University of Science and Technology of Hanoi
Address: USTH Building, 18 Hoang Quoc
Viet, Cau Giay, Hanoi

Algorithms and data structures

Labwork # 2: Linked Lists

Follow the below guide:

- After a labwork, you will have one week (or 7 days) to complete all exercises. All submissions must be sent before 23:59 of the day before the next labwork day.
- Compress all code source files in a zip file and rename it as FULLNAME-ID-TT#no.zip (e.g NguyenVanA-070-TT1.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.
- Send to this email: doan-nhat.quang@usth.edu.vn
- Copy/Paste from any source is not tolerated. Penalty will be applied for late submission.
- NOTE: You must follow the guide. Incorrect zip file name, zip files containing other files (.exe), copy/paste lead to heavy penalty.

Exercise 1: Assume that a railway train has N railroad cars attached together such as: $car_1, car_2, car_3, ..., car_N$.

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

Exercise 2: Suppose that a polynomial function $a_0 + a_1x + a_2x^2 + ... + a_nx^n$ Using Linked Lists, we would like to improve the computation of this function. The declaration of each node should be as following:

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

Implement and test the program in C using Linked 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 in the screen