Homework Assignment 1

2-AIN-205, Spring 2020

Deadline: 16.3.2020, 14:00, M-163 (under the door)
programming task until 23:59

Before you start working on this homework assignment, read general instructions at the end of this document. You can write you solutions in Slovak or English. The solutions must be your work. Do not copy from others and do not attempt to find the solutions in literature or on the internet!

1. [*20 points*] **Clumsy Coins.** One cent and two cent coins are more expensive to make than they are worth. So perhaps in a short while, the European Commission will decide to discontinue them. Since there will then be a free compartment in the cash registers, they will likely introduce a 25 cent coin. So as a result, we will have coins with values 5, 10, 20, 25, 50, 100, and 200 cents and all the prices will be rounded to the nearest 5 cents.

   The cashiers in the stores today use a simple algorithm to give a change back to customers. They repeatedly pay out the largest coin that does not exceed the sum that remains to be paid. In the current system of coins, this algorithm always led to the least possible number of coins.

   a) Prove that after the new 25 cent coin is introduced, this greedy algorithm will no longer lead to a payment with the smallest possible number of coins for certain sums.

   b) Show that the results will not be too bad: We will always use at most one additional coin compared to the best possible solution.

2. [*20 points*] **Bureaucrats.** This task is about dividing work between k bureaucrats. We need to perform n administrative tasks. In particular, we are given numbers $a_1, \ldots, a_n$, where number a_i represents the length of the i-th administrative task. The administrative tasks can be performed in any order.

   Our goal is to distribute the tasks between the bureaucrats in such a way that the last task will be completed as soon as possible. For example, if we had 7 tasks with lengths 1, 4, 10, 7, 14, 2, 2, we could distribute them between 3 beaurocrats so that the first beraucrat will get a single task with length 14, second would get tasks with lengths 10, 2, 2, and the third would perform tasks with lengths 7, 4, 1. In fact, this is the optimal solution and it has the length of 14.

   In this task, we will consider the following algorithm A:

   > Sort all tasks from the longest to the shortest. We will then distribute the tasks in that order and we will always assign the task to the bureaucrat with the smallest total amount of work. (The ties are broken arbitrarily.)

   a) Algorithm A will sometimes give a suboptimal solution. Find an example where this happens. (Also show the result of algorithm A and the optimal solution for your example.)

   b) There are cases for which we can prove that the algorithm A leads to the optimal solution.

   Prove the following claim: Let t^* be the length of the optimal solution. If $a_i > t^*/3$ for all $1 \leq i \leq n$, then algorithm A will always return the optimal solution.

   c) Using the claim from part b), prove that the algorithm A is a 4/3-approximation.

3. [*20 points*] **Programmming Task.** We are given a complete oriented graph. The goal is to remove some vertices so that the resulting graph is acyclic. Your solution does not need to remove the smallest number of vertices, but is required to be a 3-approximation.

   Input format: On the first line there is number of vertices n. After the first line, there is $n(n-1)/2$ lines, on each line there are two numbers a, b, which means that there is an edge directed from vertex a to vertex b.

Output format: On the first line, write number of vertices removed k. On the second line, write k numbers separated by a space, which represent the vertices that are removed.

Constraints and grading: To get the full grade, your program is required to give a correct answer for inputs with $n \leq 200$.

Input example:

```
4
1 2
2 3
3 1
1 4
2 4
3 4
```

Output example:

```
2
1 2
```

Note: This solution is not optimal, removing vertex number 2 is sufficient.

Hint: It is necessary to at least remove the cycles of length 3. What to do with other cycles? Recall the 2-approximation for vertex cover.

General Instructions

Theoretical tasks. The answer to theoretical tasks submite handwritten or printed on a paper under the door of the office M-163. No late submission are allowed. Do not forget to write your full name legibly on each sheet and also staple your solutions.

Write your solutions so that they contain all information necessary to easily understand them, but at the same time try to aim for brevity. Prove all claims, including in the cases when it is not explicitly written in the problem statement.

If the task is about solving an algorithmic problem, submmit the best algorithm you can design. The first criterion is that the algorithm *is correct*, the second criterion is the *running time and memory complexity*. Correct and slow algorithm is worth more points than fast incorrect algorithm. Inefficient but correct algorithms will always receive at least 50% of points. In your solution, you should:

- Describe the main idea of the algorithm.
- Write a pseudocode for your algorithm.
- Prove the correctness of your algorithm.
- Analyse its running time and memory complexity.

Programming tasks. You need to submit a functional program, no written part is required. Submit the solution through the web interface `https://testovac.ksp.sk/tasks/`. Your solutions will be evaluated immediately on several inputs and you will learn number of points (there are several sets of inputs and you only gain points if you solve correctly all inputs in a particular set). You can submit a solution several times, we will only take into account the last submitted solution before the deadline. You need to register before you can submit solutions (see the panel on the left side of the web page). The details about how to write and submit your programs (including supported programming languages) are in the section "Čo odovzdávať?".