95-771 Data Structures and Algorithms
Fall 2023

Midterm Exam
Carnegie Mellon University

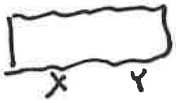
Name Key **Andrew ID** Key

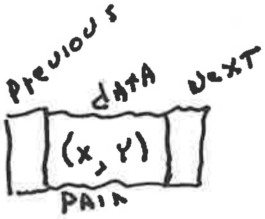
Total points = 133. Score will be a percentage of 133.

Tracing Lists and Trees (30 points)

1. You will be asked to show the exact output of the following program. (20 Points)

```
class Pair {
   private int x;
   private int y;
   public Pair(int x, int y) {
      this.x = x;
      this.y = y;
   }
   public void setX(int x) {
      this.x = x;
   }
   public void setY(int y) {
      this.y = y;
   }
   public int getX() {
      return x;
   }
   public int getY() {
      return y;
   }
   @Override
   public String toString() {
      return "x == " + x + " y == " + y + " ";
   }
}
class Node {
     private Node previous;
     private Pair data;
     private Node next;

     public Node (Node p, int x, int y, Node n) {
        previous = p;
        data = new Pair(x,y);
        next = n;
     }
    public Node getNext() {
       return next;
    }
    public Node getPrevious() {
       return previous;
```





```
    }
    public Pair getData() {
        return data;
    }
    public void setData(Pair data) {
        this.data = data;
    }
    public void setNext(Node next) {
        this.next = next;
    }
    public void setPrevious(Node previous) {
        this.previous = previous;
    }
    @Override
    public String toString() {
        return data.toString();
```

X == X   Y == Y

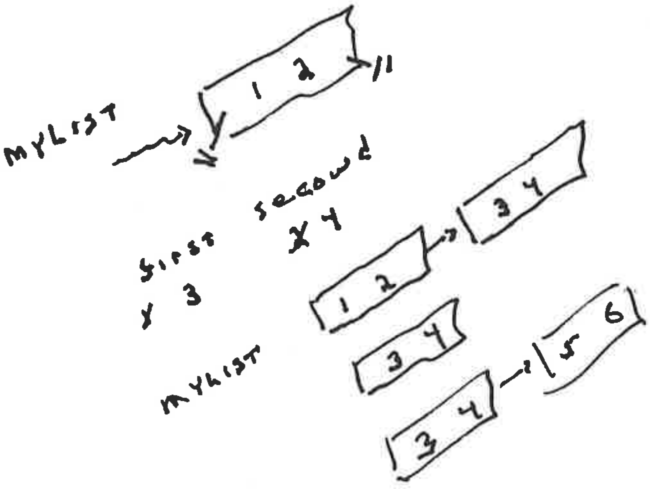
```
    }
}
public class List  {
    int first = 1;
    int second = 2;
    Node head = null;
```

first   second
1   2

```
    void add() {
        if (head == null) {
            head = new Node(null, first, second,null);
        }
        else {
            Node p = head;
            Node q = p;
            while (p != null) {
                q = p;
                p = p.getNext();
            }
            q.setNext(new Node(q,first, second,null));
        }
        first = second + 1;
        second = second + 2;
    }
    public String toString() {
        Node p = head;
        String temp = "";
        while (p != null) {
            temp = temp + p.toString() ;
            p = p.getNext();
        }
```

KEY

```
        return temp;
    }
    // In need of a pre-condition
    public void remove() {
        head = head.getNext();
        head.setPrevious(null);
    }

    public static void main(String[] args) {
        List myList = new List();
        myList.add();
        System.out.println(myList);   // Question 1.a
        myList.add();
        myList.remove();
        System.out.println(myList);   // Question 1.b
        myList.add();
        System.out.println(myList);   // Question 1.c
    }
}
```



1.(a) Show the output of the code marked (1.a). X ==1 Y==2 2 Points

1.(b) Show the output of the code marked (1.b). X ==3 Y==4 2 Points

1.(c) Show the output of the code marked (1.c). X==3 Y==4 X==5 Y==6 4 Points

1.(d) Is it correct to say that the method add() of the list class runs in O(n^2) ?
Circle [✓ YES] or NO (1 pt.)

1.(e) Is it correct to say that the method toString() of the list class runs in Ω(n) ?
Circle [✓ YES] or NO (1 pt.)

1.(f) Provide a pre-condition for the remove() method in the List class. Be as precise as possible. (4 Points)

// Precondition: head != Null or List not empty
public void remove{...}

1.(g) Write a recursive routine for the class List shown above. The variable head points to the first element of the list. This routine will print the list in reverse order. It is best to write two methods that, when used together, give the caller a convenient way to display the values on the list in reverse order. You must write valid Java and you may not use a stack. The pair data (x and y values) within a node are not reversed. The node that head points to will be printed last. (6 Points)

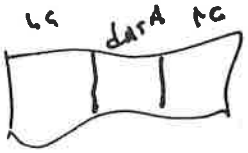
2. Study the execution of the following program. Note that it differs from those that we have studied. Questions appear below.

```
Private void Print_rec(Node h){
        if (h == Null) return
        else {
            Print_rec(h.getNext())
            System.out.println(h
        }
}
```

```
public void Print(List L Node h) {
        Print_rec(L.head);
}
```

KEY

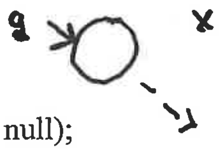
```
class Node {
   public int data;
   public Node lc;
   public Node rc;

   public Node(Node lc, int x, Node rc) {
      this.lc = lc;
      this.data = x;
      this.rc = rc;
   }
}
public class SimpleTree {

   public Node root;

   public SimpleTree() {
      root = null;
   }

   public void add(int x){
      if (root == null) {
         root = new Node(null,x,null);
      }
      else {
         Node t = root;
         Node q = t;
         while(t != null) {
            if(x < t.data ) {
               q = t;
               t = t.rc;   // This differs from what we studied
            }
            else {
               q = t;
               t = t.lc;   // Be sure to trace carefully
            }
         } // end while
         if(x < q.data) {
            q.rc = new Node(null, x, null);

         }
         else {
            q.lc = new Node(null, x, null);
         }
      }
   }
   public void traversal(Node r) {
```



root -//

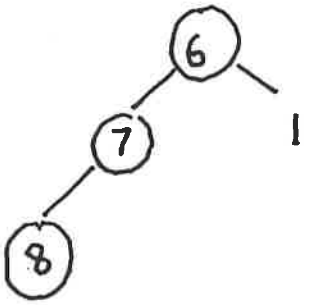
Less Than go right

else go left



```
     if(r == null) return;
     if(r.lc != null)traversal(r.lc);
     System.out.println(r.data);
     if(r.rc != null)traversal(r.rc);
  }
  public void traversal() {
     traversal(root);
  }

  public static void main(String[] args) {
     SimpleTree st = new SimpleTree();
     st.add(6);
     st.traversal();    // Question 2 (a)
     st.add(7);
     st.traversal();    // Question 2 (b)
     st.add(8);
     st.traversal();    // Question 2 (c)
     st.add(1);
     st.traversal();    // Question 2 (d)


  }
}
```

L, N, R



2.(a) What is the output from the traversal method labelled Question 2 (a)? 1 Point

6

2.(b) What is the output from the traversal method labelled Question 2 (b)? 1 Point

7 6

2.(c) What is the output from the traversal method labelled Question 2 (c)? 2 Points

8 7 6

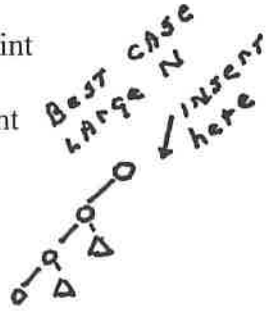
2.(d) What is the output from the traversal method labelled Question 2 (d)? 2 Points

8 7 6 1

2.(e) Does the traversal method have a worst case that differs from its best case? Circle YES or [✓ NO] 1 Point

2.(f) What is the worst case Big Theta value for the add method? $\Theta(N)$ 1 Point

2.(g) What is the best case Big Theta value for the add method? $\Theta(1)$ 2 Point

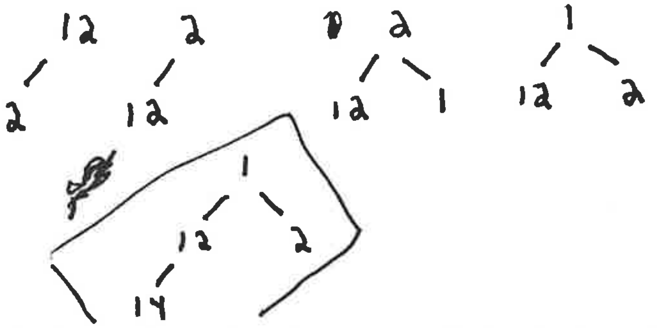


Heaps (12 points)

KEY

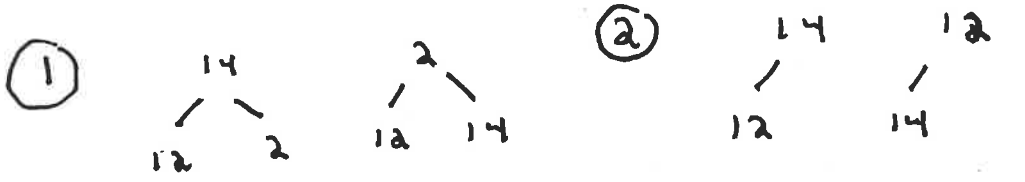
3) Insert the following 6 numbers into a min heap. Draw a new tree for each heap insertion. (4 Points)

12, 2, 1, 14



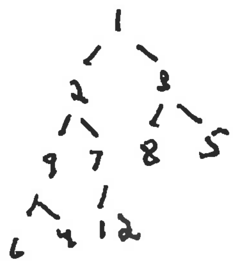
4) What is the height of the tree that you drew in question 3? (A single node in a tree gives a height of 0.)
(2 Points) 2

5) Perform exactly two deleteMin() operations on the heap that you drew in question 3. Draw the resulting trees. Make it clear to the reader what is going on. That is, show each step clearly. (3 Points)



6) Consider the following min heap implemented in an array. It is not quite correct. To make it a proper min heap exactly one swap must occur. What two numbers (child and parent) need to be swapped in order to make this a min heap? (3 points). PLACE CHECK MARKS NEXT TO THE TWO NUMBERS THAT NEED TO BE SWAPPED.

1
2
3
9 ✓
7
8
5
6
4 ✓
12



Key

12

Trees (16 points)

7. Parts (a), (b), (c) refer to the following binary tree:

```
          50
         /   \
      40       97
      / \        \
     6   5        30
    /     \       /.\
   7       3     2   16
```

(a) List the data that would be accessed by a pre-order traversal on the given tree by writing out the values in the nodes as they would be accessed, separated by commas. (3 points)

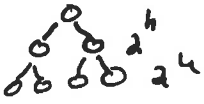
50, 40, 6, 7, 5, 3, 97, 30, 2, 16

(b) List the data that would be accessed by an in-order traversal on the given tree by writing out the values in the nodes as they would be accessed, separated by commas. (2 points)

7, 6, 40, 5, 3, 50, 97, 2, 30, 16

(c) List the data that would be accessed by a level-order traversal on the given tree by writing out the values in the nodes as they would be accessed, separated by commas. (2 points)

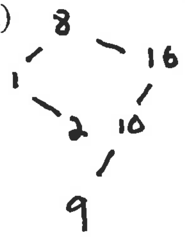
50, 40, 97, 6, 5, 30, 7, 3, 2, 16



(d) In general, if a binary tree (at most two children per node) is perfectly balanced (unlike the tree pictured above) and complete with height h, how many leaves will the tree have? (2 points) 2^h Note, this tree has a perfectly flat bottom. We need the total number of leaves in terms of h. This is an exact answer, not Big O.

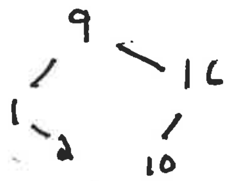
(e) In general, if a binary tree (at most two children per node) is perfectly balanced (unlike the tree pictured above) and complete with height h, how many internal nodes (non-leaves) will the tree have? (2 points) $2^h - 1$ Note, this tree has a perfectly flat bottom. We need the total number of internal nodes in terms of h. This is an exact answer, not Big O.

8. (a) Insert he following numbers into a Binary Search Tree. Draw the tree after all insertions are complete. (1 Point)
8, 1, 2, 16, 10, 9

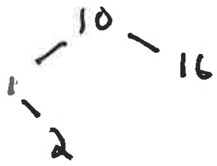


Key

(b) Delete 8 from the final tree that you drew in 8 (a). We are following the "go right once, left hard" rule. Draw this final tree. (2 Point)



(c) Delete 9 from the final tree that you drew in question 8 (b). Again, we are following the "go right once, left hard" rule. Draw this final tree. (2 Points)



Key

Project Questions (20 points)

(9) Recall the Merkle-Hellman cryptosystem that we worked with in Project 1.

Project 1 was based on the subset sum problem which is known to be NP-Complete. The problem itself can be described as follows: given a set of numbers *X* and a number **k**, is there a subset of *X*, which sums to **k**?

(a) Suppose X = {100, 9, 2, 105, 1, 7, 101} and **k** = 12. Is there a subset of X which sums to **k**?

__________ [✓ Yes]/No (1 point)

(b) The type of problem you were asked to solve in question 9 (a) is (Circle one answer): (1 Point)

1. a problem that is impossible to solve.
2. a problem that has been proven to take exponential time to solve.
3. a problem that has been proven to take factorial time to solve.
4. [✓ 4.] a decision problem.
5. an optimization problem.

(c) Suppose Alice sends the integer K to Bob. K is computed using Bob's Merkle-Hellman public key combined with the message M. The central idea behind Merkle-Hellman is that a potential eavesdropper could read the message M if the eavesdropper could (circle the one best option) (1 Point)

1. Modify the super increasing sequence.
2. Find K so that M is prime.
3. Modify Bob's public key.
4. [✓ 4.] Find a subset of Bob's public key that sums to K.
5. Find a subset of a super increasing sequence that sums to K.

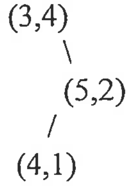
$5 \cdot 8 = 40$

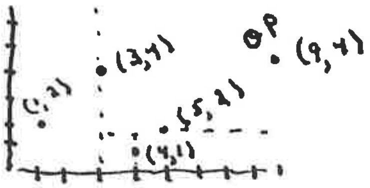
$40 \bmod 13 = 1$

$$\begin{array}{r} 3 \\ 13\overline{)40} \\ \underline{39} \\ 1 \end{array}$$

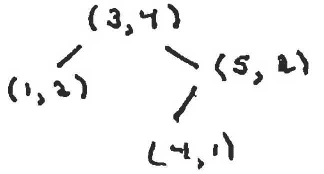
(d) Recall that a modular inverse of an integer b mod m is the integer b^{-1} such that ($b^* b^{-1}$) mod m=1. What is the modular inverse of 5 mod 13? __8__ (2 Points)

(e) The following points, in a standard (x,y) coordinate plane, have been added to a 2-d tree. (3,4), (5,2), ~~(4,6)~~ (4,1). The 2-d tree appears as follows:





Add the point (1,2) to this 2-d tree. Redraw the tree with this new point added. The first point, (3,4), breaks the plane vertically. (2 points)



KEY

(f) Consider the 2-d tree that you created, with the addition of (1,2), in (e). Suppose that we performed a nearest neighbor search for the point (9,4). Which points in the tree need to be examined? (3,4), (5,2) (4,1)
(2 Points)

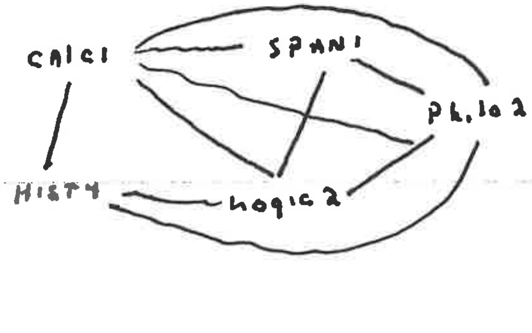
(g) In Project 3 we wrote a Red Black binary search tree. Suppose we are doing a lookup for a course name in the Red Black Tree. Let T(n) be the number of operations required to do the lookup. In the worst case, which of the following are true about T(n)? Circle all of those that are true. (You may or may not have more than one answer.) (4 Points)

1. [✓ T(n) ε O(n!)]
2. T(n) ε Ω(n)
3. T(n) ε Ω(n^2)
4. [✓ T(n) ε O(n)]
5. [✓ T(n) ε Θ(Logn)]
6. [✓ T(n) ε O(2^n)]
7. T(n) ε Θ(n)
8. [✓ T(n) ε O(Log(n))]
9. [✓ T(n) ε O(3^n)]
10. [✓ T(n) ε O($n^{1/2}$)]
11. T(n) ε Ω ($n^{1/2}$) ← Simple Power dominates Logarithmic

(h) The following is a data file for Project 3. Note the course Logic2 that is taken by Bill and Amy. The purpose of the Red-black tree was to associate an integer with each course name. What integer will be assigned to Bill's Hist4 in the Red-Black tree of Project 3? (4 Points) 4

	0	1	2	3
Amy	Calc1	Span1	Philo2	Logic2
Bill	Calc1	Philo2	Hist4	Logic2
	0	2	4	3

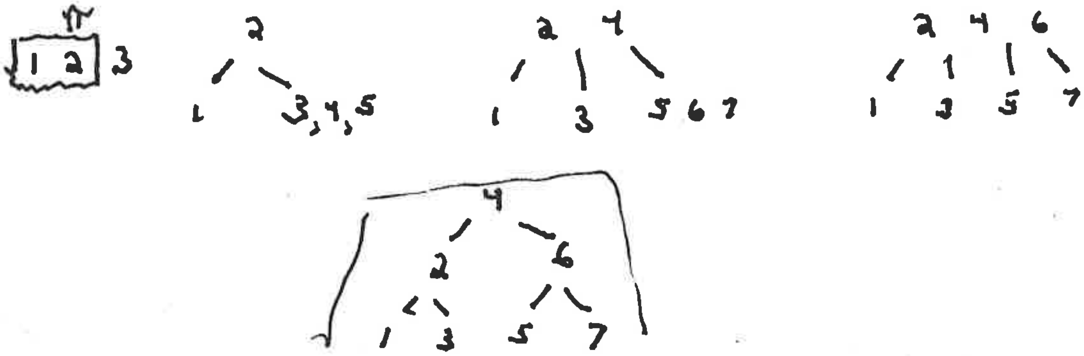
(i) Consider the graph generated by the Project 3 dataset in question 9 (h). We are referring to the previous question. How many edges will the node labelled "Logic2" have? (3 Points) 4



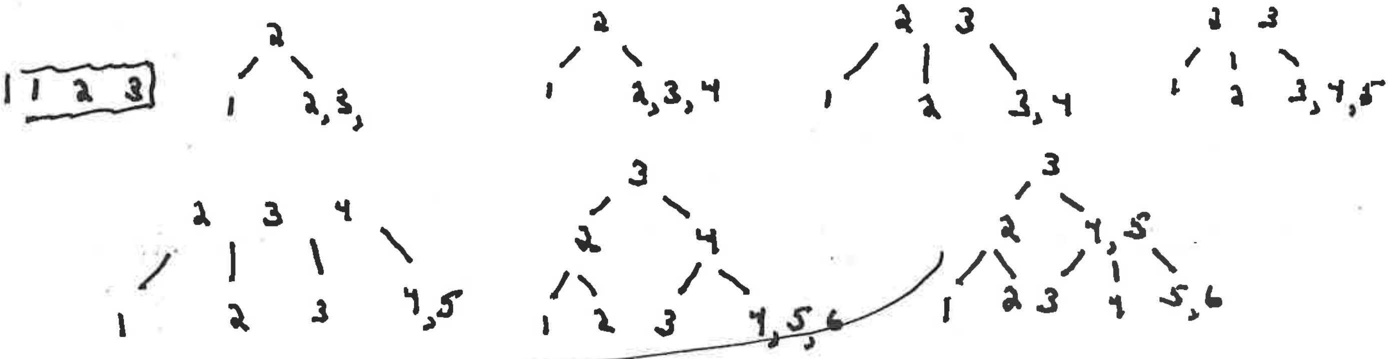
Key

B and B+ Trees (21 points)

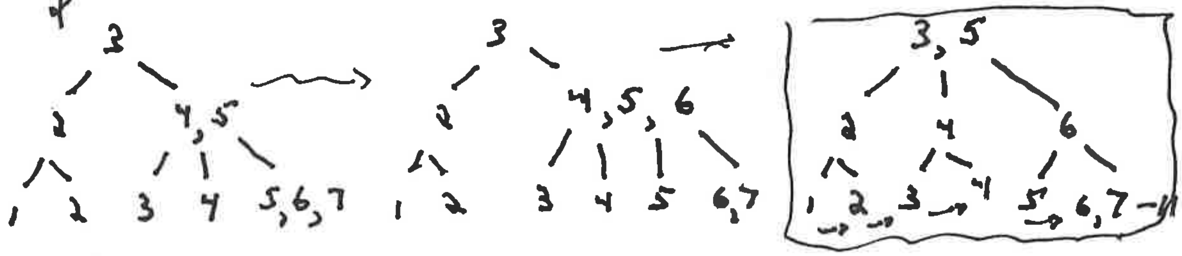
10. (a) Insert the following numbers into a B-Tree with a minimum of 1.
1, 2, 3, 4, 5, 6, 7 Draw the final tree. (7 Points)

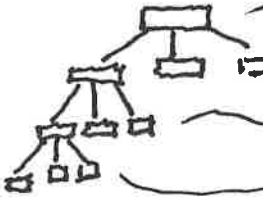


(b) Insert the following numbers into a B+ Tree with a minimum of 1.
1, 2, 3, 4, 5, 6, 7 Draw each tree for partial credit. Draw the final tree. (7 Points)
Note: This is a B+ tree and not a B-Tree.



(c) Consider a B-Tree with a minimum of 1. What is the exact maximum number of keys such a tree could hold if the tree were of height 3? 80
(7 Points)





2 = 2

3·2 = 6

3·3·2 = 18

3·3·3·2 = ~~18~~ 54

2 + 6 + 18 + 54 = 80

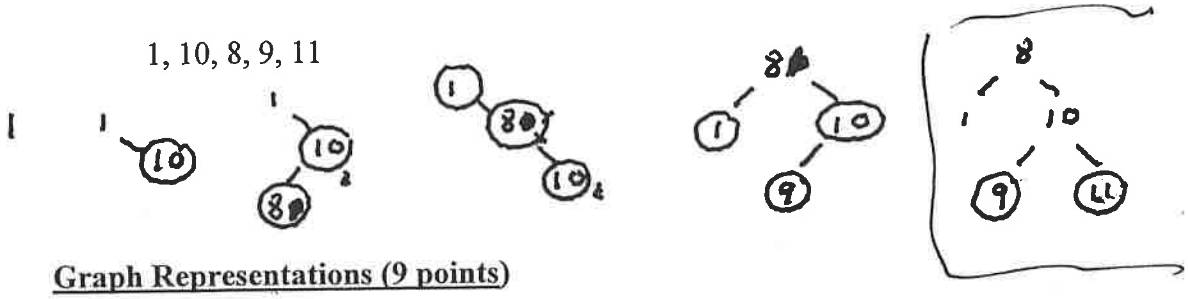
Key

Red Black Trees (8 points)

11. Red Black Trees

(a) Insert the following numbers, one by one, into a Red-Black Tree. Show the tree after each insertion. Draw RED nodes with a circle or a label 'R'. (8 points)

1, 10, 8, 9, 11



Graph Representations (9 points)

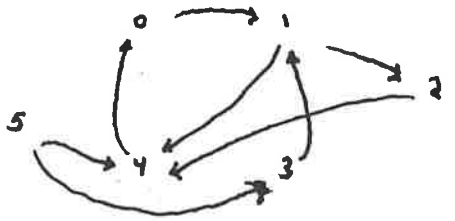
Consider a directed graph G₁. The graph is represented by an adjacency matrix m. If there is an edge from i to j then m[i,j] = true.

Matrix m

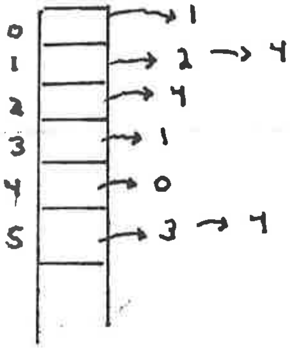
vertex	0	1	2	3	4	5
0		true				
1			true		true	
2					true	
3		true				
4	true					
5				true	true	

G₁

12. (a) Draw the graph G₁ with circles and edges. (2 Points)



12. (b) Suppose that we decide to represent the graph G₁ with an adjacency list rather than an adjacency matrix. Draw a picture of what that representation would look like.
(3 Points)



Key

12. (c) Given the graph G_1 and starting at vertex 3, what are the first 4 vertices visited by a breadth first traversal of the graph. (We are counting vertex 3 as one of the first 4). (3 Points). $3 \quad \underbrace{1 \quad 2}_{\text{ANY order}} \quad 4$

12. (d) Given the graph G_1 and starting at vertex 5, with a depth first traversal, we will visit all of the vertices in the graph. Circle [✓ TRUE] or FALSE.
(1 Point).

Graph Coloring (9 points)

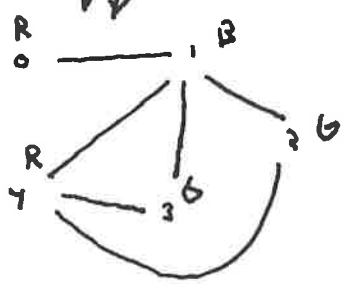
13. Color the graph G_2 with as few colors as possible. Use convenient colors - such as red, blue, yellow, green, cyan, etc. We need to see a drawing of the graph as well as the color of each node.
7 points

A matrix representation of the graph G_2 appears here:

vertex	0	1	2	3	4
0		true ✓			
1	true ✓		true ✓	true ✓	true ✓
2		true ✓			true ✓
3		true ✓			true
4		true ✓	true ✓	true ✓	

G_2

14. How many colors are required for an optimal coloring? 3 (2 Points)



Key

Big O (8 Points)

15. I have just arrived by plane at Pittsburgh International Airport and I have forgotten where I have parked my car. There are n cars in the parking lot. I decide to look for my car with a simple serial search, examining, one by one, each and every car in the lot. This algorithm could be analyzed by considering its best case, average case, and worst case. Circle each of the following answers that are mathematically correct. (4 Points)

a. In the worst case, the algorithm is Big Omega(n^2).
[✓ b.] In the best case, the algorithm is Big O(2^n).
[✓ c.] In the best case, the algorithm is Big O(n!).
d. In the best case, the algorithm is Big Theta(n).
[✓ e.] In the worst case, the algorithm is Big Omega(n).
f. In the worst case, the algorithm is Big Theta(n^2).

16. I have just arrived by plane at Pittsburgh International Airport and I have forgotten where I have parked my car. There are n cars in the parking lot. I have a new IoT gadget in my pocket. I click on the gadget and my car begins to beep loudly. I hear my car and walk directly to it. This algorithm (circle all that are correct). (4 Points)

[✓ a.] runs in Big O(n!).
b. runs in Big Omega(n!).
[✓ c.] runs in big O(n).
[✓ d.] runs in Big Theta(1).
[✓ e.] runs in Big Omega(1).
f. runs in Big Omega(n).
[✓ g.] runs in Big O(n^3).
[✓ h.] runs in Big O(1).

Key