constant method. The premise is given $\mathbf{a^n = a * a^{n-1}}$, we can use recursion to find the solution to the smallest solution, then recursively call up to find the multiplication of all bases to power n.

(b) Based on the basic idea of (a), **write a pseudocode** of your algorithm

```
Algorithm DecreaseByConstant(n):
// finds 2^n
if n = 0 // only for initial case of n=0
        return 1
else if n = 1
        return 2
else
        return 2 * DecreaseByConstant(n-1)
```