BIRZEIT UNIVERSITY

Department of Computer Science

COMP2421 (Second Semester – Spring 2021/2022)

Project#3 Due Date: 28 May 2022 @11:00 PM

In this project, you will maintain the information of different courses using **Hashing**. Your program will read the courses and their relevant information from a file named *offered_courses.txt*. As well, the user should be able also to enter new courses into the program with their relevant information.

Please use the following format for inputs
Course:CreditHours#CourseCode#Department/topic1, topic2, …, topicN

where N is the number of topics covered in the course and it is ≥ 1.

Example of input courses:

```
Course:CreditHours#CourseCode/topic1, topic2, …, topicN

Data structures:4#COMP2421#Computer Science/recursion, time analysis, linked lists, stacks, queues, trees, bst, avl, splay, b_trees, hash, heaps, sorting, graphs

Introduction to programming:3#COMP133#Computer Science/algorithms, introduction to c, functions, selection, loops, pointers, arrays, structs

Introduction to French:3#FREN111#French Language/letters, numbers, greetings, grammars, statements
```

You will have to create a hash table to store the courses' information (use course name as a key).

You should implement hashing in two different methods to resolve collisions using open-addressing and double hashing and generate 2 different hash tables each using the different methods.

The following options should be available for the user. For each option, you should provide the related information for both hash tables you have created:

1. Print hashed tables (including empty spots).
2. Print out table size and the load factor.
3. Print out the used hash functions.
4. Insert a new record to hash table (insertion will be done on both hash tables).
5. Search for a specific word (specify which table to search in).
6. Delete a specific record (from both tables).
7. Compare between the two methods in terms of number of collisions occurred.
8. Save hash table back to a file named *saved_courses.txt* (of the double hashing)

Grading policy:

1. Your application should have all functionalities working properly. **Twenty** marks will be graded for the functionality of the project;
2. The following notes will make up the remaining 10 marks of the grade:
    a. There has to be adequate documentation and comments in the code (i.e., functions, loops, etc.);
    b. Your code should follow the code convention (i.e., spaces, indentations, etc.); and

c. Your application should contain a menu to allow the user to select which option (s) he would like to run.

Notes and submission instructions:

1. **This is individual work**. It should represent your own efforts. It is fine to discuss your work and to ask your colleagues, but you are not allowed to copy/paste the work of others or give your work to anyone else. You are not allowed to post/copy from other websites and/or social media and this will be considered as cheating.
2. Any **plagiarized** code will not be marked.
3. **Document format**. Please submit only the code file (**c** file) containing the code of your project. Please rename it as follows:
   "**P3_YourStudentID_FirstNameLastName_SectionNo.c**".
4. **Input/output file name**. Make sure that the input/output file names are the same as in the specifications.
5. Include your full name, student ID, and section number in the beginning of your file.
6. Please do not compress the file, only the C-file is needed.
7. Files not following the convention in point 2 will not be marked.