21 Introduction to Data Structures
016

Midterm Exam
Carnegie Mellon University

Name key

Andrew ID key

Total points = 120. Score will be a percentage of 120.

## Coding Lists and Trees (30 points)

1. You will be asked to show the exact output of the following program. The program makes two calls on display(). Show the output from each call to display().

```
package dsmidterm;

class Node {
    private int data;
    private Node next;

    public int getData() {
      return data;
    }

    public Node getNext() {
      return next;
    }

    public void setData(int data) {
      this.data = data;
    }

    public void setNext(Node next) {
      this.next = next;
    }

    public Node(int data, Node next) {
      this.data = data;
      this.next = next;
    }
  }
class List {
    Node head;
    public List() {
      head = null;
    }
    public void add(int x) {
      head = new Node(x,head);
    }
    public int front() { return head.getData(); }
    public String toString() {
      Node v = head;
      String s = "";
      while(v != null) {
        s = s + v.getData() + " ";
        v = v.getNext();
      }
      return s;
    }
  }
```

38
12
16
20
21
8
15

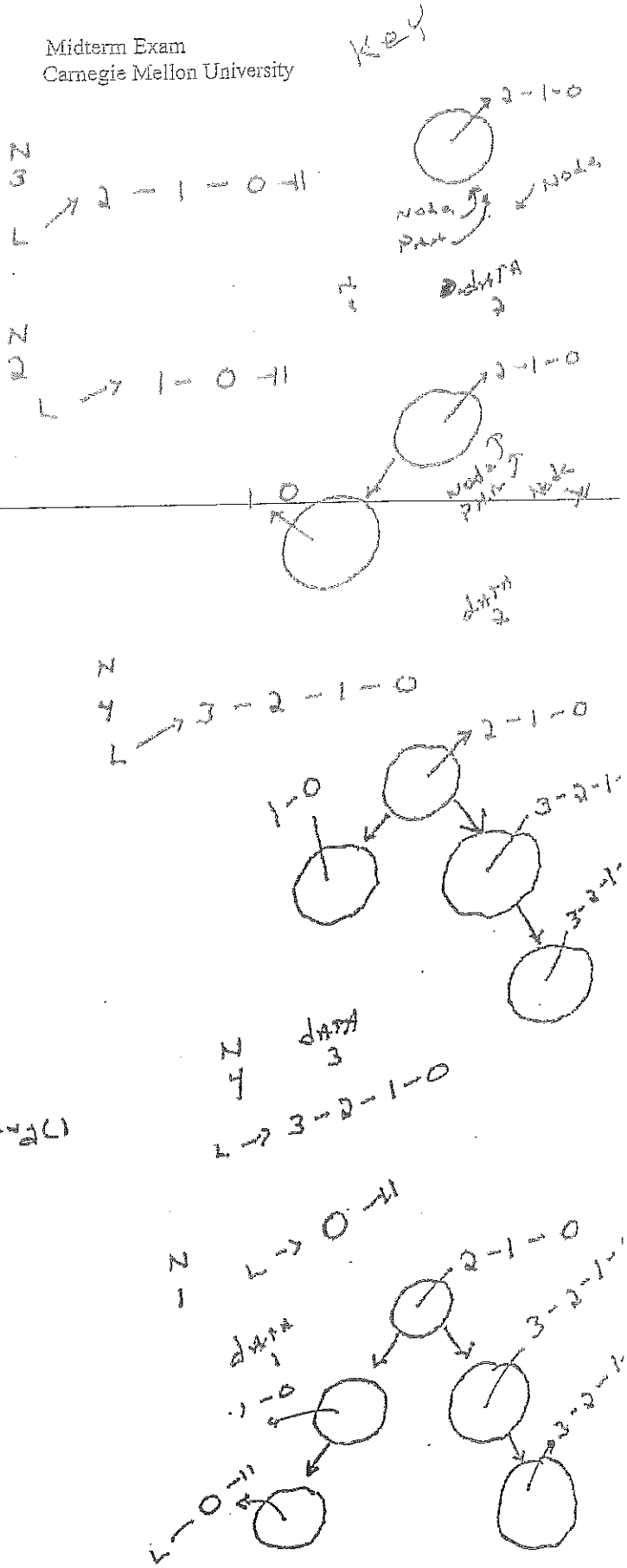
KEY

```
class TreeNode {
  List data;
  TreeNode lc,rc;

  public TreeNode(List data, TreeNode lc, TreeNode rc) {
    this.data = data;
    this.lc = lc;
    this.rc = rc;
  }

  public List getData() {
    return data;
  }

  public TreeNode getRc() {
    return rc;
  }

  public TreeNode getLc() {
    return lc;
  }

  public void setData(List data) {
    this.data = data;
  }

  public void setLc(TreeNode lc) {
    this.lc = lc;
  }

  public void setRc(TreeNode rc) {
    this.rc = rc;
  }

}
class BinarySearchTree {
  TreeNode root;

  BinarySearchTree()
  {
   root = null;
  }
```

KEY

```
public void addNode(int n)
{
  List l = new List();
  int j = 0;
  while(j < n) {
     l.add(j);
     j = j + 1;
  }
  if(root == null)
  {
     root = new TreeNode(l,null,null);
  }
  else
  {
     TreeNode node = root;
     TreeNode parent = null;
     while(node!=null)
     {
        parent = node;
        if(node.getData().front() > n)
           node = node.getLc();
        else
           node = node.getRc();
     }
     int data = parent.getData().front();
     if(data < n)
     {
        parent.setRc(new TreeNode(l,null,null));
     }
     else
     {
        parent.setLc(new TreeNode(l,null,null));
     }
  }
}
public void display(TreeNode r) {
  if(r != null) {
     display(r.getLc());
     System.out.println(r.getData());
     display(r.getRc());
  }
}
public void display() {
  display(root);
}
}
```

A List has a toString()



```
public class DSMidterm {

   public static void main(String[] args) {
      BinarySearchTree bst = new BinarySearchTree();
      bst.addNode(3);
      bst.display();    // answer question 1 a.
      bst.addNode(2);
      bst.addNode(4);
      bst.addNode(4);
      bst.addNode(1);
      bst.display();    // answer question 1 b.

   }

}
```

1.a What will the program display at bst.display() marked 1.a? 2 1 0 (6 pts)

1.b What will the program display at bst.display() marked 1.b? (10 pts)

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

1.c Is it correct to say that the toString() method of the list class runs in $O(2^n)$ ? Circle True or False (2 pts.)

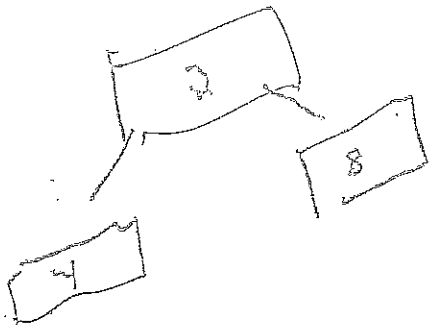
$T(N) = C \cdot N$ IF STRING OPERATIONS ARE CONSTANT TIME.

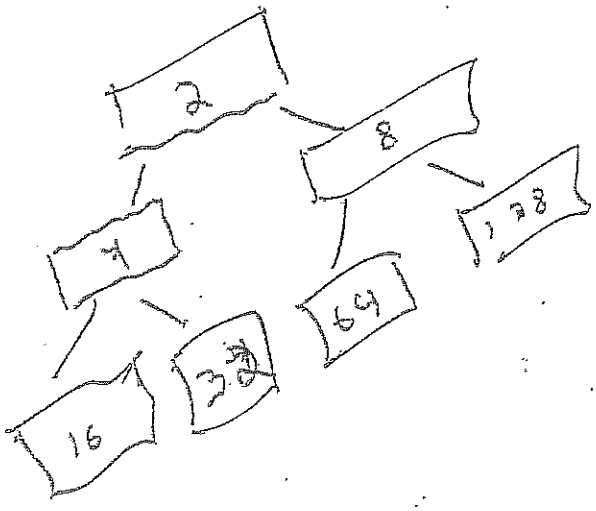
$T(N) \leq K \cdot 2^N$

2. Study the execution of the following program. Questions appear below. (10 points):

```
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 int ctr;

   public SimpleTree() {
      root = null;
      ctr = 1;
   }
```

```
20

  private Node add(Node t){

    if (t == null) {
      ctr = ctr * 2;   // NOTE: Counter is doubled
      return new Node(null,ctr,null);
    }
    t.lc = add(t.lc);
    t.rc = add(t.rc);
    return t;
  }
  public void add() {
    if (root == null) {
      ctr = ctr * 2 ;     // NOTE: Counter is doubled
      root = new Node(null,ctr,null);
    }
    else {
      add(root);
    }
  }
  public void traversal(Node t) {

    if(t != null) {
      traversal(t.rc);     // NOTE: Right child first
      traversal(t.lc);     // NOTE Left child second
      System.out.print(t.data + ":");
    }
  }
  public void traversal() {
    traversal(root);
  }

  public static void main(String[] args) {
    SimpleTree st = new SimpleTree();
    st.add();
    st.add();

    // Question 2.a
    st.traversal();
    System.out.println();

    SimpleTree coolTree = new SimpleTree();
    coolTree.add();
    coolTree.add();
    coolTree.add();

    // Question 2.b
    coolTree.traversal();

  }
}
```





2.a What will the program display at the traversal marked Question 2.a? (5 Points)

8 4 2

2.b What will the program display at the traversal marked Question 2.b? (5 Points)

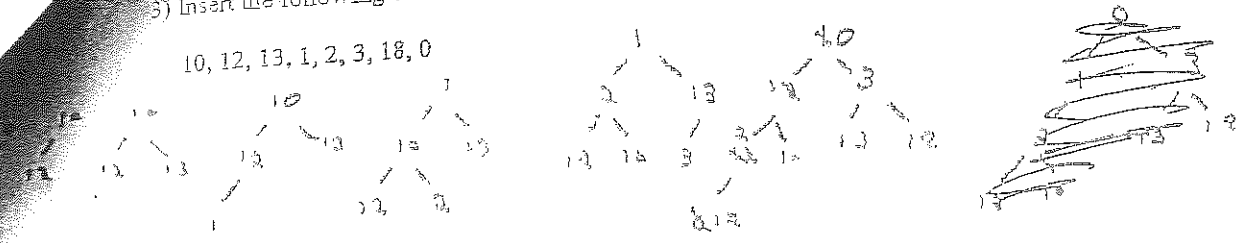
128 64 8 32 16 4 2

Key

## Heaps (12 points)

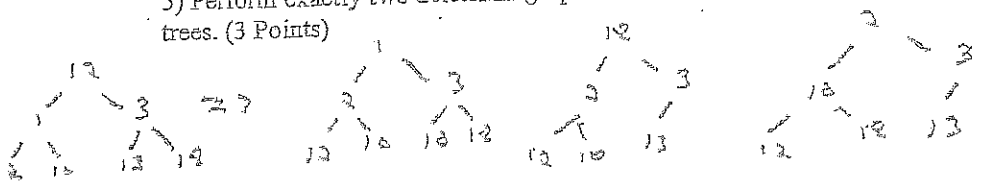
3) Insert the following 8 numbers into a min heap. Draw a new tree for each heap insertion. (4 Points)

10, 12, 13, 1, 2, 3, 18, 0



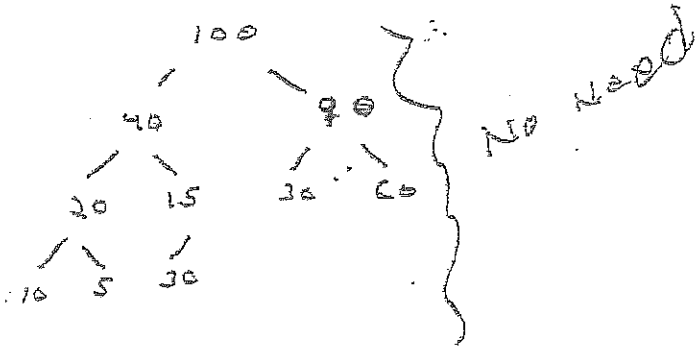
4) What is the height of the tree that you drew in question 3? (2 Points) 3

5) Perform exactly two deleteMin() operations on the heap that you drew in question 3. Draw the resulting trees. (3 Points)



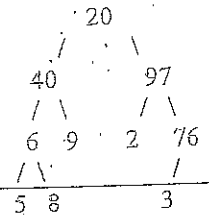
6) Consider the following max heap implemented in an array. It is not quite correct. To make it a proper max heap exactly one swap must occur. What two numbers (child and parent) need to be swapped in order to make this a max heap? (3 points). PLACE CHECK MARKS NEXT TO THE TWO NUMBERS THAT NEED TO BE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

100
40
90
20
15 ✓
30
60
10
5
30 ✓



Binary Trees (16 points)

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




(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)

20, 40, 6, 5, 8, 9, 97, 2, 76, 3

(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)

5, 6, 8, 40, 9, 20, 2, 97, 3, 76.

(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)

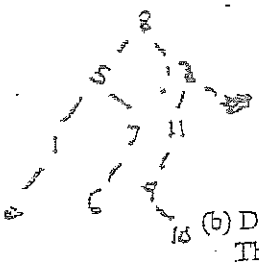
20, 40, 97, 6, 9, 2, 76, 5, 8, 3

(d) In general, if a ternary (at most three children per node) tree is perfectly balanced (unlike the tree pictured here) and complete with height h, how many leaves, in terms of h, will the tree have? (2 points) $3^h$ Note, this tree has a perfectly flat bottom.

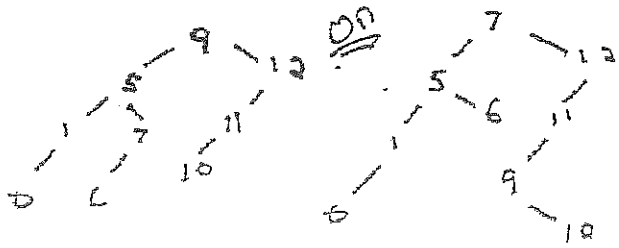
(e) In general, if a ternary tree is perfectly balanced (unlike the tree pictured here) and complete with exactly k leaves. What is the height (in terms of k) of this tree? (2 points) $\log_3 k$
Note, this tree has a perfectly flat bottom.

if 2, take 1pt.

8. (a) Insert he following numbers into a Binary Search Tree. Draw the tree after all insertions are complete. (3 Points)
8,5,12,1,7,11,0,6,9,10



(b) Delete 8 from the final tree that you drew in 8 (a). Draw this final tree. (2 Points)
There are two possible answers. Either one is fine.



Key

## Project Questions (20 points)

(9) (a) In Project 1, we wrote an RPN expression evaluator. Fill in the blanks by evaluating each expression. This is meant to show a single run of the calculator. (8 points):

```
1 2 - 4 5 - *
  1
n 3 4 + 3 * =
    21
22 n -
   1
```

(b) In Project 2 we wrote an implementation of Shunting Yard. Show the content of the stack and the output as Shunting Yard processes the following expression. Draw pictures so that it is clear to the reader that you understand the algorithm. (8 points).

3 + 2 * 4 ^ 2 ^ 1

3 2 4 2 1 ^ ^ * +

(c) In Project 3 we wrote a binary search tree containing lists of crime records. It was organized with the crime address being the key in each tree node.

Circle all of those that are true. (You may or may not have more than one answer.) (4 Points)

In the worse case, a lookup on an address in a tree with N nodes is:

1. O(LogN)
2. O(1)
3. $\Omega(N^2)$
4. O(N)
5. $\Theta$(LogN)
6. $O(2^N)$
7. $\Omega(1)$
8. $\Theta$(N)

$T(N) = c \cdot N$

-1 for each answer
MAX OFF = -4

## B Trees (21 points)

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

1 2 3 4 5

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

or

c) Consider a B-Tree with a minimum of 10. What is the exact maximum number of keys such a tree could hold if the tree were of height 1? 440 (7 Points)

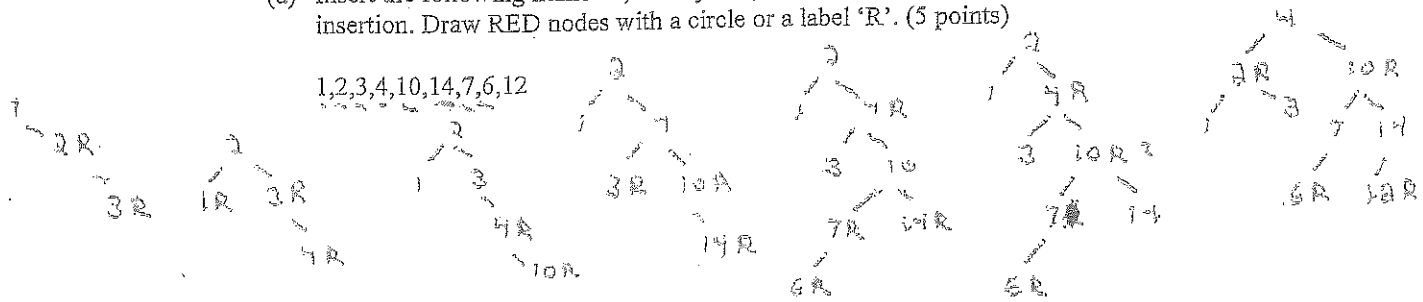
21
x 20
420
+ 20
440

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'. (5 points)

1,2,3,4,10,14,7,6,12



(b) When trying to analyze the run time complexity of an inorder traversal of a Red Black tree, one should consider the worst case, average case and best case separately. Each case may have a different run time performance. Circle TRUE or FALSE (1 point)

(c) What is the best-case runtime complexity of a Red Black Tree insert operation? Use Big Theta notation. (1 points) $\Theta(\log_2 N)$ any base

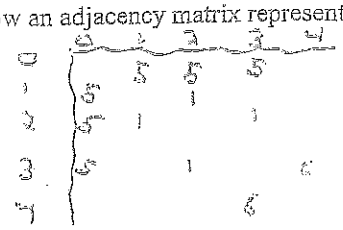
(d) What is the best-case runtime complexity of a Binary Search Tree insert operation? Use Big Theta notation. (1 points) $\Theta(1)$

Graphs Representations (15 points)

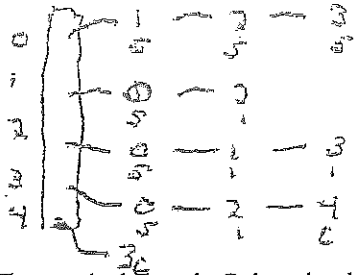
12. (a) Consider the weighted and undirected graph $G_2$ attached. When starting at vertex 0, breadth first search (BFS) could be used to enumerate (visit) each of the vertices. Show an ordering that BFS might produce. Starting from vertex 0, visit all the nodes. List the nodes as BFS would visit.
(2 points) 0, 1, 2, 3, 4 (Any ordering)

(b) If we want to learn if a path exists between vertex A and vertex B, we can simply run either BFS or DFS to find out. Circle True or False. (1 points)

(c) Show an adjacency matrix representation of the graph $G_2$ (3 points)



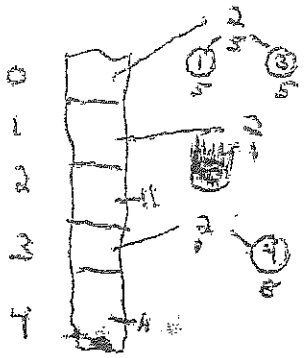
(d) Draw an adjacency list representation of the graph $G_2$. You need to think about how you will represent the distances associated with the edges. The picture that you draw will make it clear how the edge weights are being represented. (5 points)



(e) The graph shown in $G_2$ is a simple, undirected graph. It contains no loops or multiple edges. Suppose that such a graph has V vertices. What is the maximum number of edges that such a graph can contain (in terms of V)? Your answer should be an exact formula. Answers in Big Theta notation don't count. (2 point)

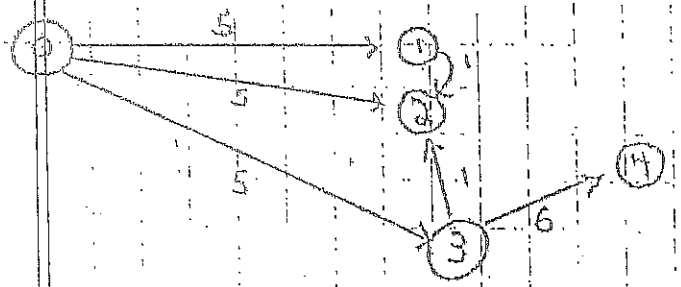
V(V-1)/2

(f) Draw an adjacency set representation of the graph $G_1$ attached. Use a Red Black tree as the set. Circle red nodes or mark them with an R. (2 Points)



Directed graph with edge weights.

$G_1$




Undirected graph with edge weights.

$G_2$