COMP 2102 Data Structures and Algorithms
Fall 2021, Project #1
Due Date:14 Nov 2021, 23:59

IŞIK UNIVERSITY
COMPUTER SCIENCE AND ENGINEERING

- You can use either your own or our code from the lab sections for the List implementations.
- **This is a pair (group project) assignment**. Students form groups of two people. Both of them are responsible for the whole project.
- Do not share your code with another group or individual. Grade will be 1 for all parties.
- Do not copy-paste online codes.
- You can discuss with people around, but you cannot get help on code, or blocks of code from anyone, including relatives.
- You are only allowed to use your lecture/lab notes. Violating this rule will result in 1/100 as Project #1 grade.
- You must demonstrate your project on declared dates. Otherwise, your project grade will be 2/100.
- Both partners must be available during the project demonstration time. One (random) person from the pair (group) will be asked questions. The group grade depends on the individual's answer. It is not possible to select or change the person called for demo. It is not possible to help the person in the demo. It is not possible to explain an unanswered question with excuses such as "My friend coded these lines, I do not know them".
- If demo student fails to answer half or more than half of the questions, your project grade will be 1.
- Late submissions will not be accepted!

Problem 1 Loose the letter and table

1. **N** people are sitting around a table to play a game. Each of these people has a name. On each turn, a random number **k** [**1<=k<=N**] is chosen. The **kth** person loses one letter* from his/her name. If a person has no letters left in his/her name, he/she loses the game, and will be removed from the table. The last person who is still in the table, wins the game.
    a. You will implement **circular linked list based** program to simulate this scenario.
    b. The names are chosen randomly from the class.txt. Use only the first names. If you like, you can limit the characters in a name with 5 or 6 to give equal chance to all names.
    c. Your program must display the value of **k** in each turn**, the selected student, and the whole list of names** (put an indicator to one who just lost a letter from his/her name)
    d. At the end, program displays the winner.

    *You can use all String and File (Scanner) library functions.

Problem 2 Student message transfer line

2. Each student in a line can pass a message (a string) to any other student after him/her. When a message request arrives to the first student (**source**), it can pass the message to other two /three students that are placed in the **kth** position next/previous to him/her. Each non-source student in the message line can only pass the message one step forward/backward. However, if the message arrives at a hub-student, he-she can reset the reach of the message to **k** students. Hub-students do not extend every message they receive; instead, they extend only the messages they are interested in. They are interested in messages which contain at least 2 letters of their names. If at least 2 letters in the message and their names match, they give a boost to the message, i.e., re-initiate the message line. You will write a program to simulate this scenario where the student message line is

modelled as a **doubly linked list.** You must create appropriate container (Node) class to include Students.

a) The list is generated randomly with 30 students.
b) Names should be chosen randomly from a txt file, e.g. class.txt
c) **M** students will be chosen randomly as hub-student.
d) The client chooses a name from the list and sends a String message.
e) The direction of message transfer is selected randomly either as forward or backwards at the source student (node). Once the direction is set, it does not change until message arrives to the final student.
f) The message can be passed to at most **k** students linked to these students.
g) If one of the **k** students is a hub and the message has at least two letters in common with his/her name, he/she extends the message to other **k** students in forward or backwards directions.
h) The program may have the following interface. You are free to change the interface.

```
…Reading class.txt
Enter the number of hubs (p): 5
…Creating a linked list of 30 students , 5 are hubs
Enter the number of students to pass the message (k):3
---Displaying the student message train,  (hubs have a * character on their names):
Berk←→Alp*←→Mert←→Mercan←→Burak*←→…←→Serdar←→Selin←→Emre←→Dilara

Enter the message: Hello
Enter the source student: Mert
Mert passes message Hello in forward direction: Mercan, Burak*, Ali
Enter the message: Break
Enter the source student: Mert

Mert passes message Break to forward: Mercan, Burak*, Ali,  Ayse, Metin
```

*Note that it is possible model Student as a recursive message passer.