**Algorithms for Optimization**

The material seen in Lecture A is completely sufficient to complete the assessed task on Greedy algorithms.

Below you can find some **extra formative exercises** to complement your knowledge on the topic of Greedy algorithms with a variety of different exercises often found in job interviews.

**Exercise 1**

This is a variant of *a typical job interview question*.

To pass the Maths unit, students need to complete exercises that give credits.

There are five types of exercises:

type A that gives 1 credit.

type B that gives 5 credits.

type C that gives 10 credits.

type D that gives 25 credits.

type E that gives 50 credits.

Students can freely choose to mix exercises of the five available types (even choosing many times the same type of exercise). There is no limit on how many times a student can choose the same type of exercise. For example, a student could choose to complete 10 exercises of type A (each of them giving 1 credit), 1 exercise of type B (giving 5 credits), and 1 exercise of type E (50 credits) getting in this way a total of 65 credits.

Implement in C# an efficient method minexercises which takes as input an int n (the number of credits necessary to pass the unit) and returns the **minimal** number of exercises that students need to complete to obtain exactly the requested n credits. The method should also display the credits of the selected exercises.

Example: If n= 42, for the types of exercises shown above, the method should return the number 5 which is the minimal number of exercises that students need to complete to obtain the requested 42 credits.

The method should also display

25, 10, 5, 1, 1

which are the credits associated to the 5 selected exercises: 1 exercise of type D of 25 credits, 1 exercise of type C of 10 credits, 1 exercise of type B of 5 credits and 2 exercises of type A, each one of 1 credit.

*Hint: Implement a more general method*

*MinExercises(int n, Exercise[] arrayex )*

*{*

*}*

*which takes as input the number of credits to pass the unit (n) and an array arrayex of the different types of exercises and their associated credits.*

**Exercise 2**

Refer to the above exercise and assume that they decide to allow only 3 types of exercises:

type A that gives 1 credit.

type B that gives 15 credits.

type C that gives 25 credits.

Can you still reuse the greedy algorithm implemented in Exercise 1 to obtain the minimal number of exercises to obtain exactly the requested n credits ? If not, find an example where it would not work.

**Exercise 3 [typical at job interviews]**

Implement an efficient algorithm in C# which takes as input an array A of single digit numbers (e.g., [0, 1, 4, 5, 8, 6, 5, 0, 5, 1]) and displays the largest possible number which can be obtained by concatenating the elements in the array (choosing each element in the array **exactly once**).

*Example*

*If the array is [2, 1, 3], the largest number can be obtained by concatenation is 321. What is the greedy algorithm idea ?*

**Exercise 4**

Modify the code of Exercise 3 to take as input an array of chars and displays the largest string (in lexicographical order) that can be obtained by concatenating the elements in the array (choosing each element in the array exactly once).

**Exercise 5 Minimal number of jumps [typical at job interviews]**

Implement an efficient (greedy) method in C# minjumps that takes as input an int K and an array of size n where each element of the array represents:

* either a safe location (Y)
* or an unsafe location (X).

See the figure below.

A person starts at position 0 (leftmost) and needs to reach the target location, which is in position n-1 (rightmost). The person can jump at most K locations and must always jump from a safe location to another safe location.

The method minjumps should display the sequence of locations the person should cross to go from the starting location to the target location using the **minimal** number of jumps.



Y

X

X

Y

X

Y

Y

X

Y

Y

X

0 1 2 3 4 5 6 7 8 9 10

**Exercise 6 Minimal stops at Petrol Stations [typical at job interviews]**

Implement an efficient method in C# minstops that takes as input an int K and an array of size n where each element of the array represents:

* either a petrol station where it is possible to fill the tank of the car (Y)
* or a place where it is not possible to fill the tank of the car (X).

See the figure below for an example.

A car starts at position 0 (leftmost) and needs to reach the target location, which is in position n-1 (rightmost). The car can travel K locations before running out of gas. The goal of the driver is to **minimize the number of stops** at the petrol stations along the route to reach the target.

The method minstops should display the sequence of stops at petrol stations that the car should do to go from the starting location to the target location using the minimal number of stops.

Y

X

X

Y

X

Y

Y

X

Y

Y

X

0 1 2 3 4 5 6 7 8 9 10

**Exercise 7 [Advanced] Rumor Spreading**

There are n people, each in possession of a different rumor. They want to share the news with each other by sending electronic messages. Design an algorithm which takes n as input and output the minimum number of messages they need to send to guarantee that everyone gets all the rumors. Assume that a sender includes all the rumors he or she knows at the time the message is sent and that a message may only have one addressee.

Hints: The minimum number of messages for n =4 is six

*From A. Levitin, M. Levitin, Algorithmic Puzzles.*