National University
of computer and emerging sciences

CL-1004
Object Oriented Programming- Lab
Spring' 2023
BS-SE

Lab Manual 06

Problem 1:

Design a class Holiday that represents a holiday during the year. This class has three private data members:

- **name: A string that represents the name of holiday.**
- **day: An integer that holds the day of the month of holiday.**
- **month: A string that holds the month the holiday is in.**

1. Write a default constructor that initializes each data member of class such that name with NULL, day with 0 and month with NULL **Holiday ()**

2. Write a constructor that accepts the arguments for each data member such that string n assigned to name, int d to day and string m to month. **Holiday (string n, int d, string m)**
   Note: Use member function initialization for all data members.

3. Generate getter setter of each member variable: such that name should never be greater than 50 characters, day should never be negative and month should not be greater than 10 characters.
   - **bool setName(string s)**
   - **string getName()**
   - **bool setDay(int u)**
   - **int getDay()**
   - **bool setMonth(string p)**
   - **string getMonth()**

4. Write a function inSameMonth (outside class) which takes two Holiday objects as arguments, compares two objects of the class Holiday, and returns true if they have the same month otherwise false.
   bool inSameMonth (Holiday &a, Holiday &b)

5. Write a function avgDate (outside class) which takes an array of type Holiday and its size as its argument and returns a double value that is the average of the entire day data member in the Holiday array arr. You may assume that the array is full (i.e. does not have any NULL entries).
   double avgDate(Holiday arr[], int size)

Problem 2

Create a class **Rational** for performing arithmetic with fractions.

- **numerator**
- **denominator**

Provide a default and a parameterized constructor that enables an object of this class to be initialized when it is declared. The constructors should store the fraction in reduced form. For example, the fraction 2/4 should be stored in object as 1 in the numerator and 2 in the denominator. In case of 3/4, store 3 in the numerator and 4 in the denominator.

Provide public member functions that perform each of the following tasks:

1. Write getter setter functions for both members.
    - **void setNumerator (int a)**
    - **int getNumerator ()**
    - **void setDenominator (int a)**
    - **int getDenominator ()**

Note: Following functions should be made outside the class.

2. Add two Rational numbers. The result should be stored in reduced form. Two rational numbers a/b and c/d can be added as follows:
   (a/b) + (c/d) = (a*d + c*b) / (b*d)

   Rational addRationalNumber(r1, r2)

3. Multiply two Rational numbers. The result should be stored in reduced form. The product of two rational numbers a/b and c/d can be found as follows:
   (a/b) * (c/d) = (a*c) / (b*d)

   Rational multiRationalNumber(r1, r2)

4. Divide two Rational numbers. The result should be stored in reduced form. Two rational numbers a/b and c/d can be divided as follows:
   (a/b) ÷ (c/d) = (a*d) / (b*c)

   Rational divRationalNumber(r1, r2);
5. Print Rational numbers in the form a/b where a is the numerator and b is the denominator.

Problem 3

Write a class Date that represents a date consisting of a

- **year**
- **month**
- **day**

A Date class should have the following methods:

- **Date()**
- **Date(int year, int month, int day)**
  Constructs a new Date object to represent the given date

- Getter setter
    - **void setDay(int d)**
    - **int getDay()**
    - **void setMonth(int m)**
    - **int getMonth()**

- **void setYear(int y)**
- **int getYear()**

- **void addD(int days)**
  Moves this Date object forward by the given number of days.
  Hint: you should decide on the basis of month and year that given month ends 30,31,28,29 days.

- **void addMD(int month , int days)**
  Moves this Date object forward by the given number of months and days. Months should be within 1 to 12 and days in 1 to 31. For Example Date 2003/12/31and addMD(1,29) =>Date will be 2004/02/29

- **void addWeeks(int weeks)**
  Moves this Date object forward by the given number of seven-day weeks.

- **void subtractM(int days)**
  Moves this Date object backward by the given number of days.
  hint: you should decide on the basis of month and year that given month ends 30,31,28,29 days.

- **void subtractMD(int month , int days)**
  Moves this Date object backward by the given number of months and days.

- **String toString()**
  Returns a String representation of this date in year/month/day order, such as "2006/07/22".

Problem 4

Create a class called **Car** that has private member variables:

- **string make**
- **string model**
- **int year**
- **int mileage**
- **double capacity**

Make default and parameterized constructors.

- **Car()**
- **Car(string mk, string md, int y, double fl)** (if year is 2015, mileage is 12000, if year is 2016, mileage is 15000, if year is 2017, mileage is 18000 and so on)

1. Write getter setter functions for both members.
   - **void setMake (string mk)**
   - **string getMake ()**
   - **void setModel (string md)**
   - **string getModel ()**
   - **void setYear (int y)**
   - **int getYear ()**
   - **void setMileage ()** (if year is 2015, mileage is 12000, if year is 2016, mileage is

15000, if year is 2017, mileage is 18000 and so on)

- **int getMileage ()**
- **void setFuelCapacity (double f)**
- **double getFuelCapacity ()**

2. A function to calculate the fuel efficiency of the car (in miles per gallon), given the car's mileage and the amount of gas used.
3. A function that calculates the age of the car based on its year and the current year.
int getAge(int current)

4. A function that checks if the car needs to be serviced based on its mileage and age, and returns a boolean value indicating whether service is needed.
Hint: If mileage greater than 24,000, service is needed, else not needed.
bool serviceNeeded()
5. A function that checks whether the tank is full or not. If capacity is less than 50, return true, else false.
bool isNotFull()
5. A function that adds a certain liters of fuel to the car's fuel tank. If the fuel tank is already full, the function should return false.
Hint: make use of isNotFull() function. Also add fuel upto tank's capacity i.e if capacity is 45, and amount to add is 7, only add 5 liters to tank.
bool refuelTank(int amount)