



OBJECT ORIENTED PROGRAMMING

LAB # 7

ASSIGNMENT

NAME: S.M.IRTIZA

ROLL NO. : 22K-4638

CLASS: 2-F

QUESTION

#

01:

Write a program that simulates a group of people using a class called Person that has a static member variable count, which holds the number of people in the group. The Person class also has static member functions to get the count and reset it to zero. The program creates three instances of the Person class with different names and increments the count each time a new Person object is created. The program then prints out the total number of people and each person's name using a non-static member function.

CODE:

```
//NAME:S.M.IRTIZA ROLL NO.:22K-4638
#include<iostream>
#include<string>
using namespace std;
class Person{
private:
static int count;
string name;

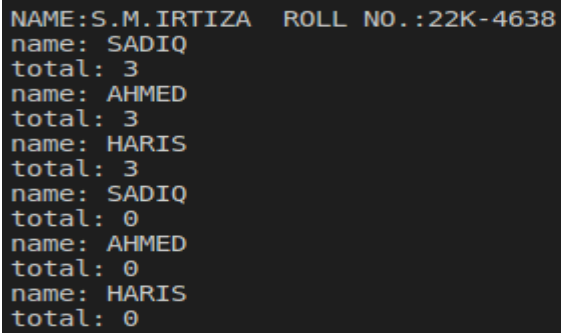
public:
Person(){ }
Person(string a){
name=a;
count++;
}
void setName(string a){
name=a;
}
string getName(){
return name;
}

static int getCount(){
return count;
}
void display(){
cout<<"name: "<<getName()<<endl;
cout<<"total: "<<getCount()<<endl;
}
static void resetCount(){
count=0;
}
};

int Person::count(0);

int main(){
cout<<"NAME:S.M.IRTIZA ROLL NO.:22K-4638"<<endl;
Person P1("SADIQ"),P2("AHMED"),P3("HARIS");
P1.display();
P2.display();
```

```
P3.display();
Person::resetCount();
P1.display();
P2.display();
P3.display();
}
```

OUTPUT :

```
NAME:S.M.IRTIZA  ROLL NO.:22K-4638
name: SADIQ
total: 3
name: AHMED
total: 3
name: HARIS
total: 3
name: SADIQ
total: 0
name: AHMED
total: 0
name: HARIS
total: 0
```

QUESTION # 02:

Make a program that consists of three classes: Person, Car, and Garage.

Add the details as following:

The Person class represents a person and has a single member variable name. The Car class represents a car and has two member variables: make and owner. make represents the make of the car and owner is a pointer to a Person object, which creates an aggregation relationship between the Car and Person classes. The Garage class represents a garage and has an array of Car pointers. This creates a composition relationship between the Garage and Car classes. In the main() function, the program creates some Person and Car objects and adds them to a Garage object. The program then prints out the details of the cars in the garage, including their make and owner.

CODE:

```
//NAME:S.M.IRTIZA ROLL NO.:22K4638
```

```
#include <iostream>
```

```
#include<string>
```

```
using namespace std;
```

```
class Person {  
private:  
string name;
```

```
public:  
Person(string name) {  
this->name = name;  
}
```

```
string getName() {  
return name;  
}  
};
```

```
class Car {  
private:  
string make;  
Person* owner;
```

```
public:  
Car(string make, Person* owner) {  
this->make = make;  
this->owner = owner;  
}
```

```
string getMake() {  
return make;  
}
```

```
Person* getOwner() {  
return owner;  
}  
};
```

```
class Garage {
private:
Car* cars[100];
int numCars;

public:
Garage() {
numCars = 0;
}

void addCar(Car* car) {
cars[numCars] = car;
numCars++;
}

void printCars() {
for (int i = 0; i < numCars; i++) {
Car* car = cars[i];
cout << "Car " << i + 1 << " make: " << car->getMake() << endl;
cout << "Car " << i + 1 << " owner: " << car->getOwner()->getName() << endl;
}
}
};

int main() {
cout<<"NAME:S.M.IRTIZA ROLL NO.:22K4638"<<endl;
Person* person1 = new Person("Jack");
Person* person2 = new Person("charlie");

Car* car1 = new Car("HONDA", person1);
Car* car2 = new Car("TOYOTA", person1);
Car* car3 = new Car("FERRARI", person2);

Garage garage;
garage.addCar(car1);
garage.addCar(car2);
garage.addCar(car3);

garage.printCars();

delete person1;
delete person2;
delete car1;
delete car2;
delete car3;

return 0;
```

}

OUTPUT :

```
NAME:S.M.IRTIZA  ROLL NO.:22K4638
Car 1 make: HONDA
Car 1 owner: Jack
Car 2 make: TOYOTA
Car 2 owner: Jack
Car 3 make: FERRARI
Car 3 owner: charlie
```

QUESTION # 03:

Suppose you're creating a Rectangle class to represent rectangles in a 2D coordinate system. The Rectangle class should have two member variables: width and height. Write a program that defines the Rectangle class using member initialization list to initialize the member variables to default values of 0.

In the main() function, create two Rectangle objects and prompt the user to enter the width and height of each rectangle. After the user enters the values, calculate and display the area of each rectangle.

CODE:

```
/"NAME:S.M.IRTIZA ROLL NO.:22K4638"
#include<iostream>
using namespace std;
class Rectangle{
private:
int height;
int width;

public:
Rectangle():height(0),width(0){}

void setHeight(int a){
height=a;
}

void setWidth(int a){
width=a;
}

int getHeight(){
return height;
}
int getWidth(){
return width;
}

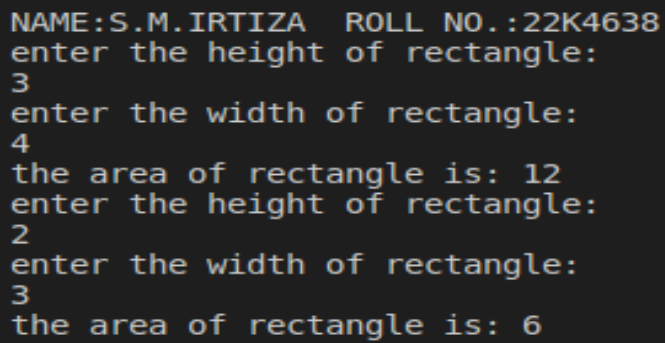
void input(){
int Width,Height;
cout<<"enter the height of rectangle: "<<endl;
cin>>Height;
height=Height;
cout<<"enter the width of rectangle: "<<endl;
cin>>Width;
width=Width;
}

int calculateArea(){
return (getHeight()*getWidth());
}
void display(){
```

```
cout<<"the area of rectangle is: "<<calculateArea()<<endl;  
}  
};
```

```
int main(){  
cout<<"NAME:S.M.IRTIZA ROLL NO.:22K4638"<<endl;  
Rectangle R,r;  
R.input();  
R.display();  
  
r.input();  
r.display();  
}
```

OUTPUT :

A screenshot of a terminal window showing the output of the C++ program. The text is as follows:

```
NAME:S.M.IRTIZA  ROLL NO.:22K4638  
enter the height of rectangle:  
3  
enter the width of rectangle:  
4  
the area of rectangle is: 12  
enter the height of rectangle:  
2  
enter the width of rectangle:  
3  
the area of rectangle is: 6
```


QUESTION # 04

Suppose you are working on a program to calculate the distance between two points in 2D space. You decide to use a class called Point to represent a point, with two private member variables x and y representing the coordinates of the point.

Using inline functions, write a program that prompts the user to enter the coordinates of two points, creates Point objects for each point, calculates the distance between the two points using an inline function, and displays the result to the user.

Hint: To calculate the distance between two points (x1, y1) and (x2, y2), use the formula $\sqrt{(x2 - x1)^2 + (y2 - y1)^2}$.

CODE:

```
//NAME:S.M.IRTIZA ROLL NO.:22K4638
```

```
#include<iostream>
```

```
#include<string>
```

```
#include<cmath>
```

```
using namespace std;
```

```
class Point{
```

```
int x;
```

```
int y;
```

```
public:
```

```
Point (){} 
```

```
Point (int a, int b){
```

```
x=a;
```

```
y=b;
```

```
}
```

```
void setX(int a){
```

```
x=a;
```

```
}
```

```
void setY(int a){
```

```
y=a;
```

```
}
```

```
int getX(){
```

```
return x;
```

```
}
```

```
int getY(){
```

```
return y;
```

```
}
```

```
inline void distance(Point P2){
```

```
int distance= sqrt((pow((x-P2.x),2)+pow((y-P2.y),2)));
```

```
cout<<"the distance is: "<<distance<<endl;
```

```
}
```

```
};
```

```
int main(){
```

```
cout<<"NAME:S.M.IRTIZA ROLL NO.:22K4638"<<endl;
```

```
Point p1(5,5), p2(2,1);  
p1.distance(p2);  
}
```

OUTPUT :

```
NAME:S.M.IRTIZA  ROLL NO.:22K4638  
the distance is: 5
```

QUESTION #05

Create a program that models a university department using classes for Instructor, Course, and Department. The Department class should contain an array of Instructor objects and the instructor class should contain an array of Course objects. Each Course object should contain a pointer to the instructor who teaches it. Implement appropriate constructors, destructors, and member functions for each class. Use the main() function to create a Department object, add Instructor and Course objects to it, and display the department's instructors and their courses.

CODE:

```
//NAME:S.M.IRTIZA ROLL NO.: 22K-4638
```

```
#include<iostream>
```

```
#include<string>
```

```
using namespace std;
```

```
class Instructor;
```

```
class Departement;
```

```
class Course{
```

```
private:
```

```
Instructor* instructor;
```

```
string courseName;
```

```
public:
```

```
Course(){} }
```

```
Course(string a, Instructor* instructor){
```

```
courseName=a;
```

```
this->instructor=instructor;
```

```
}
```

```
Instructor* getInstructor (){
```

```
return instructor;
```

```
}
```

```
void setCourseName(string a){
```

```
courseName=a;
```

```
}
```

```
string getCourseName(){
```

```
return courseName;
```

```
}
```

```
};
```

```
class Instructor{
```

```
private:
```

```
Course* course[3];
```

```
string instructorName;
```

```
int numCourse;
```

```
public:
```

```
Instructor(){} }
```

```
Instructor(string a){
```

```
instructorName=a;
```

```
}
```

```
void setInstructorName(string a){
instructorName=a;
}
string getInstructorName(){
return instructorName;
}
int getNumCourse(){
return numCourse;
}
void addCourse(Course* cou){
course[numCourse]=cou;
numCourse++;
}
```

```
Course* getCourse(int i){
return course[i];
}
};
```

```
class Departement{
private:
Instructor* instructor[10];
string DeptName;
int numINS;
public:
Departement(){ }
Departement(string a){
DeptName=a;
numINS=0;
}
```

```
void addInstructor(Instructor* ins){
instructor[numINS]=ins;
numINS++;
}
```

```
void setDeptName(string a){
DeptName=a;
}
string getDeptName(){
return DeptName;
}
```

```
void display(){
cout<<"departement name: "<< getDeptName()<<endl;
for(int i=0; i<numINS;i++){
Instructor* instructors=instructor[i];
cout << "Instructor " << i + 1 <<": "<<instructors->getInstructorName() << endl;
```

```

for(int j=0;j<instructors->getNumCourse();j++){
cout << "course " << j + 1 << ":" << instructors->getCourse(j)->getCourseName() <<
endl;
}
}

}

};

```

```

int main(){
cout<<"NAME:S.M.IRTIZA ROLL NO.:22K4638"<<endl;
Instructor* instructor1=new Instructor("John");
Instructor* instructor2=new Instructor("Jarry");

Course* course1= new Course("PF", instructor1);
instructor1->addCourse(course1);
Course* course2= new Course("AP", instructor1);
instructor1->addCourse(course2);
Course* course3= new Course("PF", instructor2);
instructor2->addCourse(course3);

```

```

Departement departement("computer science ");
departement.addInstructor(instructor1);
departement.addInstructor(instructor2);

```

```

departement.display();

```

```

delete instructor1;
delete instructor2;
delete course1;
delete course2;
delete course3;
return 0;
}

```

OUTPUT :

```

NAME:S.M.IRTIZA  ROLL NO.:22K4638
departement name: computer science
Instructor 1:John
course 1:PF
course 2:AP
Instructor 2:Jarry
course 1:PF

```