1. Employee Class

Write a class named Employee that has the following member variables:

• name. A string that holds the employee's name.

• idNumber. An int variable that holds the employee's ID number.

• department. A string that holds the name of the department where the employee works.

• position. A string that holds the employee's job title.

The class should have the following constructors:

• A constructor that accepts the following values as arguments and assigns them to the appropriate member variables: employee's name, employee's ID number, department, and position.

• A constructor that accepts the following values as arguments and assigns them to the appropriate member variables: employee's name and ID number. The department and position fields should be assigned an empty string ("").

• A default constructor that assigns empty strings ("") to the name, department, and position member variables, and 0 to the idNumber member variable.

Write appropriate mutator functions that store values in these member variables and accessor functions that return the values in these member variables. Once you have written the class, write a separate program that creates three Employee objects to hold the following data.

Name	ID Number	Department	Position
Susan Meyers	47899	Accounting	Vice President
Mark Jones	39119	IT	Programmer
Joy Rogers	81774	Manufacturing	Engineer

The program should store this data in the three objects and then display the data for each employee on the screen.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To set empty string and o value to member of class
 using constructors
*/

#include <iostream>
using namespace std;
class Employee{
 private:
 string name;
 int idNumber;
 string department;
 string position;
 public:
 // Constructor Function with arguments
```

```
 Employee(string, int, string, string);
 // Constructor Function that sets
 // department and position empty strings
 Employee(string, int);
 // constructor Function that sets
 // string to empty and 0 to int
 Employee();
 void Display();
};
Employee::Employee(string nam, int id, string dep, string pos)
{
 name = nam;
 idNumber = id;
 department = dep;
 position = pos;
}
Employee::Employee(string na, int iD)
{
 name = na;
 idNumber = iD;
 department = " ";
 position = " ";
}
Employee::Employee()
{
 name = " ";
 idNumber = 0;
 department = " ";
 position = " ";
}

void Employee::Display()
{
 cout << "\nName:" << name << endl;
 cout << "\nIdNumber:" << idNumber << endl;
 cout << "\nDepartment:" << department << endl;
 cout << "\nPosition:" << position << endl;
}
int main()
{
 int id;
 string name, dep, pos;
 cout << "\nEnter the name of Employee:\n";
 cin >> name;
 cout << "\nEnter the id of Employee:\n";
 cin >> id;
 cout << "\nEnter the department of Employee:\n";
 cin >> dep;
 cout << "\nEnter the position of Employee:\n";
```

```
 cin >> pos;
 Employee e1(name, id, dep, pos);
 e1.Display();
 Employee e2(name, id);
 e2.Display();
 Employee e3;
 e3.Display();
 return 0;
}
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To display information of Employee using Employee class
*/
#include <iostream>
#include <iomanip>
using namespace std;
class Employee{
 private:
 string name;
 int idNumber;
 string department;
 string position;
 public:
 void setEmployee(string, int, string, string);
 void Display();
};
void Employee::setEmployee(string nam, int id, string dep,
string pos)
{
 name = nam;
 idNumber = id;
 department = dep;
 position = pos;
}
void Employee::Display()
{
 cout << left << setw(15) << name;
 cout << left << setw(10) << idNumber;
 cout << left << setw(15) << department;
 cout << left << setw(15) << position;
 cout << "\n";
}
int main()
{
 Employee e1,e2,e3;
 e1.setEmployee("Susan Meyers", 47899, "Accounting",
 "Vice President");
```

```
 cout << left << setw(15) << "Name";
 cout << left << setw(10) << "IdNumber";
 cout << left << setw(15) << "Department";
 cout << left << setw(15) << "Position";
 cout << "\n";
 e1.Display();
 e2.setEmployee("Mark Jones", 39119, "IT", "Programmer");
 e2.Display();
 e3.setEmployee("Joy Rogers", 81744, "Manufacturing",
 "Engineer");
 e3.Display();
 return 0;
}
```

2. Car Class

Write a class named Car that has the following member variables:

• yearModel. An int that holds the car's year model.

• make. A string that holds the make of the car.

• speed. An int that holds the car's current speed.

In addition, the class should have the following constructor and other member functions.

• Constructor. The constructor should accept the car's year model and make as arguments. These values should be assigned to the object's yearModel and make member variables. The constructor should also assign 0 to the speed member variables.

• Accessor. Appropriate accessor functions to get the values stored in an object's yearModel, make, and speed member variables.

• accelerate. The accelerate function should add 5 to the speed member variable each time it is called.

• brake. The brake function should subtract 5 from the speed member variable each time it is called.

Demonstrate the class in a program that creates a Car object, and then calls the accelerate function five times. After each call to the accelerate function, get the current speed of the car and display it. Then, call the brake function five times. After each call to the brake function, get the current speed of the car and display it.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To display speed of a car while accelerate and break
 function is called and make use of constructor function
*/
#include <iostream>
using namespace std;
class Car{
 private:
 int yearModel;
 string make;
```

```
 int speed;
 public:
 Car(int, string); //constructor function
 void access(); //accessor function
 int accelerate(int*);
 int brake(int*);
};
Car::Car(int model ,string mak)
{
 yearModel = model;
 make = mak;
 speed = 0;
}
void Car::access()
{
 cout << "\nYearModel:" << yearModel;
 cout << "\nMaking:" << make;
 cout << "\nSpeed:" << speed;
 cout << "\n";
}
int Car::accelerate(int *speed)
{
 *speed = *speed + 5;
 return (*speed);
}
int Car::brake(int *speed)
{
 *speed = *speed - 5;
 return (*speed);
}
int main()
{
 int i,j,mod, s = 0;
 string make;
 cout << "\nEnter the model of car:\n";;
 cin >> mod;
 cout << "Enter the making of car:\n";
 cin >> make;
 Car c(mod,make);
 c.access();
 cout << "You are Accelerating at:\n";
 for(i = 0; i < 5; i++)
 {
 cout << c.accelerate(&s) << "km/h speed\n";
 }
 cout << "\nYou are Decelarating at:\n";
 for(i = 0; i < 5; i++)
 {
 cout << c.brake(&s) << "Km/h speed\n";
 }
 return 0;
}
```

3. RetailItem Class

Write a class named RetailItem that holds data about an item in a retail store. The class should have the following member variables:

• description. A string that holds a brief description of the item.

• unitsOnHand. An int that holds the number of units currently in inventory.

• price. A double that holds the item's retail price.

Write a constructor that accepts arguments for each member variable, appropriate mutator functions that store values in these member variables, and accessor functions that return the values in these member variables. Once you have written the class, Write a separate program that creates three RetailItem objects and stores the following data in them.

	Description	Units On Hand	Price
Item #1	Jacket	12	59.95
Item #2	Designer Jeans	40	34.95
Item #3	Shirt	20	24.95

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To display Description of Retail shop with the help of
 constructor function
*/
#include <iostream>
#include <iomanip>
using namespace std;
class RetailItem{
 private:
 string description;
 int unitsOnHand;
 double price;
 public:
 RetailItem(string, int, double);
 void DisplayItem();
};
RetailItem::RetailItem(string des, int units, double pr)
{
 description = des;
 unitsOnHand = units;
 price = pr;
}
void RetailItem::DisplayItem()
{
 cout << left << setw(20) << description;
 cout << left << setw(15) << unitsOnHand;
 cout << left << setw(15) << price;
 cout << "\n";
}
int main()
```

```
{
 cout << left << setw(15) << " ";
 cout << left << setw(20) << "Description";
 cout << left << setw(15) << "Units on Hand";
 cout << left << setw(15) << "Price";
 cout << "\n";
 RetailItem R1("Jacket",12,59.95);
 cout << left << setw(15) << "Item 1" ;
 R1.DisplayItem();
 RetailItem R2("Designer Jeans",40,34.95);
 cout << left << setw(15) << "Item 2" ;
 R2.DisplayItem();
 RetailItem R3("Shirt",20,24.95);
 cout << left << setw(15) << "Item 3" ;
 R3.DisplayItem();
 return 0;
}
```

4. Widget Factory

Design a class for a widget manufacturing plant. Assuming that 10 widgets may be produced each hour, the class object will calculate how many days it will take to produce any number of widgets. (The plant operates two shifts of eight hours each per day.) Write a program that asks the user for the number of widgets that have been ordered and then displays the number of days it will take to produce them.

Input Validation: Do not accept negative values for the number of widgets ordered.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To design class for widget manufacturing plant
*/
#include <iostream>
using namespace std;
class Widget{
 private:
 int hrs;
 int shift;
 int widget;
 int orderWidget;
 public:
 Widget();
 void DisplayDay();
};
Widget::Widget()
{
 hrs = 8;
 shift = 2;
 widget = 10;
 do
 {
 cout << "Enter the number of widgets to be ordered:\n";
 cin >> orderWidget;
```

```
 }while(orderWidget<1);
}
void Widget::DisplayDay()
{
 float pro,days;
 // widget production per day;
 pro = shift*hrs*widget;
 days = orderWidget/pro;
 cout << "\nDays to build " << orderWidget << " widget is ";
 cout << days;
}
int main()
{
 Widget W;
 W.DisplayDay();
 return 0;
}
```

5. TestScores Class

Design a TestScores class that has member variables to hold three test scores. The class should have a constructor, accessor, and mutator functions for the test score fields, and a member function that returns the average of the test scores. Demonstrate the class by writing a separate program that creates an instance of the class. The program should ask the user to enter three test scores, which are stored in the TestScores object. Then the program should display the average of the scores, as reported by the TestScores object.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To display testscore and find its average by
 making use of constructor,mutator functions
*/
#include <iostream>
using namespace std;
class Testscore
{
private:
 float firstTest;
 float secondTest;
 float thirdTest;
public:
 Testscore();
 void setScore();
 void getScore();
 float average();
};
// default constructor
Testscore::Testscore()
{
 firstTest = 0;
 secondTest = 0;
 thirdTest = 0;
}
```

```
// mutator function
void Testscore::setScore()
{
 cout << "Enter the score of first second and third test:\n";
 cin >> firstTest >> secondTest >> thirdTest;
}
// Accessor function
void Testscore::getScore()
{
 cout << "\nScores of three tests are\n";
 cout << firstTest;
 cout << "\n" << secondTest;
 cout << "\n" << thirdTest;
}
float Testscore::average()
{
 return ((firstTest+secondTest+thirdTest)/3);
}
int main()
{
 Testscore sc;
 sc.setScore();
 sc.getScore();
 cout << "\nAverage Score = " << sc.average();
 return 0;
}
```

6. Circle Class

Write a Circle class that has the following member variables:

radius: a double pi: a double initialized with the value 3.14159

The class should have the following member functions:

- **Default Constructor. A default constructor that sets radius to 0.0.**
- **Constructor. Accepts the radius of the circle as an argument.**
- **setRadius. A mutator function for the radius variable.**
- **getRadius. An accessor function for the radius variable.**
- **getArea. Returns the area of the circle, which is calculated as area = pi * radius * radius**
- **getDiameter. Returns the diameter of the circle, which is calculated as diameter = radius * 2**
- **getCircumference. Returns the circumference of the circle, which is calculated as**

circumference = 2 * pi * radius

Write a program that demonstrates the Circle class by asking the user for the circle's radius, creating a Circle object, and then reporting the circle's area, diameter, and circumference.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To create circle class and calculate area and circumference
```

```
 by making use of constructor function
*/
#include <iostream>
using namespace std;
class Circle{
 private:
 float radius;
 double pi = 3.14159;
 public:
 Circle();
 Circle(float);
 void setRadius();
 void getRadius();
 float getArea();
 float getDiameter();
 float getCircumference();
};
// Default Constructor
Circle::Circle()
{
 radius = 0;
}
// Constructor Function
Circle::Circle(float rad)
{
 radius = rad;
}
void Circle::setRadius()
{
 cout << "\nEnter the radius:\n";
 cin >> radius;
 Circle(radius);
}
void Circle::getRadius()
{
 cout << "Radius of circle is:" << radius;
 cout << "\n";
}
float Circle::getDiameter()
{
 return radius*2;
}
float Circle::getArea()
{
 return (pi*radius*radius);
}
float Circle::getCircumference()
{
 return (2*pi*radius);
```

```
}
int main()
{
 Circle c1;
 c1.setRadius();
 c1.getRadius();
 cout << "\nDiameter of circle is:";
 cout << c1.getDiameter() << endl;
 cout << "\nArea of Circle is:";
 cout << c1.getArea() << endl;
 cout << "\nCircumference of circle is:";
 cout << c1.getCircumference() << endl;
 return 0;
}
```

7. Inventory Class

Design an Inventory class that can hold information and calculate data for items in a retail store's inventory. The class should have the following *private* member variables:

Variable Name	Description
`itemNumber`	An `int` that holds the item's item number.
`quantity`	An `int` for holding the quantity of the items on hand.
`cost`	A `double` for holding the wholesale per-unit cost of the item
`totalCost`	A `double` for holding the total inventory cost of the item (calculated as `quantity` times `cost`).

The class should have the following *public* member functions:

Member Function	Description
Default Constructor	Sets all the member variables to 0.
Constructor #2	Accepts an item's number, cost, and quantity as arguments. The function should copy these values to the appropriate member variables and then call the `setTotalCost` function.
`setItemNumber`	Accepts an integer argument that is copied to the `itemNumber` member variable.
`setQuantity`	Accepts an integer argument that is copied to the `quantity` member variable.
`setCost`	Accepts a `double` argument that is copied to the `cost` member variable.
`setTotalCost`	Calculates the total inventory cost for the item (`quantity` times `cost`) and stores the result in `totalCost`.
`getItemNumber`	Returns the value in `itemNumber`.
`getQuantity`	Returns the value in `quantity`.
`getCost`	Returns the value in `cost`.
`getTotalCost`	Returns the value in `totalCost`.

Demonstrate the class in a driver program. *Input Validation: Do not accept negative values for item number, quantity, or cost.*

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
```

```
 Objective:
 To design a inventory class of a retail store by making
 use of constructor , accessor and mutator functions
*/
#include <iostream>
using namespace std;
class Inventory{
 private:
 int itemNumber;
 int quantity;
 double cost;
 double totalCost;
 public:
 Inventory();
 Inventory(int, double, int);
 void setItemnumber(int);
 void setQuantity(int);
 void setCost(double);
 void setTotalcost();
 int getItemnumber();
 int getQuantity();
 double getCost();
 double getTotalcost();
};
// Default Constructor
Inventory::Inventory()
{
 itemNumber = 0;
 quantity = 0;
 cost = 0;
 totalCost = 0;
}
// Parameterized Constructor Function
Inventory::Inventory(int num, double cst , int qty)
{
 itemNumber = num;
 cost = cost;
 quantity = qty;
 setTotalcost();
}
void Inventory::setItemnumber(int Item)
{
 itemNumber = Item;
}
void Inventory::setQuantity(int qty)
{
 quantity = qty;
}
void Inventory::setCost(double cst)
{
 cost = cst;
}
void Inventory::setTotalcost()
{
 totalCost = cost*quantity;
}
```

```
int Inventory::getItemnumber()
{
 return itemNumber;
}
int Inventory::getQuantity()
{
 return quantity;
}
double Inventory::getCost()
{
 return cost;
}
double Inventory::getTotalcost()
{
 return cost*quantity;
}
int main()
{
 int itemNo;
 int qty;
 double cost;
 do
 {
 cout << "Enter the item Number:\n";
 cin >> itemNo;
 cout << "\nEnter the Quantity:\n";
 cin >> qty;
 cout << "\nEnter the cost:\n";
 cin >> cost;
 }while((itemNo<1 || qty<1 || cost <1));
 Inventory I1(itemNo,cost,qty);
 I1.setItemnumber(itemNo);
 I1.setCost(cost);
 I1.setQuantity(qty);
 cout << "\n";
 cout << I1.getItemnumber();
 cout << "\t" << I1.getQuantity();
 cout << "\t" << I1.getCost();
 cout << "\t" << I1.getTotalcost();
 return 0;
}
```

8. Population

In a population, the birth rate and death rate are calculated as follows:

Birth Rate = Number of Births ÷ Population

Death Rate = Number of Deaths ÷ Population

e.g. in a population of 100,000 that has 8,000 births and 6,000 deaths per year, the birth rate and death rate are:

Birth Rate = 8,000 ÷ 100,000 = 0.08

Death Rate = 6,000 ÷ 100,000 = 0.06

Design a Population class that stores a population, number of births, and number of deaths for a period of time. Member functions should return the birth rate and death rate. Implement the class in a program.
Input Validation: Do not accept population figures less than 1, or birth or death numbers less than 0.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To create population class and display death and birthrate
 and not allow negative value for birth and death.
*/
#include <iostream>
using namespace std;
class Population{
 private:
 int population;
 float death;
 float birth;
 public:
 Population();
 float deathRate();
 float birthRate();
};
Population::Population()
{
 do
 {
 cout << "Enter the population:\n";
 cin >> population;
 }while(population<1);
 do
 {
 cout << "\nEnter number of birth:\n";
19
 cin >> birth;
 }while(birth<1);
 do
 {
 cout << "\nEnter number of death:\n";
 cin >> death;
 }while(death<1);
}
float Population::birthRate()
{
 return (birth/population);
```

```
}
float Population::deathRate()
{
 return (death/population);
}
int main()
{
 Population P;
 cout << "\nBirth rate is " << P.birthRate();
 cout << "\nDeath rate is " << P.deathRate();
 return 0;
}
```

9. Number Array Class

Design a class that has an array of floating-point numbers. The constructor should accept an integer argument and dynamically allocate the array to hold that many numbers. The destructor should free the memory held by the array. In addition, there should be member functions to perform the following operations:

- **Store a number in any element of the array**
- **Retrieve a number from any element of the array**
- **Return the highest value stored in the array**
- **Return the lowest value stored in the array**
- **Return the average of all the numbers stored in the array**

Demonstrate the class in a program.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To display highest,lowest and average of numbers
 in an array by dynamically allocating
*/
#include <iostream>
using namespace std;
class Numbers{
 private:
 float *numArray;
 int siz;
 public:
 Numbers(int); //constructor function
 float highestNum();
 float lowestNum();
 void aVerage();
 ~Numbers(); //Destructor function
};
Numbers :: Numbers(int n)
{
 int i;
 siz = n;
 numArray = new float(n);
```

```
 //dynamic memory allocation using new operator
 cout << "\nEnter the floating point value in the array:\n";
 for(i = 0;i < siz ;i++)
 {
 cin >> numArray[i];
 }
}
float Numbers :: highestNum()
{
 int i;
 float highest;
 highest = numArray[0];
 for(i = 0;i < siz;i++)
 {
 if(numArray[i] > highest)
 {
 highest = numArray[i];
 }
 }
 return highest;
}
float Numbers :: lowestNum()
{
 int i;
 float low;
 low = numArray[0];
 for(i = 0 ;i < siz;i++)
 {
 if(numArray[i] < low)
 {
 low = numArray[i];
 }
 }
 return low;
}
void Numbers :: aVerage()
{
 int i;
 float sum = 0;
 for(i = 0;i< siz ;i++)
 {
 sum = sum + numArray[i];
 }
 cout << "\nAverage of number is:" << sum/siz;
}
Numbers :: ~Numbers()
{
 delete numArray;
}
int main()
{
 int n;
 cout << "Enter the number of floating value you want
 to store:\n";
 cin >> n;
 Numbers N1(n);
```

```
 cout << "\nHighest Number in the array is:";
 cout << N1.highestNum();
 cout << "\nLowest Number in the array is:";
 cout << N1.lowestNum();
 N1.aVerage();
 return 0;
}
```

10. Payroll

Design a PayRoll class that has data members for an employee's hourly pay rate, number of hours worked, and total pay for the week. Write a program with an array of seven PayRoll objects. The program should ask the user for the number of hours each employee has worked and will then display the amount of gross pay each has earned. ***Input Validation: Do not accept values greater than 60 for the number of hours worked.***

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To create a payroll class and calculate gross pay of each
 employee by creating seven objects
*/
#include <iostream>
using namespace std;
class Payroll
{
private:
 float hourRate;
 float hoursWorked;
 float weeklyPay;
public:
 Payroll();
 float ReturnGrossPay();
};
Payroll::Payroll()
{
 hourRate = 100;
 do
 {
 cout << "\nEnter the number of hour worked:\n";
 cin >> hoursWorked;
 if(hoursWorked>60)
 {
 cout << "\nEnter again over 60hrs not allowed\n";
 }
 }while(hoursWorked>60);
}
float Payroll::ReturnGrossPay()
{
 weeklyPay = hoursWorked*hourRate;
 return weeklyPay;
}
int main()
{
```

```
 Payroll p[7];
 int i,c = 1;
 for(i = 0; i < 7; i++)
 {
 cout << "\nWeekly pay for " << c << "Employee is Rs.";
 cout << p[i].ReturnGrossPay();
 c++;
 }
 return 0;
}
```

11. Cash Register

Design a CashRegister class that can be used with the InventoryItem class the CashRegister class should perform the following:

1. Ask the user for the item and quantity being purchased.

2. Get the item's cost from the InventoryItem object.

3. Multiply the unit price times the quantity being purchased to get the purchase total.

4. Display the purchase total on the screen.

5. Subtract the quantity being purchased from the onHand variable of the InventoryItem class object.

Implement both classes in a complete program. Feel free to modify the InventoryItem class in any way necessary. ***Input Validation: Do not accept a negative value for the quantity of items being purchased.***

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To create cash register class that can be used with
 inventory class
*/
#include <iostream>
using namespace std;
class CashRegister{
 private:
 int itemNo;
 int qty;
 public:
 class Inventory
 {
 public:
 int price[4];
 int onHand[4];
 Inventory()
 {
 price[0] = 1000;
 price[1] = 1500;
 price[2] = 1200;
 price[3] = 500;
 onHand[0] = 100;
 onHand[1] = 50;
```

```
 onHand[2] = 60;
 onHand[3] = 30;
 cout << "INVENTORY:\n";
 cout << "Item no.1" << "\t" << "Clothes";
 cout << "\tPrice: " << price[0];
 cout << "\tItems left: " << onHand[0]
 << endl;
cout << "Item no.2" << "\t" << "Bags";
 cout << "\tPrice: " << price[1];
 cout << "\tItems left: " << onHand[1]
 << endl;
cout << "Item no.3" << "\t" << "Shoes";
 cout << "\tPrice: " << price[2];
 cout << "\tItems left: " << onHand[2]
 << endl;
cout << "Item no.4" << "\t" << "Gifts";
 cout << "\tPrice: " << price[3];
 cout << "\tItems left: " << onHand[3]
 << endl;
 }
 };Inventory I;
 void Display()
 {
 do
 {
 cout << "\nEnter the item no and quantity  to purchase:\n";
 cin >> itemNo >> qty;
 if(qty < 0)
 {
 cout << "\nPlease enter again negative quantity not allowed:";
 }
 }while(qty < 0);
 switch(itemNo)
 {
 case 1:
 {
 cout << "Total Price: "
<< ((I.price[0])*qty);
 RemainItems((I.onHand[0])-qty);
 break;
 }
 case 2:
 {
 cout << "Total Price: "
<< ((I.price[1])*qty);
 RemainItems((I.onHand[1])-qty);
 break;
 }
 case 3:
 {
 cout << "Total Price: "
<< ((I.price[2])*qty);
 RemainItems((I.onHand[2])-qty);
 break;
 }
 case 4:
```

```
 {
 cout << "Total Price: "
<< ((I.price[3])*qty);
 RemainItems((I.onHand[3])-qty);
 break;
 }
 default:
 exit(0);
 }
 }
 void RemainItems(int n)
 {
 cout << "\nRemaining quantity is: " << n;
 }
};
int main()
{
 CashRegister cr;
 cr.Display();
 return 0;
}
```

12. Trivia Game

In this programming challenge you will create a simple trivia game for two players.
The program will work like this:

• Starting with player 1, each player gets a turn at answering five trivia questions. (There are a total of 10 questions.) When a question is displayed, four possible answers are also displayed. Only one of the answers is correct, and if the player selects the correct answer he or she earns a point.

• After answers have been selected for all of the questions, the program displays the number of points earned by each player and declares the player with the highest number of points the winner.

In this program you will design a Question class to hold the data for a trivia question.
The Question class should have member variables for the following data:

- **A trivia question**
- **Possible answer #1**
- **Possible answer #2**
- **Possible answer #3**
- **Possible answer #4**
- **The number of the correct answer (1, 2, 3, or 4)**

The Question class should have appropriate constructor(s), accessor, and mutator functions.
The program should create an array of 10 Question objects, one for each trivia question.
Make up your own trivia questions on the subject or subjects of your choice for the objects.

```
/*
 Programmers Name: Ankit Sharma
 Date: 6/28/2020
 Objective:
 To create a trivia game that can ask five questions
 to each player and decide the winner by creating 10
 question objects.
```

```
*/
#include <iostream>
#include <iomanip>
using namespace std;
class Question{
 private:
 string question;
 string ans1;
 string ans2;
 string ans3;
 string ans4;
 string ans;
 public:
 static int count;
 //Constructor function
 Question(string quest)
 {
 question = quest;
 }
 // mutator function
 Question(string one,string two,string three,string four)
 {
 ans1 = one;
 ans2 = two;
 ans3 = three;
 ans4 = four;
 }
 // accessor function
 void DisplayQuest()
 {
 cout << question << endl;
 }
 void DisplayAns()
 {
 cout << left << setw(15) << ans1;
 cout << left << setw(15) << ans2;
 cout << left << setw(15) << ans3;
 cout << left << setw(15) << ans4;
 cout << "\n";
 }
};
int Question:: count;
int main()
{
 int a, b,i,j;
 string ans1,ans2;
 string correct[10] =
{"Everest","Seven","Tokyo","Jhonkha","Marconi","ElonMusk","Barcelona","Fifteen","JeffBezos","Stroustrup"};
 cout << "\t\t\t" << "TRIVIA GAME" << endl;
 Question q[10] ={ Question("1.Which is the highest pick in the world ?"),
 Question("2.How many states are there in Nepal ?"),
Question("3.What is the capital of Japan ?"),
Question("4.Which is the  official language of Bhutan ?"),
Question("5.Who is the inventor of Radio ?"),
Question("6.Who is the founder of space-X ?"),
 Question("7.Which club Lionel Messi Plays For ?"),
```

```
Question("8.How many players are there in a rugby team ?"),
Question("9.Who is the richest man in the world ?"),
Question("10.Who invented C++ language ?")
 };
 Question ans[10] ={Question("Everest","Annapurna","K2","Gaurishankar\n"),
Question("Eight","Nine","Seven","Fourteen\n"), Question("Beijing","Tokyo","Shengahai","Ankara\n"),
Question("Jhonkha","Ronkha","Lopcha","Hindi\n"), Question("Einstein","Galileo","Newton","Marconi\n"),
Question("ElonMusk","JebBoss","JackMa","Malaila"), Question("RealMadrid","Barcelona","Napoli","Dortmund"),
Question("Seventeen","Eleven","Thirteen","Fifteen"),Question("JeffBezos","Trump",
"MukeshAmbani","Billgates"), Question("Stroustrup","Sutter","Thompson","Ritchie")
 };
 cout << "\tWelcome Player No.1" << endl;
 for(i = 0;i < 5; i++)
 {
 q[i].DisplayQuest();
 ans[i].DisplayAns();
 cout << "Enter the answer\n";
 cin >> ans1;
 if(ans1 == correct[i])
 {
 Question::count++;
 }
 }
 a = Question:: count;
 cout << "You Got " << a << "Points" << endl;
 cout << "\tWelcome Player No.2" << endl;
 Question:: count = 0;
 for(j = 5;j < 10; j++)
 {
 q[j].DisplayQuest();
 ans[j].DisplayAns();
 cout << "Enter the answer\n";
 cin >> ans2;
 if(ans2 == correct[j])
 {
 Question::count++;
 }
 }
 b = Question::count;
 cout << "You got" << b << " Points" << endl;
 if(a > b)
 {
 cout << "Player 1 won ";
 }
 else if(b > a)
 {
 cout << "Player 2 won ";
 }
 else
 cout << "Its a Draw";
 return 0;
}
```