**Exercise No. 1**

Design a Date class, with year month and day as data members and overload the following operators as:

* **++ Pre x and post x increment operators.** These operators should increment the day only.
* **-- Pre x and post x decrement operators.** These operators should decrement the day only.
* **<<cout stream insertion operator.** This operator should cause the date to be displayed in the form April 18, 2012
* **>>cin s stream extraction operator.** This operator should enter a date to be stored in a Date object.

The class should detect the following conditions and handle them accordingly:

* When a date is set to the last day of the month and incremented, it should become the first day of the following month.
* When a date is set to December 31 and incremented, it should become January 1 of the following year.
* When a day is set to the first day of the month and decremented, it should become the last day of the previous month.
* When a date is set to January 1 and decremented, it should become December 31 of the previous year.
* Your program should support the following formats of date:
  + DD-MM-YYYY
  + MM-DD-YYYY
  + YYYY-MM-DD
  + DD-MMM-YYYY
* Proper error check should be done;
  + User should not be able to enter year later than the present year.
  + User should not be able to enter invalid month or year.
  + User should not be able to enter 31st of AUG.
  + Cater the concept of leap year (occurring once every four years, which has 366 days including February 29).

Demonstrate the class Date capabilities in a simple program.

**Exercise No. 2**

Implement **Matrixtype** classfor the given **driver.cpp** file.

**Exercise No. 3**

**Implement the following class and write a menu driven main to test you program. (Make sure that denum can never be zero)**

Implement the following Fraction class and over the described operator.

class fraction

{

int num, denum;

public:

fraction(int = 0, int = 1); //Constructor

void operator!(void) const; // print the fraction

fraction& operator~(void); // reduce the fraction

fraction operator-(void) const; // negative of fraction

fraction operator\*(void) const; // reciprocal of fraction

//Fraction’s Arithematic Operators

fraction& operator+=(constfraction&); //frac1 +=frac2

fraction& operator-=(constfraction&); //frac1 -=frac2

fraction& operator\*=(constfraction&); //frac1 \*=frac2

fraction& operator/=(constfraction&); //frac1 /=frac2

// Fraction’s Relational Operators.

bool operator>(constfraction&) const;

bool operator<(constfraction&) const;

bool operator>=(constfraction&) const;

bool operator<=(constfraction&) const;

bool operator==(constfraction&) const;

bool operator!=(constfraction&) const;

fraction& operator++(); // prefix operator returns by reference

fraction operator++(int); // postix operator returns by value

};

class fraction

{

**POST LAB: (Submission deadline 28th March, 2017)**

1. Write a class named as “**Car**” having following Attributes:

● ‘**make’** of type cstring

**● ‘car\_model’** of type cstring

● ‘**reg\_no’** of type cstring

● ‘**color’** of type cstring

● ‘**year’** of type int

2. Now write class “Garage” of your GMS having the following attributes

● **‘name’** of type cstring

● **‘index’** of type int

● **‘capacity’** of type int

● Dynamic Array of objects of **‘car’** class with maximum size equals to ‘capacity’ Specified by owner of the garage.

Now write the header file to make the following Main.cpp code run:

#include<iostream>

#include"Garage.h"

using namespace std;

int main()

{

int Capacity;

cout <<"Please Enter the total capacity of the Garage"<< endl;

cin >> Capacity;

Garage gar(Capacity, "Section A's Garage");

int choice;

Car c;

char \* regno;

regno = new char [10];

while (1) {

cout <<"\n --------------Garage Menu------------ "<< endl

<<" Press 1. To Push the Car"<< endl

<<" Press 2. To Display all the cars parked"<< endl

<<" Press 3. To Remove a Car from Garage"<< endl

<<" Press 4. To Find a Car from Garage"<< endl

<<" Press 5. To Exit "<< endl

<<"\n ------------------------------- "<< endl

<<"Your Choice :";

cin >> choice;

switch (choice) {

case 1:

cin >> c;

gar.PushCar(c);

break;

case 2:

cout << gar;

break;

case 3:

cout <<" Enter the Car's Registration Number to be Removed : ";

cin.getline(regno,10);

gar.RemoveCar(regno);

break;

case 4:

cout <<" Enter the Car's Registration Number to Find : ";

cin >> regno;

gar.FindCar(regno);

break;

case 5:

cout <<" \n Exiting .... "<< endl;

return 0;

default:

cout <<"\n Invalid Option, Please use any of the listed options! "

<< endl;

}

}

}