Assignment – 3

Q1. I**magine a tollbooth at a bridge. Carspassing by the booth are expected to pay five-rupeetoll. Mostly they do, but sometimes a car go by without paying. The tollbooth keeps track of thenumber of cars that have gone by and of the total amount of money collected.Model this tollbooth with class (say TollBooth). The two data members to hold number of carsand amount of money collected. The constructor may initialize both these to 0. A memberfunction (say payingCar() increments the car total and adds 5 to the cash total, while anothermember function (say noPayCar()) increments the car total but adds nothing to the cash total.Finally member function display() displays the two totals. Hint: allow user to press a particularkey to indicate a paying car and another key to indicate non paying car.**

Answer :

```
#include <iostream>

using namespace std;

class TollBooth
{

public:
   int noOfCar, moneyCollected;
   TollBooth()
   {

      noOfCar = 0;
      moneyCollected = 0;
   }

   void payingCar()
   {
      noOfCar++;
      moneyCollected += 5;
      cout << endl
           << "Paying Car Added" << endl;
   }

   void noPayCar()
   {

      noOfCar++;

      cout << endl
           << "Non Paying Car Added" << endl;
   }
```

```
    void display()
    {

        cout << endl
             << "Total No of Car : " << noOfCar << endl
             << "Total Collected Money : " << moneyCollected << endl;
    }

    void displayMenu()
    {

        cout << endl;
        cout << "1. Add Paying Car" << endl;
        cout << "2. Add Non Paying Car" << endl;
        cout << "3. Display " << endl;
        cout << "4. Exit Program";
        cout << endl;
    }
};

int main()
{
    int input;
    TollBooth tollBooth;

    while (1)
    {
        tollBooth.displayMenu();
        cout << endl
             << "Input : ";
        cin >> input;
        cout << endl;

        switch (input)
        {

        case 1:
            tollBooth.payingCar();
            break;
        case 2:
            tollBooth.noPayCar();
            break;

        case 3:
            tollBooth.display();
            break;
        case 4:
            exit(1);
```

**Q2. Define a class to represent a bank account. Include the following members:**
**Data Members:**
**i) Name of the account holder**
**ii) Account number**
**iii) Balance Amount is the account**
**Member functions:**
**i) Open an account**
**ii) Deposit and withdraw money**
**iii) Display account information**
**Write a program to test this class for 10 costumers. Make use of all three types of constructors (whenever appropriate)**

```
#include <iostream>
#include <cstring>
#include <cstdlib>

using namespace std;

class Bank
{
public:
   char accountHolderName[20];
   int balanceAmount, accountNumber;

   Bank()
   {
      balanceAmount = 0;
      accountNumber = 0;
   }
   int openAccount(char name[20])
   {
      strcpy(accountHolderName, name);
      int max = 100; //set the upper bound to generate the random number
      accountNumber = (rand() % max);
      cout << endl
          << "Account Successfully Opened with : " << endl;
      cout << "Account Name : " << accountHolderName << endl;
      cout << "Account Number : " << accountNumber << endl;
      cout << "Balance Amount : " << balanceAmount << endl;
      return 1;
   }

   int depositMoney(int money)
```

```
    {
        balanceAmount += money;
        cout << money << " is Deposited";
        return 1;
    }

    int withdrawMoney(int money)
    {
        if (money > balanceAmount)
        {
            cout << endl
                 << "Insufficient Balance";
            return 0;
        }
        balanceAmount -= money;
        cout << money << " is Withdrawn";
        return 1;
    }

    void accountInformation()
    {
        cout << endl
             << "Account Name : " << accountHolderName << endl;
        cout << "Account Number : " << accountNumber << endl;
        cout << "Total Balance : " << balanceAmount << endl;
    }
};

int main()
{
    int input, money;
    Bank bank[10];

    //Account Creation

    char name[20];
    for (int i = 0; i <= 9; i++)
    {
        cout << endl
             << "Enter Account Holder Name : ";
        cin >> name;
        bank[i].openAccount(name);
    }
    cout << endl
         << "Deposit money : ";

    for (int i = 0; i <= 9; i++)
    {
```

```
        cout << endl
             << "Customer No : " << (i + 1) << endl;
        cout << endl
             << "Amount you want to deposit : ";
        cin >> money;
        bank[i].depositMoney(money);
    }

    cout << endl
         << "Withdraw Money : " << endl;

    for (int i = 0; i <= 9; i++)
    {

        cout << endl
             << "For Customer No : " << (i + 1);
        cout << endl
             << "Amount you want to withdraw : ";
        cin >> money;
        bank[i].withdrawMoney(money);
    }

    for (int i = 0; i <= 9; i++)
    {
        bank[i].accountInformation();
    }

    return 1;
}
```

**Q3. Create a class called Mountain with data members name, height and location. Use constructor that initialize the members to the values passed to it as parameters. A function cmpHeight() to compare height of two objects and function displayInfo() to display information of mountain. In main create two objects of the class mountain and display the information of mountain with greatest height.**

```
#include <iostream>
#include <cstring>
using namespace std;

class Mountain
{

private:
    char name[10], location[15];
    int height;
```

```
public:
   Mountain()
   {
      strcpy(name, "");
      strcpy(location, "");
      height = 0;
   }

   Mountain(char n[], char l[], int h)
   {
      strcpy(name, n);
      strcpy(location, l);
      height = h;
   }

   void displayInfo()
   {
      cout << endl
          << "Name : " << name << endl
          << " Height : " << height << endl
          << " Location : " << location
          << endl
          << endl;
   }

   friend void cmpHeight(Mountain m1, Mountain m2);
};

void cmpHeight(Mountain m1, Mountain m2)
{
   (m1.height > m2.height) ? m1.displayInfo() : m2.displayInfo();
}
int main()
{
   Mountain m1("Sagarmatha", "Nepal", 8488);
   Mountain m2("Annapurna", "Pokhara", 8000);

   cout << endl
       << "Given Info \n\n ";
   m1.displayInfo();
   m2.displayInfo();
   cout << endl
       << "====================\n";

   cmpHeight(m1, m2);

   return 1;
}
```

```
        default:
            cout << " Wrong Input";
        }
    }

    return 1;
}
```

**Q4. Can Constructor be Overloaded? Explain with an example program.**

Ans. Yes, the constructor can be overloaded.

Example :

```
#include <iostream>
using namespace std;

class Person {
   private:
    int age;

   public:
    // 1. Constructor with no arguments
    Person() {
        age = 20;
    }

    // 2. Constructor with an argument
    Person(int a) {
        age = a;
    }

    int getAge() {
        return age;
    }
};

int main() {
    Person person1, person2(45);

    cout << "Person1 Age = " << person1.getAge() << endl;
    cout << "Person2 Age = " << person2.getAge() << endl;

    return 0;
}
```

Output:

Person1 Age = 20
Person2 Age = 45

In this example, there are two constructor ; one with no argument and another with an argument ; which shows constructor overloading.

**Q5. Can destructor be Overloaded? If yes, explain with an example program**

Ans.  No, the destructor cannot be overloaded.Only one empty destructor per class should be there. It must have a void parameter list. Destructor in C++ neither takes any parameters nor does it return anything.