National University of Computer and Emerging Sciences, Lahore Campus

	Course:	**Object Oriented Programming**	**Course Code:**	**CS1004**
	Program:	**BS(Computer Science)**	**Semester:**	**Fall 2023**
	Topic:	**OOP concepts covered so far**	**Total Marks:**	
	Due Date:		**Weight**	**TBD**
	Section:	**CS-3A and CS-3B**	**Page(s):**	**4**
	Exam:	**Assignment 2**	**Reg. No**	

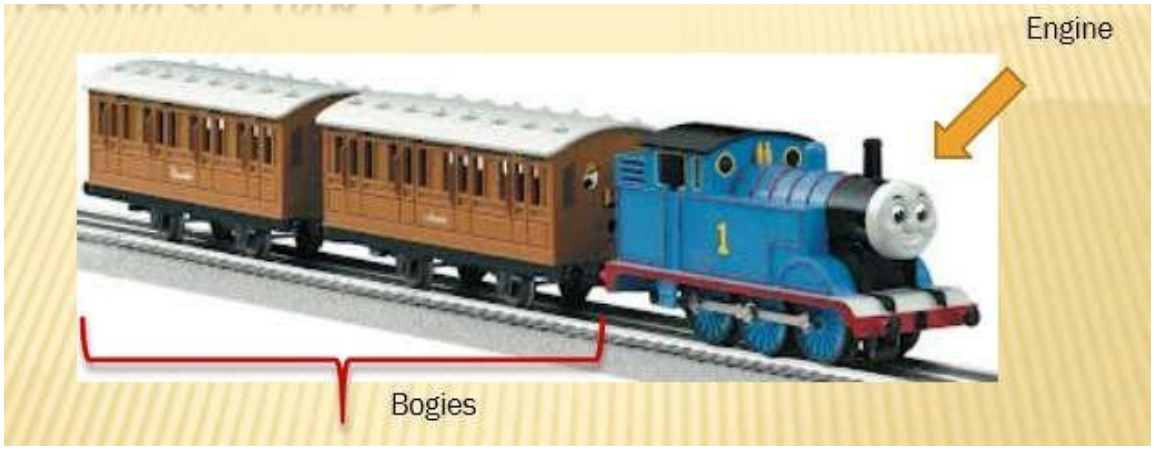
Submission Folder:

On Google Classroom

Instructions:

1. Submit your running files **(.CPP only)** carefully. No updated files will be accepted after the deadline.
2. You can take help from each other but copy/plagiarism in any case will not be tolerated.
3. You have to build a console application for this assignment.
4. You can't use any other built-in data structure except the arrays.
5. Your code must not have any dangling pointers and memory leaks.
6. Follow proper and acceptable procedures for writing the code e.g. make your code properly indent, give suitable names (identifier) to your classes, objects and variables etc.

Train



A train is a form of transport consisting of a series of connected ***vehicles/bogies*** that generally runs along a railroad track to transport cargo or passengers. The word "train" comes from the Old French trainer, derived from the Latin trahere meaning "to pull" or "to draw".

A passenger train includes many passenger-carrying vehicles/bogies that are connected together and can often be very long and fast. Every bogie has a unique identification number.

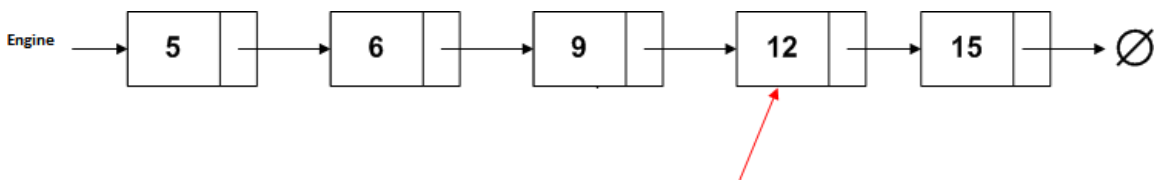
Every train has an engine (special form of bogie).

Now you have to implement the model of the train such that the engine is the starting point of the train. Engine contains the address of the first bogie (connected to it). Initially when no bogie is attached Engine is NULL.

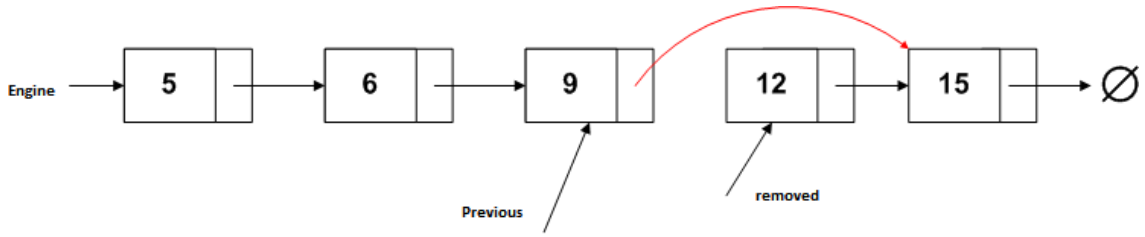
Similarly each bogie will contain the address of the next bogie attached to it. Every new bogie will be attached at the end of the train only.

We can simply detach any bogie (based on `Bogie_ID`) by placing the address of the next bogie in the address part of the previous bogie.

Suppose you have the following train and you want to remove the bogie 12.



After removal your train will look like this



Now you can safely delete bogie 12.

Keep in mind you cannot directly access any bogie. Your traversal should start from the engine.

Each bogie can accommodate only 10 people (4 adults and 6 kids) at a time (maximum). Only one family can reserve one bogie. Your system should be able to print the bogie number and following information.

Adult	Kids
Name Age Gender Occupation Qualification NIC number	Name Age Gender B-form number

Partially filled classes are given to you. You can add any other data member or function or any other class. You have to implement all the functionality.

Your system should be able to maintain the data structure of the train. You are not allowed to use any built in data structure of C++.

```
class Person {
	string Name;
	int    Age;
	char Gender;

};
class Adult :public Person {
	string Occupation;
	string Qualification;
	string NIC;

};
class kid :public Person {
	string B_form_number;
};
```

```
class Bogie {
	int    Bogie_ID;
	Bogie *next;
	Person * Adults;
	Person * kids;
	string familyName;
public:
	Bogie(int id);
	bool AddPassengers(); // should add adults and kids information etc
	void Print();
	~Bogie();

};

class train {
	Bogie * engine;
public:
	train() {
		engine = nullptr;
	}
	void addBogie(int ID); //add bogie at the end of the train
	bool removeBogie(int ID); //search the bogie and delete it
	void printTrain(); //print only the Bogie_ID of all bogies
	bool SearchBogie(int ID);//search the particular bogie
	~train();
};
```