Object Oriented Programming

Assignment 01

Due: Sunday, Feb 24, 2019

Instructions

You are free to consult each other for verbal help. However **copying or sharing the code with each other will not only result into the cancellation of the current assignment, it may impact your grade in all the assignments and exams as well**.

Question #01

Define a struct appropriate for holding location information. At minimum, this should include a Location Number (an integer), ID (a string containing no more than 15 characters), a Description (a string containing no more than 50 characters) and a Latitude and Longitude (both floating point values).

Ask the user for the number of locations, and use this number as the initial size of your LocationArray. (Hint: for easier testing, use a small number, such as two or three.) Allocate an appropriate amount of memory from the heap to create a dynamic array (named LocationArray) of Location structs, large enough to hold the number of locations entered by the user.
Provide a menu that allows the user to choose among the following options:

Add a location to LocationArray - This will prompt the user to enter information about the location (ID, Description, Lat/Lon), and save this in the next uninitialized element in LocationArray.

Print the current list of locations - This will print all elements of each initialized Location struct)

Delete a location from LocationArray - This will prompt the user to enter location ID.

Edit a location in LocationArray: This will prompt the user to enter location ID to be edited.

Search a location in LocationArray: This will prompt the user to enter location ID to be searched.

Quit the program

Create a function called "ResizeArray" to be used whenever the number of locations to be added to LocationArray would exceed the current bounds of the array. The user should not be aware that the size of the array is changing. Rather, s/he should simply be allowed to keep adding locations until s/he is done, and ResizeArray should be called (transparently to the user) whenever the number of locations to be added would exceed the bounds of the array so that the user may add as many locations as s/he likes. Each call to ResizeArray should increase the size of the existing LocationArray. Make sure you test your function by adding more locations than

originally requested at the start of your program.

Be sure to include comments within your code that explain in high-level terms what the various parts of your code are doing.

Question #02

Write a program that calculates sum of array elements where array elements can be accessed using a pointer to an array?

Question #03

Define an integer pointer array of 10 numbers. Initialize them to any integer values from the key board. Find the sum and average of these 10 integers.

Question# 04

Create an integer 3-diemensional array of size 4*5*10 dynamically. Take input from the user using pointer notation. After displaying all elements of the array deallocate the memory.

Question #05

The seven most commonly used functions in the string library are:

1. strcat - concatenate two strings

   char *strcat(char * s1, const char * s2);

2. strcmp - compare two strings

   int strcmp(const char *s1, const char *s2);

3. strcpy - copy a string

   char *strcpy(char * s1, const char *s2);

4. strlen - get string length

int strlen (const char *s);

5. strncat - concatenate one string with part of another

   char *strcat(char * s1, const char * s2,int n);

6. strncmp - compare parts of two strings

   int strcmp(const char *s1, const char *s2,int n);

7. strncpy - copy part of a string

    char * strcpy(char * s1, const char *s2, int n);

Provide definition of the above functions to use it without including any library. The input of the above functions will be char array.

Question #06

You are required to design and implement a small student's database. The system is supposed to maintain a record of students enrolled at a college/university. Following information is stored for each student.

- student_id of type integer
- student_name of type string

. Date of birth of type Date(user defined data type)

- gender of type character
- semester of type integer
- department of type string
- gpa of type float.
- extra of enumerated data type extracurricularSkill

(Note: ExtracurricularSkill can have the following possible values: SocietyMember, Sportsman, InternetGeek and None.)

The program should enable its user to perform the following operations.

1. Declare/define the data structure for the given problem.

2. Assign dynamic memory for an initial list of five students. Initially, store data for ten students, using assignment statements.

3. Display the information of all the students on campus, in the tabular form.

4. Update the information of an old student (e.g., change his date of gpa, semester and etc.)

5. Add the information for a new student. The program should be able to handle all the possibilities (i.e., insert at the start, end or anywhere).

6. Search and display the information of a particular student on campus. (The student information will be searched by giving the full student name. In case, no student with that name is found, an error message will be displayed on the screen.)

7. Search and Display the information of all the students of a particular semester and a particular department, in the tabular form.

8. Search and Delete the information of an old student from the record, using the student_id. (If no record entered so far, display an error message). The program should be able to handle all the possibilities (i.e., delete from the start, end or anywhere).

9. Search for duplicated entries in the database. Once a duplicated entry is found, Delete one of the entries from the database. (Note: In order to verify this function, you must have some duplicated entries in the database).

10. Suppose the total student community can be divided into three groups

- Nerds
- Vibrants
- Dumbs

The Nerds group will be composed of students with GPA more than 3.5 and having extracurricularSkill IntenetGeek or none. The Vibrants will be the group of people with GPA more than 3.0 and having any extracurricularSkill other than none. The Dumbs will be the group of people with GPA less than 2.0 and having extracurricularSkill none.

Your job is to read the original list of students and extract the above three groups of students and move them to three new lists, Nerds, Vibrants, and Dumbs, respectively. After this operation the original list of students should be left with only those students which belong to none of these groups. (Note: In order to verify this function, you must have data for each group in the original database).

Further Instructions

- The number of students in the record is dynamic. For the possible demo / viva voce, your program should already have at least five student records stored in it.
- If the current size of the dynamic array is not enough to accommodate the Add operation, you have to increase the size of the dynamic array. Whenever the array size has to be increased, it can be done in one of the following ways
  - Increase by a fixed amount, e.g., increment by 10. □ 10, 20, 30, 40, …..
  - Double the current size of the array. □ 10, 20, 40, 80, …..
- You are required to make a function for each possible operation performed on the list. The operations should be executed with the help of an interactive user interface (e.g., a menu showing all the possible operations).
- Example of the tabular form data display is as follows

ID	Name	Gender	Semester	Department	GPA	Extracurricular Skill
20120001	Ammir	M	8	FCSE	3.0	Sportsman
20120001	Bilal	M	7	FME	3.5	None
. . .						
. . .						
20120001	Zain	M	6	FEE	2.0	InternetGeek

Question #07

Create two matrices of user defined size dynamically. Perform the following operations on matrices.

- Addition
- Subtraction
- Multiplication

For each operation write a separate function.

Question #08

Write functions which will sort an integer array.

- Bubble Sort
- Selection Sort
- Insertion Sort

Best of Luck!