**CSC20 HW3 Train of Cars Sp2021**

You are the manager of the train depot. One of your tasks is to efficiently order the train cars so that it is easy to attach and detach them. In this assignment, you will implement Train Cars program that reads car details from the file called car.txt (given) and stores them in a LinkedList data structure.

Cars in the train are linked in a specific order. Each car in the train contains various materials that must be attached and detached in the most efficient manner possible. Each car ( node of type Car) has a stop number, factory name, material name.

* + - **Stop number** – determines which stop the car must be detached, use this to order the train cars. They should be ordered either in increasing or decreasing stop numbers.
    - **Factory Name** – name of the factory at which the car must be detached
    - **Material Name** – name of the material being transported in the car.

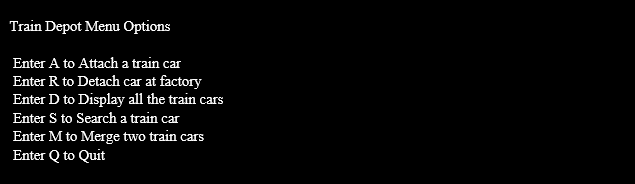
For example, the Jiffy Mix plant at stop 2 needs sugar, flour, cornmeal, etc. Write the program to perform each operation specified to get materials from source to destination efficiently.

The train needs to be created as a linked list which has a car at each node.

Your program needs to be menu based and have the following capabilities:

* + - Display all the train car details
    - Attach a train car (given Factory Name, Stop Number, Material Name)
    - Detach cars at factory (given Factory Name)
    - Search a train car (given Factory Name)
    - Merge two train cars (given update.txt)

Here is what the menu looks like



**Steps to follow**

1. You will need to write the class *Car.java* that is used to create cars. The shell is given below:

class Car {

     public String factory;

     public int stop;

      public String material;

Car next;

public Car(){

// default constructor

}

     public Car(String fact, int s, String m){

     // regular constructor sets next to null

     }

public Car(String fact, int s, String m, Car next){

// regular constructor sets   this.next=next;

     }

   }

1. You will need to write the class *Train.java* that creates the linked list and offers methods to manage the linked list. It should have one variable called *front* of type Car that points to the beginning of the linked list. It should have two constructors, a default and a regular constructor. The regular constructor reads the car details from the file “car.txt” and creates the train.
2. The class Train.java creates and manages a linked list of cats and should implement the following methods:

public Train(String carFile) throws FileNotFoundException

//constructor: reads each train car details from the file into the LinkedList

public void detach(String factoryName)

//remove a car given factoryName, this should delete the node in the LinkedList.

public void attach(String factoryName, int stopNumber, String materialName)

// add a new car(node) to the LinkedList

public void search(String factoryName)

**//** search for cars that correspond the given factory name. Note there may be

// more than one car so you may use helper method that follows

public List<String> getCars(String factoryName)

// material name lookup given factory name

public void displayTrainCars()

// display all the train cars in sorted order

public void merge(String update)

// reads from update.txt and merges it with the new train car. Merge two LinkedLists and should maintain the sorted order.

1. Read the given text file *car.txt* and split the input strings into 3 tokens using

*String tokens[]= inline.split(",");*

*tokens[0]* is Factory Name,

*tokens[1]* is Stop Number,

*tokens[2] is Material Name* as a single string.

The tokens[] should be passed to the LinkedList.

1. When the node is added to the LinkedList, it should always be in the sorted order, i.e., if the stop number being added is less than the existing stop numbers in the LinkedList, add the node to the head of the LinkedList. if the stop number being added is greater than the existing stop numbers in the LinkedList, add the node to the tail of the LinkedList. Otherwise, add the node in the appropriate position of the stop number.

**Here is what the User Interface for the Train Cars should look like:**

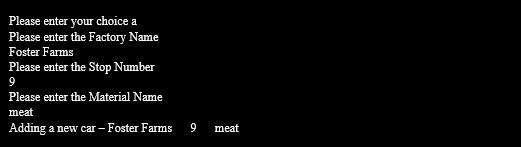
Main menu: Should accept upper and lower case:

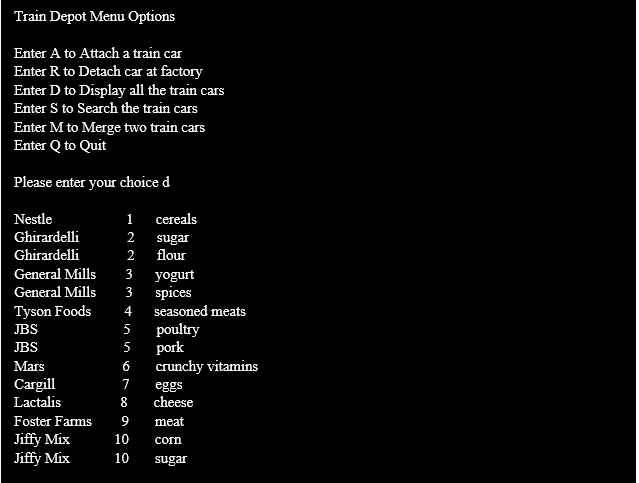
**Option d: display all car details**



**Option a: attach a car**

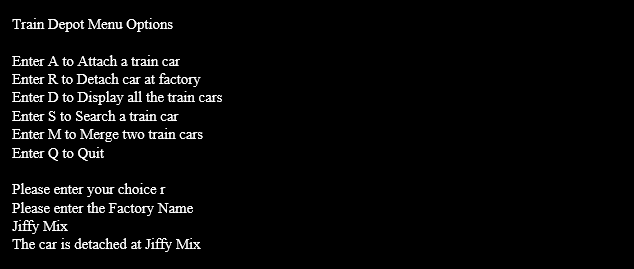
*Note: when you attach the car, the car should be placed in the sorted order*.



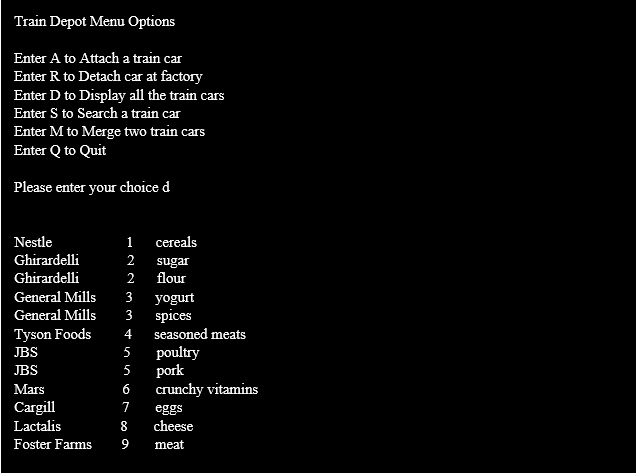


**Option r: detach car at factory**

1. Given a factory name, all cars belonging to that factory must be detached



After detaching the car, the display should look like

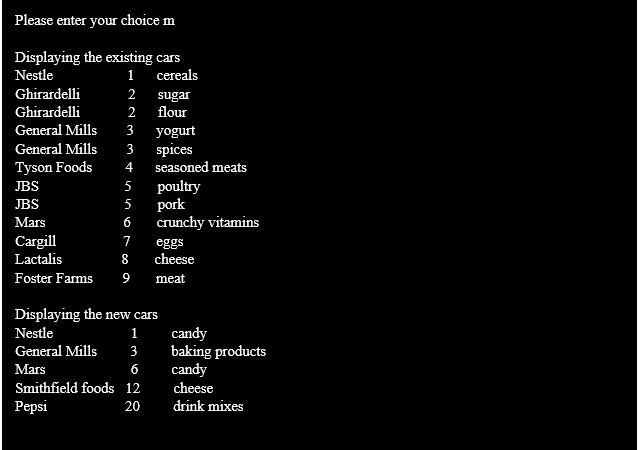


**Option s: search a train car**

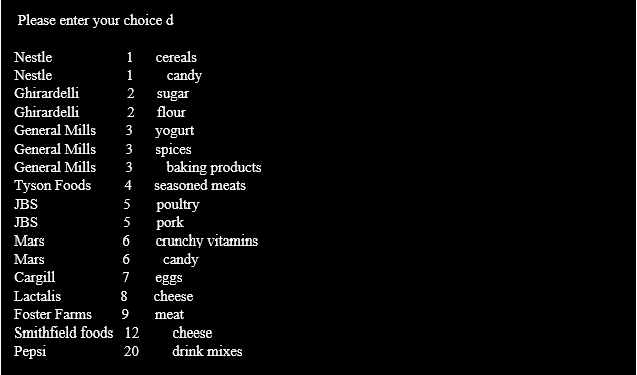
1. Given a factory name, all the materials belonging to the factory must be displayed

**Option m: merge two train cars**

*Note: while merging the cars the order should be maintained.*



Displaying the merged car details



**Option q: quit**

