Boğaziçi University, Dept. of Computer Engineering

CMPE 250, DATA STRUCTURES AND ALGORITHMS

Spring 2012, Midterm 2

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 **5** questions. Point values are given in parentheses.
- You have **100 minutes** to do all the problems.

Q	1	2	3	4	5	Total
Score						
Max	15	15	20	30	20	100

1. (Sorting) Fill in the following table. (Leave empty if you are unsure as a wrong answer cancels one right answer)

	Insertion Sort	Heapsort	Mergesort	Quicksort
Stable? (yes/no)				
In Place? (yes/no)				
Worst case time complexity				
Average case time complexity				

(15 points)

2. (Partition in Quicksort) For the following array show the result of calling `partition` with

   (a) Center element

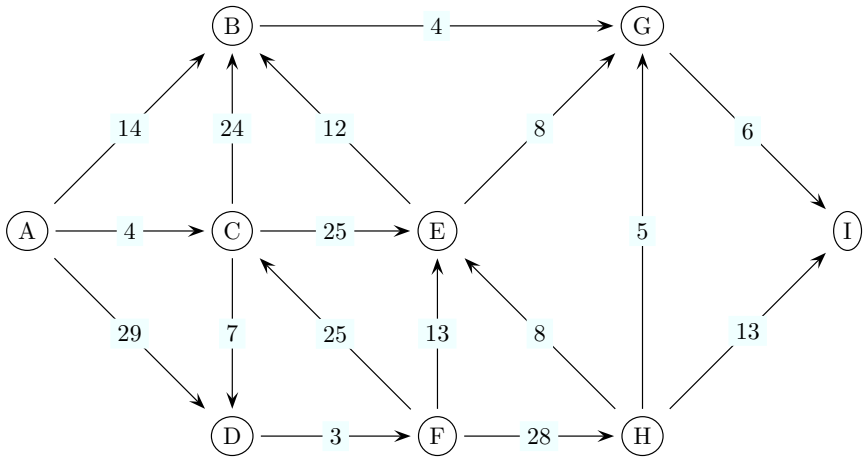
   (b) Median of the first three elements

   as the pivot. The pivot selection is assumed to have no side effects.

$$5, 91, 9, 12, 34, 3, 25, 41, 8$$

(15 points)

3. Run Dijkstra's algorithm on the weighted digraph below, starting at vertex A.



(a) List the vertices in the order in which the vertices are dequeued (for the first time) from the priority queue and give the length of the shortest path from A.

```
vertex:  A C ___ ___ ___ ___ ___ ___ ___
distance: 0 4 ___ ___ ___ ___ ___ ___ ___
```

(b) Draw the edges in the shortest path tree with thick lines in the figure above.

(20 points)

4. (Perfect matching) Consider a bipartite graph $G = (R \cup C, E)$, a graph such that each edge has one endpoint in R and one endpoint in C, and R and C have the same size.

   (a) Below is part of the adjacency matrix of G showing links from R to C

	C_1	C_2	C_3	C_4	C_5
R_1	0	1	0	1	1
R_2	1	0	0	0	1
R_3	0	1	1	1	1
R_4	0	1	0	1	0
R_5	0	1	0	1	0

   Draw this bipartite graph.

   (b) Describe an algorithm to find a set of edges $M \subset E$ such that M is a perfect matching, that is edges in M don't touch eachother and each node in R is associated with a single node in C. This is also known as the stable marriage problem. Using the algorithm find such a perfect matching and show it on the graph you drew. You must show all the intermediate steps

(30 points)

5. (C++)

(a) The following class needs a proper copy constructor, destructor and overloaded assignment operator. Implement those functions properly (make deep copy, no memory leak, etc...) *(13 points)*

```
template <class T>
class A{
public:
        A(int n){
                this->n = n;
                t_ptr = new T[n];
        }

        /* implement the copy constructor here: */









        /* implement the destructor here: */









        /* overload the assignment operator here: */









private:
        T *t_ptr;
        int n;
};
```

(b) What is the output of the following program? Justify your answers ! *(7 points)*

```
#include <iostream>
using namespace std;

template <typename T>
struct obj{
        T i;
        obj(T j=0) : i(j) {};
        ~obj(){cout<<'-';};
        void swap(obj<T> o){
                cout<<"s";
                T temp = i;
                i = o.i;
                o.i = temp;
        }
};

template <typename T>
void swap(obj<T> o1, obj<T> o2){
        cout<<"w";
        T temp = o1.i;
        o1.i = o2.i;
        o2.i = temp;
}

int main(int argc, char *argv[]){
                                                    // outputs:
        obj<int> o1(3.7);
        obj<int> o2(5);
        swap(o1,o2);                                // ---------- (2pts)
        cout<<o1.i<<" "<<o2.i;                      // ---------- (1pts)
        o1.swap(o2);                                // ---------- (2pts)
        cout<<o1.i<<" "<<o2.i;                      // ---------- (1pts)
        return 0;                                   // ---------- (1pts)
}
```