King Fahd University of Petroleum & Minerals

College of Computer Science and Engineering

Information and Computer Science Department

First Semester 142 (2014/2015)

ICS 202 – Data Structures
Final Exam
Monday, May 25th, 2015
Time: 120 minutes

Name: _____

ID#									

Section 01 Dr. Sami
Section 02 Dr. Ramadan

Question #	CLO	Max Marks	Marks Obtained
1	3	25	
2	4	35	
3	4	20	
4	4	20	
Total		100	

Instructions

1. Write your name and ID in the respective boxes above and circle your section.
2. This exam consists of 10 pages, including this page, plus one reference sheet, containing 4 questions.
3. You have to answer all 4 questions.
4. The exam is closed book and closed notes. No calculators or any helping aids are allowed.
5. Make sure you turn off your mobile phone and keep it in your pocket if you have one.
6. The questions are not equally weighed.
7. The maximum number of points for this exam is 100.
8. You have exactly 120 minutes to finish the exam.
9. Make sure your answers are readable.
10. If there is no space on the front of the page, feel free to use the back of the page. Make sure you indicate this in order not to miss grading it.

Q.1 [25 points] Multiple Choice Questions: Mark the best answer for each question below. Note: only one choice should be chosen.

1. Consider the following recursive function f(n)

```
public static void f(int n)
{
   if (n == 1)
      System.out.println(n);
   else{
      f(n - 1);
      System.out.println(n);
      f(n - 1);
   }
}
```

   The number of times the printing statement is executed for the method f(n) is:
   a. n^2
   b. n^2-1
   c. 2^n
   d. 2^n-1
   e. none of the above.

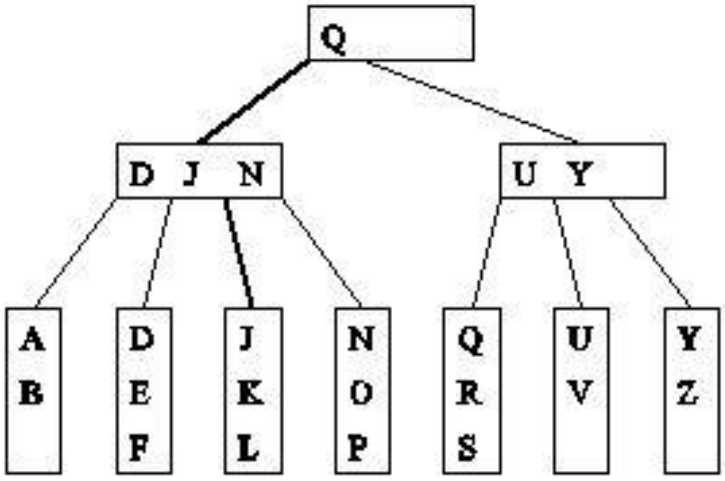
2. Consider the following code segment

```
for (int i=1, sum=0; i<n-1; i++)
     for (int j=i-1, sum=0; j<i+1; j++)
        sum+=4; // Statement 1
```

   The complexity of the above code segment is
   a. O (n^2)
   b. O (n log n)
   c. O (n)
   d. O ($2^n - 1$)
   e. none of the above.

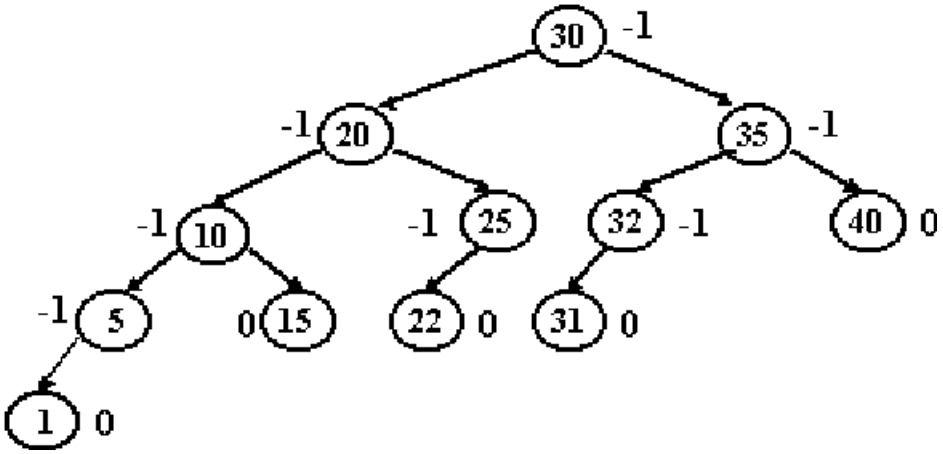
3. The big-O notation
   a. Can compare algorithms in the same complexity class
   b. Is an upper bound on the asymptotic complexity of the program
   c. Is a bottom bound on the asymptotic complexity of the program
   d. Provide a measure which is valid for different operating systems, compilers and CPUs.

4. Inserting M to the following B+ tree will result in how many splittings?



a. 0
b. 1
c. 2
d. 3
e. 4

5. Deleting 40 from the following tree will result in how many single rotations?



a. 0
b. 1
c. 2
d. 3
e. 4

Q2. [35 points] (Graphs):

A. Consider the following weighted directed graph G:

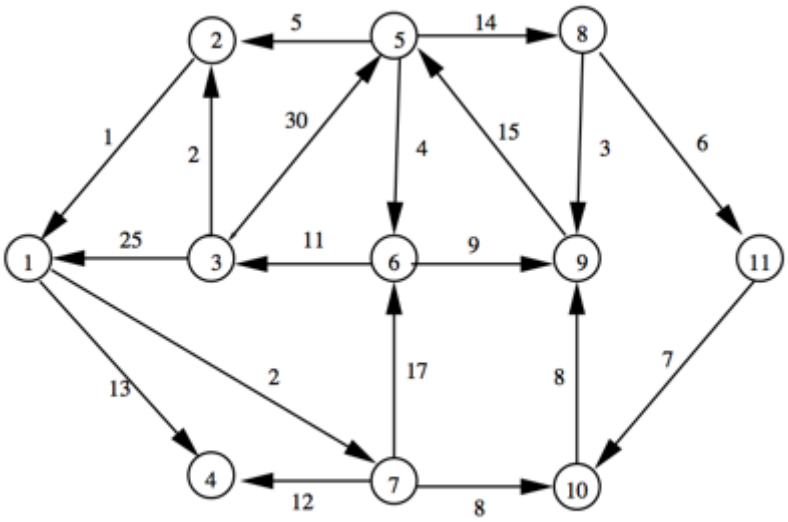


Figure 4

Apply the Dijkstra algorithm on G starting from **vertex 8**:

active	Initially												Weight	Pred
1														
2														
3														
4														
5														
6														
7														
8														
9														
10														
11														

B. Show how the table can used to find the shortest path between vertices 8 and 1.

C. Can the table be used to find the shortest path starting from vertex 1 going to vertex 11?

D. Fill in the following table with the big-O complexity of each operation

Operation / Data Structure	Adjacency Matrix	Adjacency List
Is there an edge from x to y		
Edge Insertion		
Edge deletion		
Get successor vertices of vertex x		
Get predecessor vertices of vertex x		
Visit all edges		
Space complexity		

E. Use Prim's algorithm to find a minimum spanning tree in the same graph.

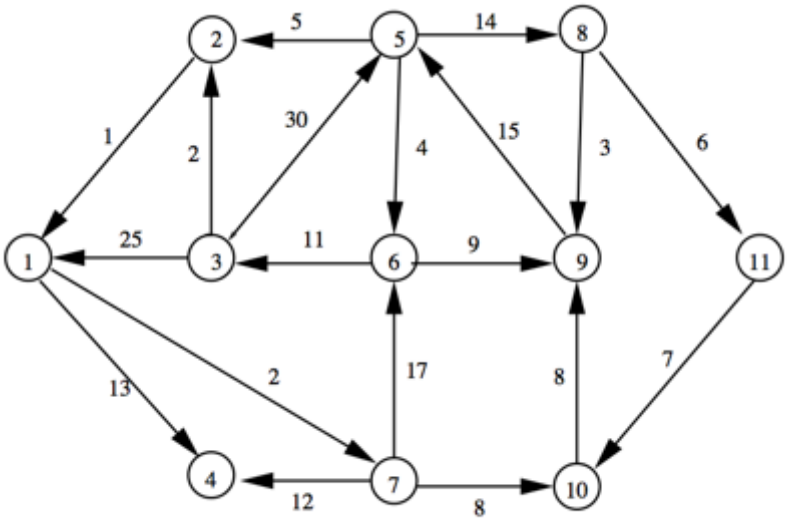


Figure 4

active	Initially												Weight	V1
1														
2														
3														
4														
5														
6														
7														
8														
9														
10														
11														

F. Draw the resulting minimum spanning tree.

Q3. [20 points] (Hashing):

Consider inserting the following keys:

33, 34, 31, 20, 21, 22, 44, 41, 8

respectively, into a hash table of size 11, using open addressing and hash function:

$h(key) = key \% 11$

A. Use quadratic probing as a collision resolution policy with **$c(i) = \pm i^2$**
Show the hash table after the insertions, showing all your work.

0	1	2	3	4	5	6	7	8	9	10

B. Use double hashing as a collision resolution policy with

$h_p(key) = 1 + key \% 10$

Show the hash table after the insertions, showing all your work.

0	1	2	3	4	5	6	7	8	9	10

Q.5 [20 points]: (**Compression**)

a) [6 points] Using Huffman coding, show the resulting Huffman coding tree for compressing the following message. Make sure you show all your work.

thisisthisisisss

b) [4 points] Compute the compression ratio, showing your work. Make sure you state any assumptions.

c) [6 points] Compress the following message using LZ78. Make sure you show all your work:

thisisthisisisss

d) [4 points] Compute the compression ratio, showing your work. Make sure you state any assumptions.

Quick Reference Sheet

```
public class SLLNode<T> {
    public T info;
    public SLLNode<T> next;
  public SLLNode();
  public SLLNode(T el)
  public SLLNode(T el, SLLNode<T> ptr);
}

public class SLL<T> {
    protected SLLNode<T> head, tail;
  public SLL();
  public boolean isEmpty();
  public void addToHead(T el);
  public void addToTail(T el);
  public T deleteFromHead();
  public T deleteFromTail();
  public void delete(T el);
  public void printAll();
  public boolean isInList(T el);
}

public class DLLNode<T> {
    public T info;
    public DLLNode<T> next, prev;
  public DLLNode();
  public DLLNode(T el);
  public DLLNode(T el, DLLNode<T> n,
                       DLLNode<T> p);
}

public class DLL<T> {
    private DLLNode<T> head, tail;
  public DLL();
  public boolean isEmpty();
  public void setToNull();
  public void addToHead(T el);
  public void addToTail(T el);
  public T deleteFromHead();
  public T deleteFromTail();
  public void delete(T el);
  public void printAll();
  public boolean isInList(T el);
}
```

```
public class Queue<T> {
     private …; // array or linked list
   public Queue();
   public void clear();
   public boolean isEmpty();
   public T firstEl();
   public T dequeue();
   public void enqueue(T el);
   public String toString();
}

public class BSTNode<T extends Comparable<?
super T>> {
    protected T el;
    protected BSTNode<T> left, right;
    public BSTNode();
    public BSTNode(T el);
    public BSTNode(T el, BSTNode<T> lt,
                         BSTNode<T> rt);
}

public class BST<T extends Comparable<?
super T>> {
    protected BSTNode<T> root = null;
    public BST();
    protected void visit(BSTNode<T> p);
    protected T search(T el);
    public void breadthFirst();
    public void preorder();
    public void inorder();
    public void postorder();
    protected void inorder(BSTNode<T> p);
    protected void preorder(BSTNode<T> p);
    protected void postorder(BSTNode<T> p);
    public void deleteByCopying(T el);
    public void deleteByMerging(T el);
    public void iterativePreorder();
    public void iterativeInorder();
    public void iterativePostorder2();
    public void iterativePostorder();
    public void MorrisInorder();
    public void MorrisPreorder();
    public void MorrisPostorder();
    public void balance(T data[], int first,
                                  int last);
    public void balance(T data[]);
    public void insert(T el)
}
```

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}, \quad \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}, \quad \sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2,$$

$$\sum_{i=0}^{n} x^i = \frac{x^{n+1}-1}{x-1}, \quad 2^{\lg n} = n, \quad \lg ab = \lg a + \lg b, \quad \lg a^b = b \lg a$$