Middle East Technical University Department of Computer Engineering

CENG 301

Algorithms and Data Structures

Fall 2020-2021

Programming Assignment 1

Due date: 25.11.2020, Wednesday, 23:55

1 Introduction

In this assignment you are going to practice on linked list implementation in C++. The details of the structure is explained further in the following sections.

Keywords: *Linked list, C++*

2 Problem Definition

A bus company wants a computer program that can arrange its weekly services from Monday to Sunday at a fixed time interval(start time and finish time of the day) and each bus leaving one hour after the departure of the previous bus.

The bus company will provide the following information as input to the computer program:

- Start time and end time of working hours in a day
- The license plate number of the buses
- The drivers' names and surnames

Based on this information, a weekly schedule from Monday to Sunday will be created within the specified working hours, taking into account the order of the buses in the given input.

The program must consider the followings:

- Arrangement of bus schedules according to the order provided by the bus company
- When the number of buses is insufficient to complete a daily schedule, the daily schedule should be completed again, starting with the first bus.
- if the number of buses is more than a day's need, arranging the bus trips by transferring the excess buses to the next day without violating the order of the buses
- In case of a bus malfunctioning, the bus must be removed from the list without violating the list provided by the bus company.
- In case of purchasing a new bus, the bus must be added to the end of the list without violating the list order provided by the company.

3 Specifications

- You will be implementing some functions (marked with "TODO") of a linked list. The bare header file, "busSchedule.h", is given below. You are not allowed to change anything in the implemented functions. **Note that you are expected to implement "readInput", "scheduleBusesWeekly", "printWeeklySchedule", "addBusNode", "deleteBusNode" and "listAllBus" functions.**
- Your main duty is implementing the functions (marked by "TODO") in "busSchedule.h"
- You are given a sample main.cpp for testing purpose. Your code will not be graded with these test cases. However, test cases will be similar to the given ones.

- Any incorrect input will not be given to your functions.
- Do not change the implementations of the already implemented functions.
- You can write helper functions if necessary. Moreover, you can also define variables.

Function	Parameters	Return	Definition
BusNode * **readInput**(int * start, int * finish)	**(int * start, int * finish):** start and finish time are given as parameters to the readInput	Head pointer of the linked list of the buses	This function reads the input provided by the bus company and forms the linked list of the busses
BusNode * **scheduleBusesWeekly**(BusNode *, int, int)	**(BusNode * head, int start, int finish):** head pointer to the list of the busses, start time and finish time of the transportation are given as parameters	This function returns a BusNode pointer as the head of the weekly schedule	This function forms a linked list of the weekly schedule of the busses
void **printWeeklySchedule**(BusNode *headSchedule, int start, int finish)	**(BusNode *headSchedule, int start, int finish)** : headschedule is the pointer to the weekly schedule of the busses, start time and finish time of the transportation are given as parameters	No return value	This function prints the weekly schedule
void **addBusNode**(BusNode * head, string number, string name, string surname)	**(BusNode * head, string number, string name, string surname):** the head is the pointer to the linked list of the busses, the number is the license plate number of the bus, the name is the driver's name, and surname is the driver's surname	No return value	In case of a new bus purchasing, this function adds the new bus to the end of the list provided by the bus company
void **deleteBusNode**(BusNode * head, string number)	**(BusNode * head, string number):** the head is the pointer to the list of the busses, the number is the license plate number of the bus,	No return value	In case of a bus malfunctioning, this function deletes a bus from the list provided by the bus company according to the given license plate number.
void **listAllBus**(BusNode *head)	**(BusNode *head):** the head is the pointer to the linked list of the busses.	No return value	This function lists the buses in the company's inventory in the order they are given.

```
#include<iostream>
using namespace std;
typedef struct BusNode {
  string number;
  string drivername;
  string driversurname;
  BusNode *next;
} BusNode;

BusNode * readInput(int * start, int * finish){
  BusNode * head = new BusNode();
  //TODO
  //This function reads the input file and
  //forms a linked list of the buses in the given order
  //returns the pointer to the first bus(BusNode)
  //HINT: head is a dummy node whose next is the first BusNode in the linke list

  return head;

}
BusNode * scheduleBusesWeekly(BusNode *head, int start, int finish){
  BusNode *headSchedule = new BusNode();
  //TODO
  //This function forms the weekly schedule as linked list
  //head is the pointing node to the first bus in the linked list of buses
  //start is starting time of the buses for instance 6(implies 6:00)
  //finish is finishing time of buses for instance 22(implies 22:00)
  //returns headSchedule is a dummy node whose next is the first Bus in the weekly schedule
  return headSchedule;

}
void printWeeklySchedule(BusNode *headSchedule, int start, int finish){
  string weekdays[7] = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "↩
      Sunday"};
  //TODO
  //This function prints the weekly schedule as illustrated in the output file
  //headSchedule which is the node pointing to the first Bus in the weekly schedule
  //start is starting time of the buses for instance 6(implies 6:00)
  //finish is finishing time of buses for instance 22(implies 22:00)

}
void addBusNode(BusNode * head, string number, string name, string surname){
  //TODO
  //This function inserts a new bus to the linked list of the buses
  //The parameter head is the pointing node to the first bus in the linked list of buses
  //number is license plate number of the bus
  //name is driver's name
  //surname is driver's surname

}
void deleteBusNode(BusNode * head, string number){
  //TODO
  //This function deletes a malfunctioning bus from the linked list of the buses
  //The parameter head is the pointing node to the first bus in the linked list of buses
  //number is license plate number of the bus

}
void listAllBus(BusNode *head){
  //TODO
  //This function lists all buses in the linked list of the buses
  //The parameter head is the pointing node to the first bus in the linked list of buses

}
```

4 Execution

4.1 Windows

1. You should install gcc compiler for Windows in order to execute the files. You can install gcc by following the instructions at `https://www.youtube.com/watch?v=QonIPpKodCw`
2. Open cmd and move to the directory of the linked list files with cd command. For instance if files are located at Desktop, you should execute "cd C:\Users\username\Desktop" command on cmd.
3. After moving to the correct directory, execute the following command (on cmd) to compile your files:

   g++ main.cpp -o main

4. Run the compiled files by executing (on cmd) :

   ./main <input.txt >output.txt

4.2 Linux

1. You can check whether gcc is installed (skip 2) or not by executing the following command on terminal:

   gcc -version

2. If gcc is not installed on your computer, you can install gcc for Linux by typing (on terminal):

   sudo apt-get install build-essential

3. Open the terminal and move to the directory which contains linked list files using cd command. For instance, if files are located at Desktop, you should execute "cd \home\username\Desktop" command on terminal.
4. After moving to the correct directory, execute the following command (on terminal) to compile your files:

   g++ main.cpp -o main

5. Run the compiled files by executing (on terminal) :

   ./main <input.txt >output.txt

5 Regulations

1. You have to write the program in C++
   .
2. If your submission fails to follow the specifications or does not compile, there will be a "significant" penalty of grade reduction.
3. **Late Submission:** Late submission policy is stated in the course syllabus.
4. **Cheating:** We have zero tolerance policy for cheating. In case of cheating, all parts involved (source(s) and receiver(s)) get zero. People involved in cheating will be punished according to the university regulations.
5. **Newsgroup:** You must follow the announcements (in Odtuclass) for discussions and possible updates on a daily basis.

6 Submission

- Submission will be done via Odtuclass (odtuclass.metu.edu.tr).
- Do not modify already implemented functions in "busSchedule.h"
- You need to submit "busSchedule.h" only.
- Do not submit a main function.
- Additional test cases may be used for evaluation.
- Only the last submission before the deadline will be graded.