Algorithms and data structures

Labwork # 2: Linked Lists

Please use Google classroom for labwork submission; join the class by the following link: https://classroom.google.com/c/NTQ1NjMwOTMwNzU0?cjc=gqtgfva

There are 6 Labworks in this course. After each lab work session:

- You need to complete one of the given exercises and upload your files to the assignment ”Labwork No. - Group No. - Version 1 (Attendance)”. Submission must be done within 30 mins after the lab; otherwise, it will be considered a late submission.
- You will have one week (or 7 days) to complete the remaining exercises and upload your files to the assignment ”Labwork No. - Group No. - Version 2 (Complete)”
- Compress all code source files in a zip file and rename it as FULLNAME-ID-Lab#no.zip (e.g NguyenVanA-070-Lab1.zip). Save your files according to the exercise number i.e Ex1.cpp, Ex2.c, etc. Incorrect filenames will result in no score for the respective exercises.
- Only code source files (.c or .cpp) should be in the zip files. Other files (.exe, .o) MUST be removed from the zip file.
- Copy/Paste from any source is not tolerated. The penalty will be applied for a late submission.

NOTE: You must follow the guide. Incorrect zip file names, zip files containing other files (.exe), and copy/pasting lead to heavy penalties.

There are 6 Labworks in this course. After each lab work session:

- You need to complete one of the given exercises and upload your files to the assignment ”GroupNo.-LabworkNo.-First Version”. Submission must be done within 30 mins after the labwork, otherwise, it will be considered as a late submission.
- You will have one week (or 7 days) to complete the remaining exercises and upload your files to the assignment ”GroupNo.-LabworkNo.-Complete Version”
- Compress all code source files in a zip file and rename it as FULLNAME-ID-Lab#no.zip (e.g NguyenVanA-070-Lab1.zip). Save your files according to the exercise number i.e Ex1.cpp, Ex2.c, etc. Incorrect filenames will result in no score for the respective exercises.

- Only code source files (.c or .cpp) should be in the zip files. Other files (.exe, .o) MUST be removed from the zip file.
- Copy/Paste from any source is not tolerated. The penalty will be applied for a late submission.

NOTE: You must follow the guide. Incorrect zip file names, zip files containing other files (.exe), and copy/pasting lead to heavy penalties.

Exercise 1: Suppose that a polynomial function $a_0 + a_1x + a_2x^2 + ... + a_nx^n$ Using List ADT, we would like to improve the computation of this function. The declaration of each element in the List should be as follows:

- a value stands for a constant of each term a_i $(i = 0, .., n)$
- a degree indicates the degree of each term

Implement and test the program in C using a Static Array-based List to manage this polynomial function:

- add new terms, verify that the old term exists, if in the case, return the addition between the old and new ones i.e given the polynomial function $a_0x^0 + a_1x^1$, a new term is added into the function then the final term should be $a_0^{total} = a_0^{new} + a_0$
- remove a term from the function
- enter a value for x then calculate the whole function
- display the whole function on the screen

Exercise 2: Assume that a railway train has N railroad cars attached together such as $car_1, car_2, car_3, ..., car_N$. A train can be considered as a list and each car corresponds to a node in this list.

- Each car carries a number of passengers (int type) and has a name (char type). Both variables are user-defined values.
- If there are any cars that don't have any passengers, they should be removed from the train.
- It is possible to add new cars to the train.

- A function is available to display all cars' information or the length of the train.

Implement a program in C using Linked List to manage the train and test all functions.