Boğaziçi University, Dept. of Computer Engineering

CMPE 250, DATA STRUCTURES AND ALGORITHMS

Fall 2014, Midterm 1

Name: _____

Student ID: _____

Signature: _____

- Please print your name and student ID number and write your signature to indicate that you accept the University honour code.
- During this examination, you may not use any notes or books.
- Read each question carefully and **WRITE CLEARLY**. Unreadable answers will not get any credit.
- For each question you do not know the answer and leave blank, you can get %10 of the points, if you write only "I don't know the answer but I promise to think about this question and learn its solution".
- There are **4** questions. Point values are given in parentheses.
- You have **90 minutes** to do all the problems.

Q	1	2	3	4	Total
Score					
Max	20	20	40	20	100

1. Suppose we are given a graph with the following adjacency list representation of form

```
Source : (Target1, Weight) (Target2, Weight) .. (TargetN, Weight)

A :(D, 3) (E, 7)
B :(A, 1) (E, 9) (F, 13)
C :(B, 2) (D, 5) (F, 15) (G, 21)
D :(A, 4) (C, 6) (F, 17)
E :(A, 8) (B, 10) (C, 11) (D, 12) (F, 19)
F :(B, 14) (C, 16) (D, 18) (E, 20) (G, 24)
G :(C, 22) (D, 23) (F, 25)
```

   (a) Is this graph an undirected graph? Why, or why not?

   For parts (b) and (c) consider the dfs algorithm:

```
void dfs(vertex v) {
  visited[v] = true;

  print(v)  // Preorder  (1)

  for each vertex w adjacent to v
    if (not visited[w]) dfs(w)

  print(v)  // Postorder  (2)
}
```

   (b) Show the sequence of nodes printed by a depth first search traversal starting from node A, when the nodes are printed only the postorder (when there is no line (1)).

   ---- ---- ---- ---- ---- ---- ---- ----

   (c) Show the sequence of nodes printed by a depth first search traversal starting from node A, when the nodes are printed only the preorder (when there is no line (2)).

   ---- ---- ---- ---- ---- ---- ---- ----

   (d) Show the sequence of nodes visited by a breadth first search traversal starting from node A, in the order that they are poped out from the queue.

   ---- ---- ---- ---- ---- ---- ---- ----

(20 points)

2. a) Complete the below C++ function *ReverseLinkedList* which reverses a given linked list. Your functions space complexity must be O(1), which means that you cannot create and return a new list, but you can use a few temporary variables.

(15 points)

```
typedef struct Node {
  DataItem item;
  struct Node* next;
} Node;

void ReverseLinkedList( Node **head ){
```

b) Give three good test cases to test your function. Your set case set should cover different inputs to your function. (Think about possible problematic cases). Make sure your function handles those test cases.

(5 points)

3. (Minimum spanning tree). Consider the following weighted undirected graph with the adjacency list representation

```
A :(A C, 30) (A D, 29)
B :(B D, 26) (B E, 11)
C :(C A, 30) (C D, 25) (C E, 23) (C F, 27)
D :(D A, 29) (D B, 26) (D C, 25) (D E, 30) (D F, 18) (D G, 26)
E :(E B, 11) (E C, 23) (E D, 30) (E G, 20)
F :(F C, 27) (F D, 18) (F G, 3)
G :(G D, 26) (G E, 20) (G F, 3)
```

(a) (10 pts) Show the final priority queue when all edges are inserted one by one in the order given above, i.e., AC, AD, BD etc. (When the weight of the parent is equal to its child, don't swap).

```
                                        FG 3
                                        ----

                    ----                                    ----

          ----                  ----                ----                  ----

    ----      ----      ----        ----      ----        ----      ----        ----

----  ----  ----  ----  ----  ----  ----  ----  ----
```

(b) (10 pts) Show the final priority queue when the heap is build from the below array using the build heap method discussed in clas. (When the weight of the parent is equal to its child, don't swap).

```
(AC, 30) (AD, 29) (BD, 26) .... (GE, 20) (GF, 3)
```

```
                                        GF 3
                                        ----

                    ----                                    ----

          ----                  ----                ----                  ----

    ----      ----      ----        ----      ----        ----      ----        ----

----  ----  ----  ----  ----  ----  ----  ----  ----
```

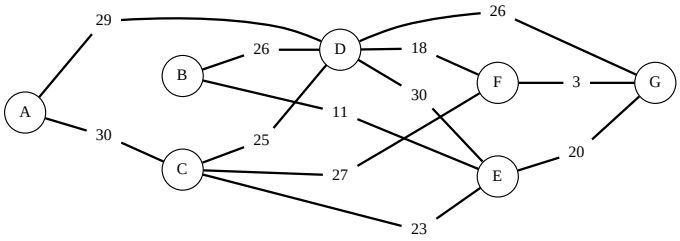
(c) (10 pts) Complete the sequence of **edges** in the MST in the order that Prims's algorithm includes them.

---- ---- ---- ---- ---- ---- ---- ----

(d) (5 pts) Complete the sequence of **edges** in the MST in the order that Kruskal's algorithm includes them.

---- ---- ---- ---- ---- ---- ---- ----

(e) (5 pts) Find the minimum spanning tree and find the total weight.



4. A graph is connected if there is an undirected path from every vertex to every other vertex in the graph. A graph that is not connected consists of a set of connected components, which are connected subgraphs.

   (a) For a graph represented just as a list of edges, describe a method using the disjoint set class data structure (the one used in Kruskal's algorithm) to find the number of connected components. (no disjoint class, no points!)

   (b) Draw the resulting disjoint set class data structure (a tree where each node points to its parent node) for the following graph represented as a list of edges.

```
0 5
4 3
0 1
9 12
6 4
5 4
0 2
11 12
9 10
0 6
7 8
9 11
5 3
```

(20 points)