95-771 Data Structures and Algorithms
Fall 2015

Midterm Exam

Print Full Name Key

Coding Lists and Trees (18 points)

1. The following program contains four println() statements. Describe the output of the program when each of the println() statements is executed. (8 points):

```
package dsamidterm;


class Node {
  private int data;
  private Node next;
  public Node(int data, Node next) {
    this.data = data;
    this.next = next;
  }
  public int getData() {
    return data;
  }
  public void setData(int data) {
    this.data = data;
  }
  public Node getNext() {
    return next;
  }
  public void setNext(Node next) {
    this.next = next;
  }
}
public class DSAMidterm {
  public Node list = null;
  public void insert1(int x) {
    list = new Node(x,list);
  }
  public void insert2(int x) {
    if(list == null) insert1(x);
    else {
      Node m = list;
      Node p = list;
      while (m != null) {
        p = m;
        m = m.getNext();
      }
      p.setNext(new Node(x,null));
    }
  }
  public String toStringToo(Node n) {
    if (n == null) return "";
    else return toStringToo(n.getNext()) + n.getData();
  }
  public String toString() {
    return toStringToo(list);
  }
```

key

```
  int xyz(Node p) {
    if ( p == null) return 0;
    else return p.getData() + xyz(p.getNext());
  }
  int abc() {
    return xyz(list);
  }

  public static void main(String args[]) {
    DSAMidterm dsa = new DSAMidterm();
    for(int i = 1; i <= 8; i++) dsa.insert2(i);

    // Show output as 1.a
    System.out.println(dsa);

    // Show output as 1.b
    System.out.println(dsa.abc());

    dsa = new DSAMidterm();
    for(int i = 1; i <= 4; i++) dsa.insert1(i);

    // Show output as 1.c
    System.out.println(dsa);

    // Show output as 1.d
    System.out.println(dsa.abc());
  }
}
```

1.a What will the program display at the println marked 1.a? 8 7 6 5 4 3 2 1

1.b What will the program display at the println marked 1.b? 36

1.c What will the program display at the println marked 1.c? 1 2 3 4

1.d What will the program display at the println marked 1.d? 10

2. The following program contains a System.out.print() statement in the traversal method. (10 points):

```
package simpletree;
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 static int ctr = 1;
   public SimpleTree() {
      root = null;
   }
   private Node add(Node t){
      if (t == null) {
         ctr = ctr * 2 ;
         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 + 1;
         root = new Node(null,ctr,null);
      }
      else {
         add(root);
      }
   }
   public void traversal(Node t) {
      if(t != null) {
         traversal(t.lc);
         traversal(t.rc);
         System.out.print(t.data + " ");
      }
   }
   public void traversal() {
      traversal(root);
   }
```

Key

```
  public static void main(String[] args) {

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

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

     st.add();

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

  }
}
```

2.a What will the program display at the traversal marked Question 2.a?

4 8 2 ~~(4,8,2)~~

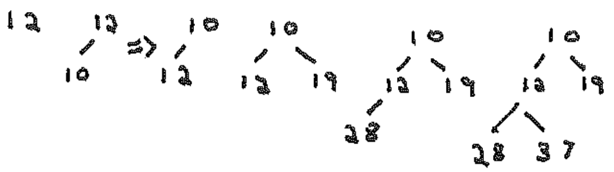
2.b What will the program display at the traversal marked Question 2.b?

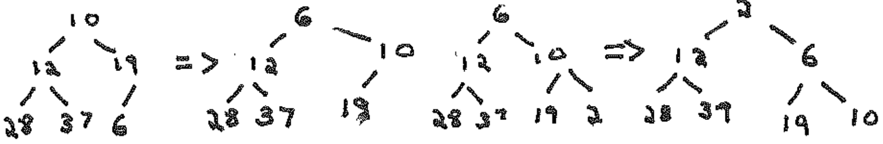
16 32 4 64 128 2 ~~(16 32 4 64 128 8 2)~~

Heaps (12 points)

3) Insert the following 7 numbers into a min heap. Draw a new tree for each heap insertion. (4 Points)

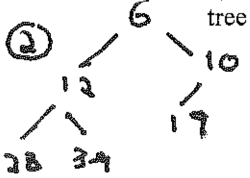
12, 10, 19, 28, 37, 6, 2

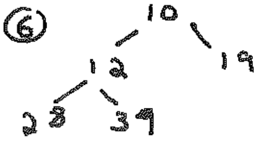




4) What is the height of the tree that you drew in question 3? (2 Points) 2

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





6) Suppose that you want to sort an array x of n integers. The only data structure that you have available is a MaxHeap. Show how you can sort the array of n integers making good use of the MaxHeap class. Use this Java code as a starting point. (3 points)

```
// pre: x is an array of n integers
// post: the array x is sorted in increasing order x[0] <= x[1] <= … <= x[n-1].

public void sort(int x[], int n) {

  MaxHeap h = new MaxHeap();
  // Your code goes here.
  for (int i = 0; i < n; i++) h.insert(x[i]);
  for (int i = n-1; i >= 0; i--) x[i]= h.deletemax();


}
```

Binary Trees (16 points)

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

```
                10
              /     \
            40          15
           / \           \
          6   8           76
         / \   \
        5   1   3
           /\
          6  9
```

(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)

10, 40, 6, 5, 1, 6, 9, 8, 3, 15, 76

(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, 6, 1, 9, 40, 8, 3, 10, 15, 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)

10, 40, 15, 6, 8, 76, 5, 1, 3, 6, 9

(d) In general, if a binary tree is perfectly balanced (unlike the tree pictured here) and complete with height 10, how many leaves, will the tree have? (1 point) $2^{10} = 1024$ Note, all of the leaves of this tree are at the same depth. How many internal nodes would such a tree have? (1 Point)

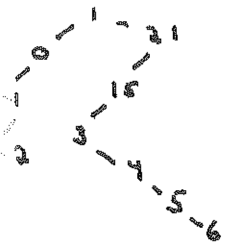
$2^{10} - 1 = 1023$

(e) In general, if a binary tree is perfectly balanced (unlike the tree pictured here) and complete with exactly 64 leaves. What is the height of this tree? (1 point) 6 Note, all of the

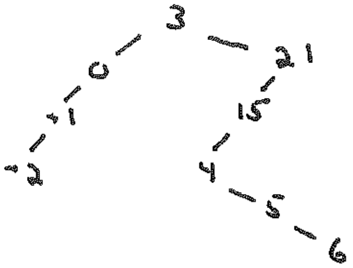
Key

leaves of this tree are at the same depth. How many internal nodes will the tree have? (1 point) $63 = 2^6 - 1$

8. (a) Insert he following numbers into a Binary Search Tree. Draw the tree after all insertions are complete. You need not show each step. (2 Points)
1,21,15,3,4,5,6,0,-1,-2

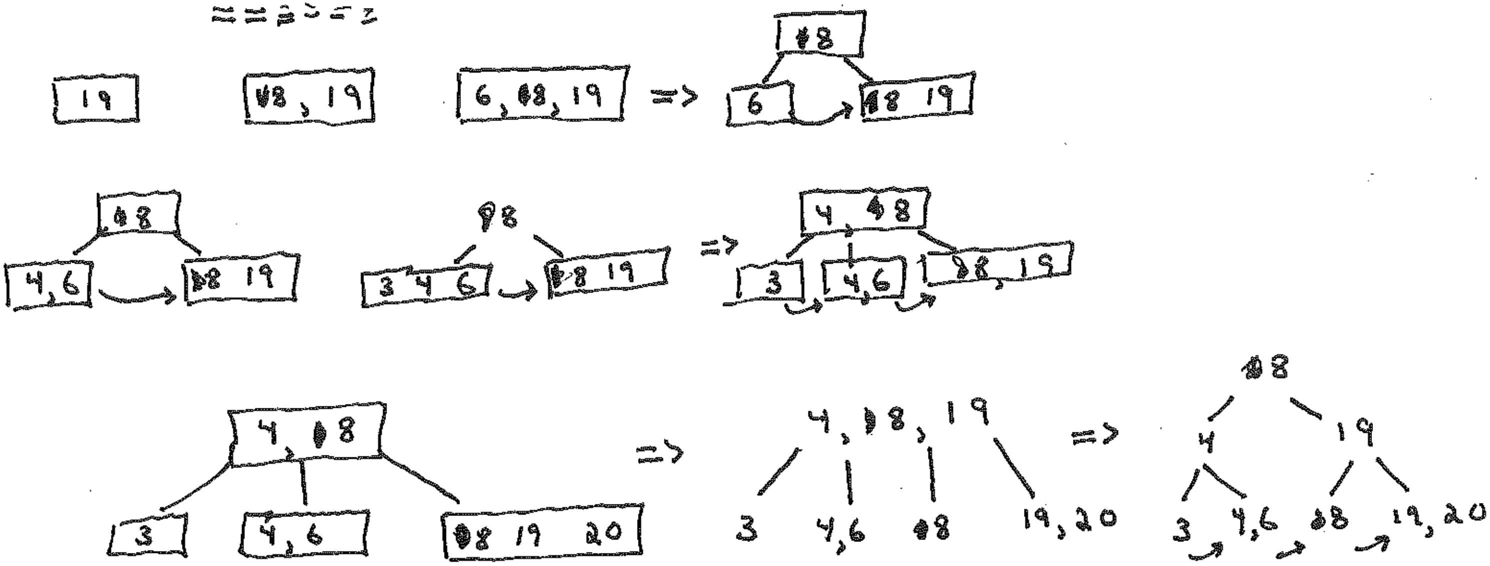


(b) Delete 1 from the final tree that you drew in 8 (a). Draw this final tree. (1 Points)



(c.) Insert the following numbers into a B+ tree. The min=1 and max = 2. Please note, this is a "B Plus" tree. (2 Points) Show the tree after each insertion.

19, 8, 6, 4, 3, 20



Project Questions (18 points)

Key

9. Recall the Merkle-Hellman cryptosystem that we worked with in Project 1, the spell checker application in Project 2, and the graph coloring problem from Project 3.

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 = {56,3,9,12,4,2,234} and k = 19. Is there a subset of X which sums to k?
Yes [✓ Yes]/No (2 points)

(b) The type of problem you were asked to solve in question 9 (a) is (Circle one answer): (2 Points)

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

(c) Suppose Alice sends a message (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 eaves dropper could read the message M if the eaves dropper could (2 Points)

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

(d) Recall that a modular inverse of an integer b mod m is the integer b^{-1} such that $(b^* b^{-1}) \bmod m=1$. What is the modular inverse of 4 mod 13? 10 (4 Points)

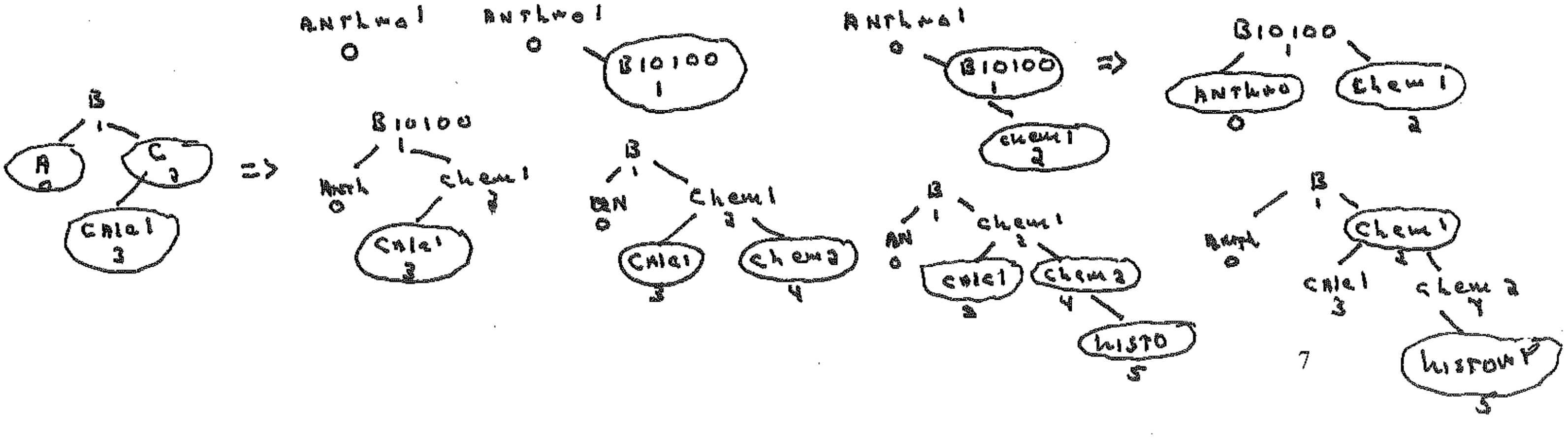
$4 \cdot 10 \bmod 13 = 40 \bmod 13 = 1 \bmod 13$

(e) In Project 3, we used a Red Black Tree to check for new course names or existing course names – assigning small integers counting from 0.

Draw what the Red Black Tree would look like after the following course names are read from the input. The tree must show the course name and the value assigned to each course.

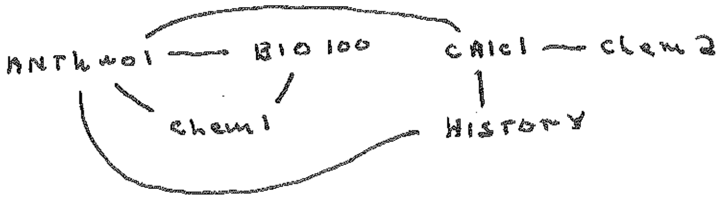
Draw RED nodes as circled and black nodes as un-circled. If you show each step clearly, you will receive partial credit. (4 Points)

Smith, Amy Anthro1, Bio100, Chem1
Bell,Amy Calc1, Chem2
Obama,Barak Calc1, History, Anthro1



Key

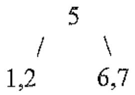
(f) In Project 3 we built a graph from data as in part e. Draw the graph that would result from reading the data in part e. (3 Points) This is not the Red Black Tree.



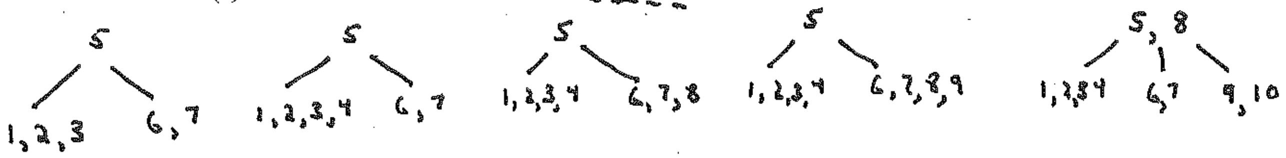
(g) The graph derived from the data in e. can be colored with two colors. (Circle True, or [✓ False]) (1 Point)

Balanced Trees (15 points)

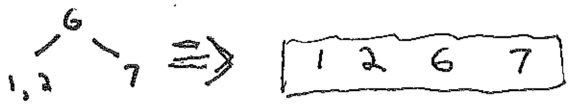
10. Consider the following B-Tree with a minimum of 2 and a maximum of 4.



(a) Redraw the tree after inserting 3,4,8,9,10. Show steps clearly for partial credit.(3 points)



(b) Delete the value 5 from the B-Tree shown above. Begin work from the original tree, not the tree with the values added in Part a. (2 points)



1 key too few

(c) Consider again the original tree of height of 1. What is the maximum number of keys that this type of tree (min = 2, max=4) could hold with a height of 1? (1 points). $4 + 5 \cdot 4 = 24$

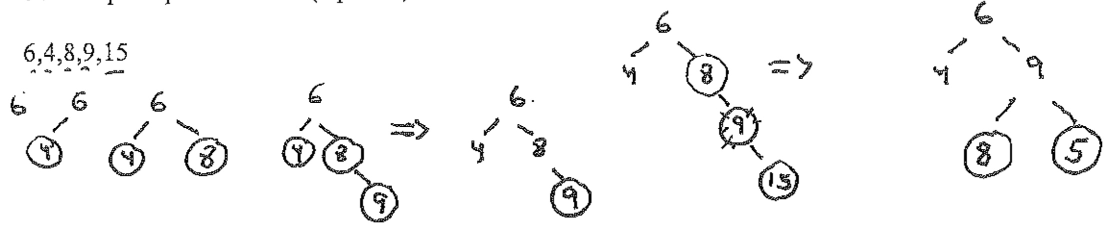
(d) Consider again the original tree of height of 1. What is the maximum number of keys that this type of tree (min = 2, max=4) could hold with a height of 2? (2 points). $4 + 5 \cdot 4 + 5 \cdot 5 \cdot 4 = 124$

Key

11. Red Black Trees

(a) Insert the following numbers, one by one, into a Red-Black Tree. Show the tree after each insertion. red vertices should be circled and black vertices should appear without circles. Show each step for partial credit. (5 points)

6,4,8,9,15



(b) What is the runtime complexity of an inorder traversal of a Red Black Tree ? Use Big Theta notation. (1 Point) $\Theta(N)$

(c) What is the worst-case runtime complexity of a Red Black Tree lookup operation? Use Big Theta notation. (1 point) $\Theta(\text{Log } N)$

Key

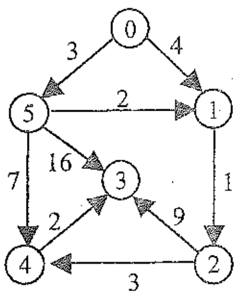
Graph Algorithms(21 points)

12. (a) What is the shortest path from the start node 0 to node 3 in the graph immediately below? Your path must be a list of ordered pairs.(1 point) ~~(0,1)~~ ((0,1),(1,2),(2,4),(4,3))

(b) Show the list of nodes that would be visited by a breadth first search in the graph below. We are starting from vertex 0. (1 point) 0, {1 5} {5 1} {3, 4, 2 IN ANY order}

(c) Draw the contents of the distance array for each iteration of Dijkstra's Algorithm as it works on this graph. The initial state is given. **Mark the node to be selected next to the left of the array (note how 0 is marked to the left of the first array.)** Fill in each array cell working downward. That is, complete the first column of arrays before the second column of arrays. (6 Points)

4 Pts



0	0	?	?	?	?	?
	0	1	2	3	4	5

		0				0
5	0	4	?	?	?	3
	0	1	2	3	4	5

		0		5	5	0
1	0	4	?	19	10	3
	0	1	2	3	4	5

		0	1	5	5	0
2	0	4	5	19	10	3
	0	1	2	3	4	5

				2	2	0
4	0	4	5	14	8	3
	0	1	2	3	4	5

		0	1	4	2	0
3	0	4	5	10	8	3
	0	1	2	3	4	5

2 pts.

(d) After Dijkstra is complete it also collects parent pointers for each node in the graph. Complete the chart below showing the parent of each node as computed by Dijkstra. (5 point)

Node	Parent
0	Nil
1	0
2	1
3	4
4	2
5	0

5 pts.

→ 0 ↗ 5
→ 1 → 2 → 4 → 3

(e) Assume that you have routines to mark a vertex, process a vertex and place a vertex in a queue. You also have a routine that allows you to access each unmarked neighbor of a vertex. Write an algorithm that performs a breadth first traversal on the graph like the one above. You will need to process each vertex visited by the breadth first traversal. That is, call process(x) for each vertex visited. The input to your algorithm is vertex v. (3 points)

ANY order

```
{ Put V ON Queue Q
{ Process(v)
{ MARK (v)
  While Q NOT empty
      v = dequeue
      For each unmarked neighbor of V
          Process, MARK, Add to Q.
```

3 pts.

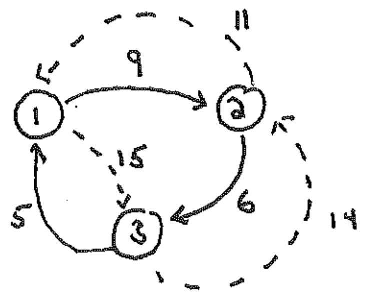
(f) If a directed graph with |V| vertices and |E| edges is represented by an adjacency matrix, what will be the space complexity of such a representation? (2 Points) $\Theta(|V|^2)$

KeY

(g) Below is a recursive version of Floyd Warshall. Your problem is to show the output of this program. Note that it is too time consuming to trace the recursion. You will need to think about what Floyd Warshall does and compute the correct values yourself. The values -1 are used to signify that this column or row is unused. Java uses 0 indexing in arrays but Floyd Warshall describes the first node in the graph as node 1. The values 1000 are used to signify infinity. Show the exact output as computed by this program. (3 Points)

```
public class FloydWarshallRecursive {

  public static int d(int w[][],int i, int j, int k){

      if ( k == 0) return w[i][j];

      else return Math.min(d(w,i,j,k-1),(d(w,i,k,k-1) + d(w,k,j,k-1)));
  }

  public static void main(String[] args) {

    int cost[][] = { {-1, -1, -1, -1}, {-1, 0, 9, 1000}, {-1, 1000, 0, 6}, {-1, 5, 1000, 0} };

    System.out.println(d(cost,1,2,3));     9
    System.out.println(d(cost,1,3,3));     15
    System.out.println(d(cost,2,1,3));     11
    System.out.println(d(cost,2,3,3));     6
    System.out.println(d(cost,3,1,3));     5
    System.out.println(d(cost,3,2,3));     14
  }
}
```

	0	1	2	3
0	-1	-1	-1	-1
1	-1	0	9	1000
2	-1	1000	0	6
3	-1	5	1000	0


11
9
1
2
15
5
6
14
3