Big Homework 2

Data Structures and Algorithms
Linked Lists & Graphs

General mentions

• For this project you will work either in teams of 2 people or alone.

• The homework will be uploaded on the Moodle platform (fils.curs.pub.ro), for Hw2 assignment. If you encounter problems with the platform, contact by email your lab assistant.

• The homework must be submitted by **22.04.2019, at 08:00 AM (Monday morning)**. No late submissions will be accepted.

• The evaluation of the project will take place during the first lab after the submission deadline. Presence at the lab will be mandatory for presenting your projects.

• The final submission will contain an archive named Student1FamilyName_Student1Name_ Student2FamilyName_Student2Name_HW1 with:

- the source files of your project (.cpp and .h) and not the object files (.o) or executables (.exe)
- a README file in which you will specify all the functional sections of the project, together with instructions for the user; additionally, if you have parts of the homework that don't work, you may offer solution ideas for a partial score on these sections.
- .txt files, if asked.

• For all questions regarding the project, communicate by email with your lab assistant.

• **Warning:** we will use plagiarism detection software on your submissions. Copied homework will be marked with **0 points**.

! Observation: The linked list and graph used will be in headers (.h) and will be generic classes (template) – the implementation from the course, at which you can add other methods, if necessary.

Tasks:

1. **Projects at FILS Scientific Session (5p)**

   The Data Structures teacher wants to distribute projects to students, for them to participate at FILS Scientific Session. She prepared k projects and she has n students who want to participate (n>=k). Each project must be given at least once, but the n students can work in teams to implement a certain project. This means that each project is given to a team of students, which is built from 1 or more students.

   The values n and k will be read from the console, as well as the title of the subject and the name of the students.

   You will have to:

   1.1. (2p) Display at the console all possibilities of distributing the k projects to the n students, as well the number of possible solutions, in the below form:

   For n=3 (students), k=2 (projects)

   There are 6 solutions:

   Tom - "Java Animation", Jane-"Java Animation", Kate - "3D Game"

   Tom - "Java Animation", Jane - "3D Game", Kate- "Java Animation"

   Tom-"Java Animation",Jane-"3D Game",Kate-"3D Game"

Tom - “3D Game”, Jane - “Java Animation”, Kate –“Java Animation”
Tom - “3D Game”, Jane - “Java Animation”, Kate – “3D Game”
Tom - “3D Game”, Jane - “3D Game”, Kate –“Java Animation”

1.2. (1.5p) Use a linked list to store the distribution from point 1.1. Each element of the linked list will contain a student (id and name) and a project (id and title). You will know that every n-th element of the list is the ending of a distribution. Implement the add function, in which you add a pair of (student, project) to the list.

1.3. (1p) A student is sick, and the teacher has to delete the student from all possible distributions. Implement the delete function.

1.4. (0.5p) The “3D Game” project uses an Oculus Rift for implementation and the Oculus is broken. The teacher decides to replace the project from the list with another project. Implement the replace function.

2. Johny’s Trip (2pts)

Johnny bought a car and decided to visit all his friends, but he has a lot of friends, on every street he has another friend. He started thinking how he can make his trip as shorter as possible. Soon, he realized that the best way is to pass on every street once and only once and he should finish his trip from the place where he started – his parents ’house. The streets are uniquely numbered from 1 to n, n<1995. The intersections are numbered from 1 to m, m<=44. No intersection connects more than 44 streets. Every intersection is uniquely numbered. Each street is determined by exactly 2 intersections. If there are more than one possible route, he has to choose the shortest one, from lexicographically point of view. All the streets have double ways, there are not isolated streets, but the streets are very narrow, and Johnny can’t turn the car on the street, just in intersections. Help Johnny to find the best route for his trip. He leaves at intersection 0. Use a graph to solve the problem. The input data will be read from the console or from a file and the proposed route will be written at the console. If there are no solutions, a message will be displayed to Johnny.

3. Ancient alphabet decoder (3pts – 0.5 pts for working with files, 2.5 pts for topological sort)

Sydney Fox (world famous teacher of ancient history) and her assistant, Nigel Bailey, found a book written in an unknown language, but using the same letters as the English alphabet - no uppercase, only lowercase (example a, b, c ....). The book contains a brief index, but the order of the index words is different from the English alphabet. The treasure hunters attempted to use this index to determine the order of characters in the unknown language but failed. Write a program to help treasure hunters to determine the order of characters in the unknown language and display this order in a file. You must use topological sorting in graphs.

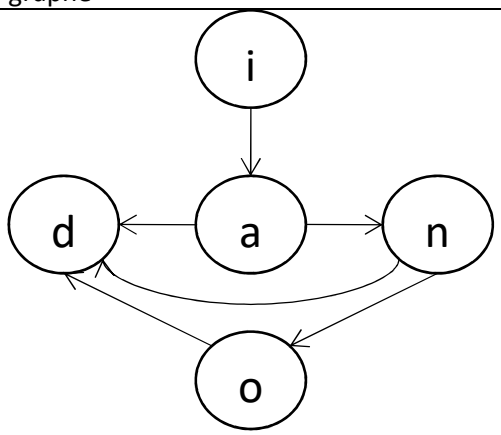
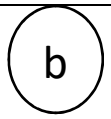
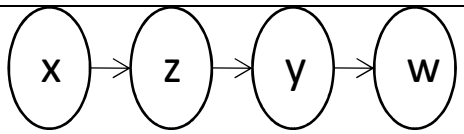
Input data: index.in

In the file "index.in" there is at least 1 and at most 50 words followed by a line containing the character "." (point). Each word has a maximum of 10 characters. The words of an index contain only the lowercase characters of the English alphabet and appear in ascending order of the unknown alphabet. Not necessarily all lowercase letters appear in the index, but an index will result in a unique sequence of characters that appear.

The output data: index.out

In the file "index.out" we write the unique sequence of characters that appear in "index.in".

Example:

index.in	graphe	index.out
ion ana adonia doina doinn ddan ddao .		ianod
b .		b
xwy zx zxy zxw ywwx .		xzyw

Useful references:

fclose: http://www.cplusplus.com/reference/cstdio/fclose/
fopen: http://www.cplusplus.com/reference/cstdio/fopen/
fprintf: http://www.cplusplus.com/reference/cstdio/fprintf/
fscanf: http://www.cplusplus.com/reference/cstdio/fscanf/
http://www.cplusplus.com/doc/tutorial/files/
http://www.cs.washington.edu/education/courses/cse373/02au/lectures/lecture19l.pdf