CSE ASSIGNMENT (C++)

Problem 1

Question:

> *Create a class Date whose object can store a day, month and year. Include the necessary constructors to initialize the objects and function to display the date in ‘dd/mm/yyyy’ format.*

Code:

```
#include <iostream>
#include <iomanip>

using namespace std;

class Date
{
    private:

        short int date;
        short int month;
        short int year;

    public:

        Date()
        {
            date = 01;
            month = 01;
            year = 2000;
        }

        Date(int a_date, int a_month, int a_year)
        {
            date = a_date;
            month = a_month;
            year = a_year;
        }

        void display();
};

void Date::display()
{
    cout << setw(2) << setfill('0') << right << date;
    cout << "/";
    cout << setw(2) << setfill('0') << right << month;
    cout << "/";
    cout << setw(4) << setfill('0') << right << year;
}

int main()
{
    Date date_default, date(30, 12, 2003);

    date_default.display();

    cout << endl;

    date.display();

    cout << endl;

    return 0;
}
```

Output:

```
01/01/2000
30/12/2003
```

Problem 2

Question:

> *Imagine a tollbooth at a bridge. Cars passing by the booth are expected to pay Rs. 50 as toll fee. Mostly they do, but sometimes a car goes by without paying. The tollbooth keeps track of the number of cars that have gone by, and of the total amount of money collected. Model this tollbooth with a class called TollBooth. The two data items are a type unsigned int to hold the total number of cars, and a type double to hold the total amount of money collected. A constructor initializes both of these to 0. A member function called payingCar() increments the car total and adds 50 to the cash total. Another function, called nonPayingCar(), increments the car total but adds nothing to the cash total. Finally, a member function called dispDetails() displays the two totals.*
> *Make appropriate member functions const. Write a program to test this class. This program should be menu-driven to allow the user to select paying car, or nonpaying car or to print the total cars and total cash collected and finally exit the program.*

Code:

```
#include <iostream>

using namespace std;

class TollBooth
{
private:
    unsigned int car_count = 0;
    double amount_collected = 0;

public:
    TollBooth()
    {
        car_count = 0;
        amount_collected = 0;
    }

    void payingCar()
    {
        car_count += 1;
        amount_collected += 50;
    }

    void nonPayingCar()
    {
        car_count += 1;
    }

    void dispDetails() const
    {
        cout << "Total number of cars counted : " << car_count << endl;
        cout << "Total amount collected : " << amount_collected << endl;
    }
};

int main()
{
    TollBooth booth;
    short int choice = 0;

    do
    {
        cout << "Enter your choice [1, 2, 3]";
        cout << "Entering any other number will result in the program exiting" <<
endl;
        cout << "1. Paying Car" << endl;
        cout << "2. Non-Paying Car" << endl;
        cout << "3. Display Details" << endl;
        cout << " : ";

        cin >> choice;

        if (choice == 1)
        {
            booth.payingCar();
        }

        if (choice == 2)
        {
            booth.nonPayingCar();
        }

        if (choice == 3)
        {
            booth.dispDetails();
        }
    } while (choice == 1 || choice == 2 || choice == 3);

    return 0;
}
```

Output:

```
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
```

```
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 3
Total number of cars counted : 9
Total amount collected : 350
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 3
Total number of cars counted : 11
Total amount collected : 400
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
```

```
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 3
Total number of cars counted : 15
Total amount collected : 450
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 2
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
```

```
 : 1
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 3
Total number of cars counted : 20
Total amount collected : 600
Enter your choice [1, 2, 3]Entering any other number will result in the program
exiting
1. Paying Car
2. Non-Paying Car
3. Display Details
 : 4
```

Problem 3

Question:

> *Write a C++ program to create a class called Complex to implement the following functions on Complex type objects. READ/INPUT, ADD, SUBTRACT, MULTIPLY and PRINT with return the resultant object. Write the program with required constructors, member functions with necessary arguments and return types.*

Code:

```
#include <iostream>

using namespace std;

class Complex
{
private:
    double real;
    double imaginary;

public:
    Complex()
    {
        real = 0;
        imaginary = 0;
    }

    Complex(double real_part, double imaginary_part)
    {
        real = real_part;
        imaginary = imaginary_part;
    }

    double getReal()
    {
        return real;
    }

    double getImaginary()
    {
        return imaginary;
    }

    Complex operator+(const Complex &operand)
    {
        return Complex(real + operand.real, imaginary + operand.imaginary);
    }

    Complex operator-(const Complex &operand)
    {
        return Complex(real - operand.real, imaginary - operand.imaginary);
    }

    Complex operator*(const Complex &operand)
    {
        return Complex((real * operand.real) - (imaginary * operand.imaginary), (real
* operand.imaginary) + (imaginary * operand.real));
    }
};

std::ostream &operator<<(std::ostream &os, Complex &obj)
{
    return os << obj.getReal() << " + (" << obj.getImaginary() << ") i";
}

int main(int argc, char const *argv[])
{
    Complex operand_1(3, 4), operand_2(7, -13), result;

    result = operand_1 + operand_2;

    cout << "Complex addition: " << result << endl;

    result = operand_1 - operand_2;

    cout << "Complex Subtraction: " << result << endl;

    result = operand_1 * operand_2;

    cout << "Complex Multiplication: " << result << endl;

    return 0;
}
```

Output:

```
Complex addition: 10 + (-9) i
Complex Subtraction: -4 + (17) i
Complex Multiplication: 73 + (-11) i
```

Problem 4

Question:

> *Create a class called CarPark that has the members CarRegnno(int), ChargePerHour(int) and ParkingDuration(float). Set the data and show the charges and parked hours of a car based on CarRegnNo. Make two member functions for setting and showing the data. Member function should be called from other functions.*

Code:

```
#include <iostream>

using namespace std;

class CarPark
{
private:
    int CarRegnno;
    int ChargePerHour;
    float ParkingDuration;

public:
    void setData(int reg, int charge, float duration)
    {
        CarRegnno = reg;
        ChargePerHour = charge;
        ParkingDuration = duration;
    }

    void showData()
    {
        cout << "Car Reg.No : " << CarRegnno << endl;
        cout << "Parking Duration : " << ParkingDuration << endl;
        cout << "Charge : " << ChargePerHour * ParkingDuration << endl;
    }
};

int main(int argc, char const *argv[])
{
    int reg, duration, charge = 50;

    cout << "Enter your Car Reg.No : ";
    cin >> reg;

    cout << "How many hours has the car been parked for : ";
    cin >> duration;

    CarPark data;

    data.setData(reg, charge, duration);
    data.showData();

    return 0;
}
```

Output:

```
Enter your Car Reg.No : 1250
How many hours has the car been parked for : 12
Car Reg.No : 1250
Parking Duration : 12
Charge : 600
```

Problem 5

Question:

> *Create a class called TIME that has separate data members for Hours(int), Minutes(int) and Seconds(int). One constructor should initialize this data to zero, and another should initialize it to fixed values. Overload the constructors to do this. Add a member function to display time in 11:59:59 format. Add another member function addTIME to add two objects of type TIME passed as arguments. The main() program should create two initialized Time objects and one that is not initialized. Then it should add two initialized value together, leaving the result in the third Time variable. Finally, it should display the value of this third variable.*

Code:

```
#include <iostream>

using namespace std;

class TIME
{
private:
    int hours;
    int minutes;
    int seconds;

public:
    TIME()
    {
        hours = 0;
        minutes = 0;
        seconds = 0;
    }

    TIME(int hrs, int mins, int secs)
    {
        hours = hrs;
        minutes = mins;
        seconds = secs;
    }

    void display()
    {
        cout << hours << ":" << minutes << ":" << seconds;
    }

    TIME addTIME(const TIME &twoime)
    {
        int hrs = hours + twoime.hours;
        int mins = minutes + twoime.minutes;
        int secs = seconds + twoime.seconds;

        if (secs > 59)
        {
            secs -= 60;
            mins += 1;
        }

        if (mins > 59)
        {
            mins -= 60;
            hrs += 1;
        }

        if (hrs > 24)
        {
            hrs -= 24;
        }

        return TIME(hrs, mins, secs);
    }
};

int main(int argc, char const *argv[])
{
    TIME t1(12, 34, 55), t2(13, 55, 59), result;

    result = t1.addTIME(t2);

    cout << "Time : ";

    result.display();

    cout << endl;
    return 0;
}
```

Output:

```
Time : 2:30:54
```