Boğaziçi University, Dept. of Computer Engineering

CMPE 250, DATA STRUCTURES AND ALGORITHMS

Fall 2012, Final

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 **8** questions. Point values are given in parentheses.
- You have **150 minutes** to do all the problems.

Q	1	2	3	4	5	6	7	8	Total
Score									
Max	20	10	10	10	20	20	20	20	150

1. For each function given below, find the order of growth of the *running time* in terms of *Big-Oh* notation. **Justify each of your answers.**
(It is clear that the first function runs in $O(N)$ time.)

```
__O(N)___    int f1(int N) {
                 int x = 0;
                 for (int i = 0; i < N; i++)
                     x++;
                 return x;
             }

_______      int f2(int N) {
                 int x = 0;
                 for (int i = 0; i < N; i++)
                     for (int j = 0; j < i; j++)
                         x += f1(j);
                 return x;
             }

_______      int f3(int N) {
                 if (N == 0) return 1;
                 int x = 0;
                 for (int i = 0; i < N; i++)
                     x += f3(N-1);
                 return x;
             }

_______      int f4(int N) {
                 if (N == 0) return 0;
                 return f4(N/2) + f1(N) + f1(N) + f1(N) + f4(N/2);
             }


_______      int f6(int N) {
                 if (N == 0) return 1;
                 return f6(N-1) + f6(N-1) + f6(N-1);
             }

_______      int f7(int N) {
                 int x = 0;
                 while (N > 0) {
                     x++;
                     N = N / 2;
                 }
                 return x;
             }
```

(20 points)

2. Short Definitions (Give concise answers) (2pts each)

   2.1. Express the following statement in big-Oh notation: for some constant C and n_0, $Cf(n)$ is an upper bound of $g(n)$ for $n > n_0$.

   2.2. What is a residual flow graph?

   2.3. What is memoization (Yes, memoization, not memorization)?

   2.4. (4 pts) Fill in the blanks: [(a)______________] is a method for solving a computational problem expressed as a [(b)______________] of steps. Each step can be specified by a list of [(c)______________] that describe a computation. For a given initial state and admissible input, the computations proceed through a well-defined series of successive states and eventually [(d)______________] .

   - (a):
   - (b):
   - (c):
   - (d):

(10 points)

3. Heap Data Structures. Consider a min-heap A storing the values 1, 2, 3, ..., 15. Answer each of the following questions, and justify your answers. Assume that A[0] is empty.

   3.1. Where in the heap can the value 1 possibly go?

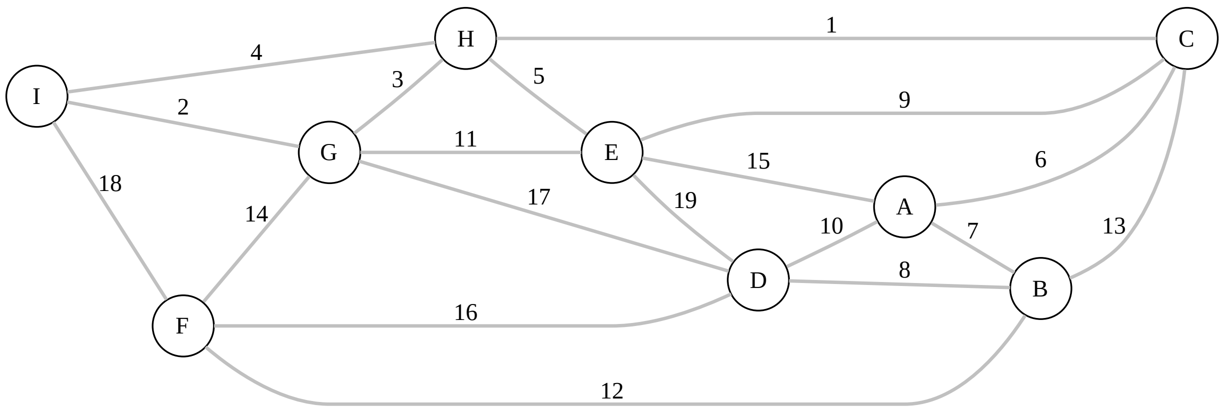
   3.2. Which values can possibly be stored in entry A[2]?

   3.3. Where in the heap can the value 15 possibly go?

   3.4. Where in the heap can the value 6 possibly go?

(10 points)

4. **Maximum** spanning tree. For parts (a), and (b) consider the following weighted graph with 9 vertices and 19 edges. Note that the edge weights are distinct integers between 1 and 19.



4.1. Complete the sequence of edges in the MST in the order that Kruskal's algorithm includes them.

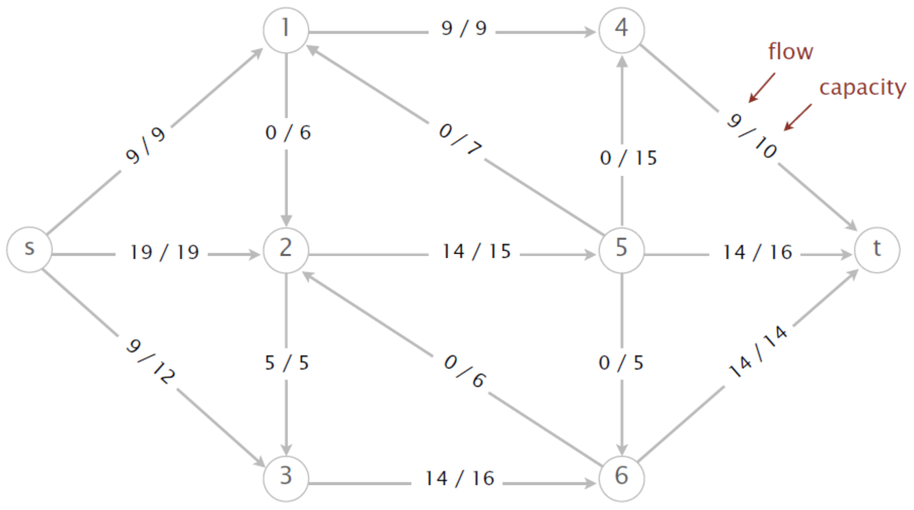
```
19 ____ ____ ____ ____ ____ ____ ____
```

4.2. Complete the sequence of edges in the MST in the order that Prims's algorithm includes them.

```
15 ____ ____ ____ ____ ____ ____ ____
```

(10 points)

5. Maximum Flow. Consider the following st-flow network and feasible flow f.



5.1. What is the value of the flow f?

5.2. Perform one iteration of the maximum flow algorithm, starting from the flow f. Give the sequence of vertices on the augmenting path.

5.3. What is the value of the maximum flow?

5.4. List the vertices on the t side of the minimum cut.

5.5. What is the capacity of the minimum cut?

(20 points)

6. Consider the 3-PARTITION problem. Given integers $a_1, \ldots, a_n$, we want to determine whether it is possible to partition of $\{1, \ldots, n\}$ into three disjoint subsets I, J, K such that

$$\sum_{i \in I} a_i = \sum_{i \in J} a_j = \sum_{i \in K} a_k = \frac{1}{3} \sum_{i=1}^{n} a_i$$

   For example, for input $(1, 2, 3, 4, 4, 5, 8)$ the answer is yes, because there is the partition $(1, 8)$, $(4, 5)$, $(2, 3, 4)$. On the other hand, for input $(2, 2, 3, 5)$ the answer is no. Devise a dynamic programming algorithm for 3-PARTITION that runs in time polynomial in n and in $a_1 + a_2 + \cdots + a_n$.

   [Hint: Don't write a program, define the appropriate indices and just write the formula for filling up the table.]

   (20 points)

Name: ______________________________

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

	Insertion Sort	Heapsort	Mergesort	Quicksort
Sequence num				
Stable?				

Below, the column on the left is the original input of strings to be sorted; the column on the right are the string in sorted order; the other columns are the contents at some intermediate step during one of 7 sorting algorithms some of which are listed above. Match up each column by writing its number to the corresponding row labeled as 'sequence'. Use each number exactly once. **Briefly, justify your answer**.

```
rush abba blue abba fixx abba neyo zman abba
korn acdc cars blue inxs acdc korn yani acdc
fixx blue devo cars korn beck fixx yoyo beck
inxs beck enya devo rush blue inxs tatu blue
cars cars fixx dido cars cake cars styx cake
enya cake fuel enya devo cars enya ween cars
devo devo inxs fixx enya cher devo seal cher
fuel epmd korn fuel fuel devo fuel lons devo
tatu cher moby inxs blue dido lons kiss dido
styx inxs rush korn moby doom mims nofx doom
blue dido styx moby styx enya blue pras enya
moby fuel tatu muse tatu epmd moby rush epmd
abba doom abba rush abba rush abba neyo fixx
muse kiss muse seal dido muse muse muse fuel
seal enya seal styx muse seal cher mims inxs
dido lons dido tatu seal tatu dido fuel kiss
beck fixx beck acdc acdc fixx beck beck korn
kiss neyo kiss beck beck kiss kiss inxs lons
acdc korn acdc doom kiss korn acdc acdc mims
yani moby yani kiss yani yani epmd cars moby
nofx muse nofx nofx doom nofx nofx korn muse
doom pras doom pras nofx styx doom doom neyo
pras mims pras yani pras pras pras blue nofx
yoyo seal yoyo yoyo yoyo yoyo cake moby pras
ween nofx ween ween cake ween rush fixx rush
zman tatu zman zman neyo zman zman abba seal
neyo rush neyo neyo ween neyo ween enya styx
cake yani cake cake zman inxs yoyo cake tatu
epmd ween epmd epmd cher moby yani epmd ween
cher zman cher cher epmd fuel seal cher yani
mims styx mims mims lons mims styx devo yoyo
lons yoyo lons lons mims lons tatu dido zman
--------------------------------------------
 U    1    2    3    4    5    6    7    S
```

(20 points)

8. Suppose we have n objects corresponding and each object is either of class A or of class B. For each object, we can't directly observe its class but we can compare any two objects and decide if these belong to the same class or not. In other words, for each pair of objects i and j, we label the pair (i, j) either "same" (meaning we believe them both to come from the same class) or "different" (meaning we believe them to come from different classes). We also have the option of rendering no judgment on a given pair, in which case well call the pair ambiguous. So now we have the collection of n objects, as well as a collection of m judgments (either "same" or "different") for the pairs that were not declared to be ambiguous. We would like to know if this data is consistent with the idea that each object is from one of the classes A or B. So more concretely, we'll declare the m judgments to be consistent if it is possible to label each specimen either A or B in such a way that for each pair (i, j) labeled "same," it is the case that i and j have the same label; and for each pair (i, j) labeled "different," it is the case that i and j have different labels. Give an algorithm with running time $O(m + n)$ that determines whether the m judgments are consistent.

(20 points)