# Applied Algorithm Design: Exam

Prof. Pietro Michiardi

SPRING 2013

## Rules and suggestions

- The exam is composed solely of questions related to the course contents
- The following evaluation scheme is used:
  - A correct answer: +2 points
  - An empty answer: 0 points
  - A wrong answer: -2 points

**Note**. When multiple sub-questions appear, e.g. in question 1), points are awarded or deduced by spreading evenly the total points on all sub-questions. This means, for example: each sub-question in question 1) is worth 0.5 points. If you answer correctly the first three bullets and you get the last bullet wrong, you will obtain 0.5*3-0.5 = 1 point. If you answer correctly the first two bullets and leave the two other bullets empty, you will obtain 0.5*2+0=1 point.

# Questions

1. **The Gale-Shapley algorithm**. Please, answer the following questions:

    - What is the best data structure to store men, women, and their respective preference lists? Please, simply name the data structure.
    - What is the best data structure to store the list of free men, that appear in the main `while` loop of the algorithm? Please, simply name the data structure.
    - Given the above selection of data structures, what is the worst-case running time of the Gale-Shapley algorithm? Please, simply use the $O()$ notation.
    - Why is it important for the Gale-Shapley algorithm to output the same result no matter what the proposal order is? Please, be brief.

2. Give an example of an algorithm for each of the following worst case running times. Please, be brief; well-known and simple algorithms are accepted as good answers to the question.

    - $O(n)$: linear running time
    - $O(n \log n)$
    - $O(n^2)$: quadratic running time

3. **Graph traversal**. Please, name an algorithm to find, given an undirected graph $G = (V, E)$, if there exist a path from $s \in V$ to $t \in V$. A pseudo-code for the algorithm is *not* necessary for a correct answer.

4. Given an undirected graph $G = (V, E)$, name a simple algorithm to check for bipartiteness. A pseudo-code for the algorithm is *not* necessary for a correct answer.

5. Please, define and explain (in a few bullets) the pros and cons of the two following data structures to store a **sparse** graph $G = (V, E)$:

    - Adjacency matrix
    - Adjacency list

6. Name and explain (in a few bullets) an important feedback centrality index we studied in class. Please, answer briefly

7. **The PageRank Algorithm**. Draw a simple (e.g. 5 nodes) graph, and illustrate (in a sequence of graphs) at least two iterations of the PageRank algorithm: in particular, each node in the drawings should indicate what is the current PageRank value.

8. **Data clustering**: please, give the pseudo-code of the $k$-means algorithm applied to data points in a 2-dimensional plane. In doing so, please also clearly formulate the problem statement you are addressing with the $k$-means algorithm.

   [HINT:] look at the questions below, where a short description of a problem statement is given as a starting point.

9. **Interval scheduling**. We have a set of requests $\{1, \cdots, n\}$ where the $i$-th request corresponds to an interval of time starting at $s(i)$ and finishing at $f(i)$. We say that a subset of the requests is compatible if no two of them overlap in time. Please, answer the following questions:

   - Give the pseudo-code of an algorithm that accepts as large a compatible subset of requests as possible, that is we want an optimal schedule.
   - Find and disucss the worst-case running time of your algorithm.

10. **Load Balancing**. We are given a set $M = \{M_1, M_2, \cdots, M_m\}$ of $m$ machines and a set $J = \{1, 2, \cdots, n\}$ of $n$ jobs, with job $j$ having a processing time $t_j$. Please, answer the following questions:

    - Give the pseudo-code of an algorithm that assigns each job to one of the machines so that the “load” placed on all machines is optimally “balanced”.
    - If optimality cannot be achieved, simply state (no proofs are required) what is the approximation quality achieved by your algorithm: please use the $\alpha$ notation, by stating that your algorithm is an $\alpha$-approximation algorithm.