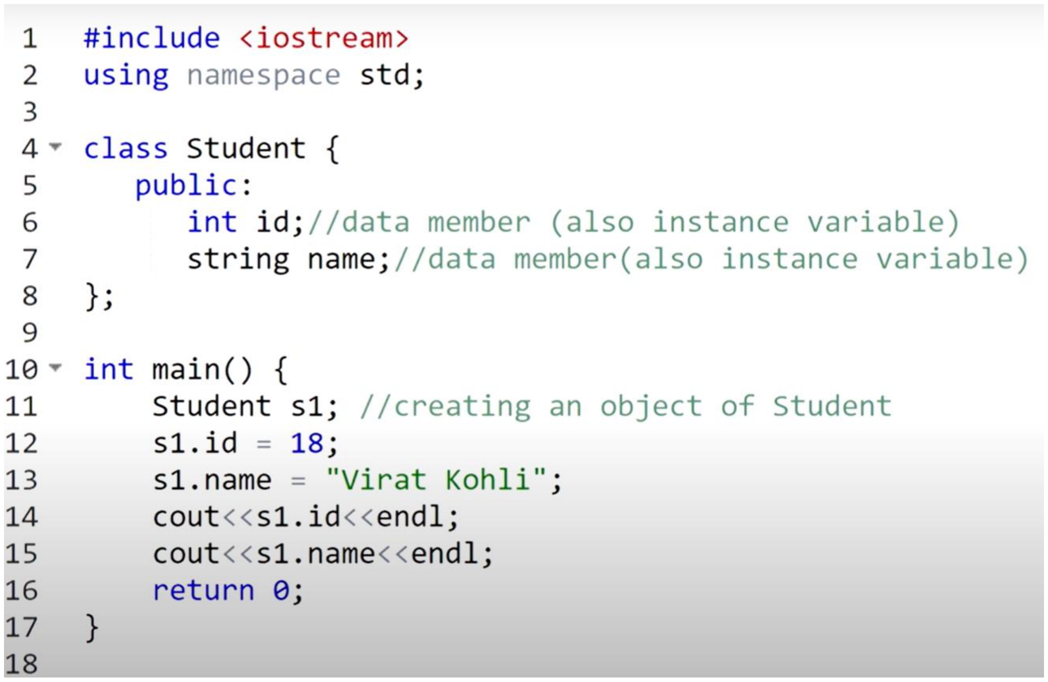
FUNDAMENTALS OF OBJECT ORIENTED PROGRAMMING

Lecture03

Lecture 3: Introduction to Member Data and Member Functions in C++

welcome to lecture 3 introduction to object oriented programming in the last lecture we had seen this program if you recall all. So here we define the class and the objects. So, here the class is student and you can see student s1. s1 is the object so this s1 object which is invoking the member data id and name so s1.id so if you look at the class you have ID and name as a data members or member data.



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

class Student {
   public:
      int id;//data member (also instance variable)
      string name;//data member(also instance variable)
};

int main() {
    Student s1; //creating an object of Student
    s1.id = 18;
    s1.name = "Virat Kohli";
    cout<<s1.id<<endl;
    cout<<s1.name<<endl;
    return 0;
}
```

So, now line number 12 and 13, s1.ID as I said in the last class this dot operator is a bridge between the object and the data member. So, s1.ID is 18 and s1.name is Virat Kohli and then we are trying to print these two and we know we have got this as output. So now in today's lecture, I will introduce the programming point of view, which is called developer C++.

So we will run this code with the help of developer C++. So we will see the first code. Yeah, the same code that we have written, right? So you can use any compiler, and here, I mean, throughout my lecture, I am planning to use this developer C++, right? So, you can install this and those who have already done the basic C++ programming up to procedural oriented programming, you might have done, right?

So, we will run the code. This code we have already seen and we will run the code. So, when I run this code, compile and run the code, so you can see the output over here, right? So, you will get the output over here. So when you look at here, so you can see 18 and Virat Kohli.

So I will just press escape, press any key to continue. So we will go to the next part of the slide, right. So when we run the code, we got this output 18 and Virat Kohli, right. Also we have seen this example. Box is the class and you have the objects box 1 and box 2.

We use the variable name volume so we calculated the volume for both the boxes volume of the box 1 and volume of the box 2. So when you multiplied height length and breadth you will get the output. So this we had seen in the last class and in fact when we run the code .So you have height 5.0 length 6.0 and breadth 7.0, similarly you have a height 10.0 length 12.0 and breadth 13.0. So, when you calculate the volume you are seeing the output over here. So, this code also we will run. So, here you go the same code and if you run this code compile and run.

So, you can see the output right whatever output we got. So, here you can look and we can see the same output. So, again I will go back to the slide. So, both the code we have run. and we got this as the output in the compiler also developer C++ compiler also we have got the same output.

So now I will slowly introduce the member function so you know in the last two programs how to use the member data or data member. So now we will introduce the member function. So here you go, class student which is nothing but a class Right. The class student, which is the class. And then you are here.

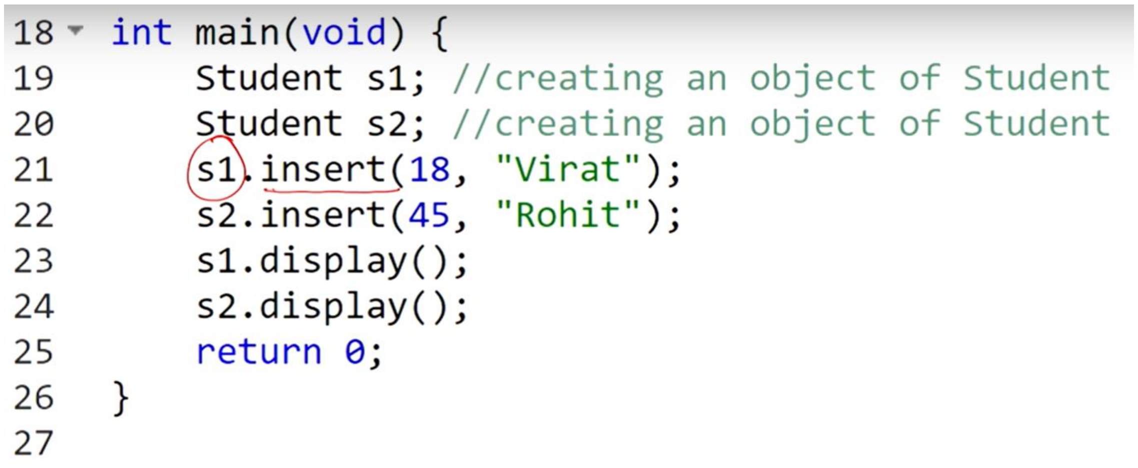
You have a two data members. One is ID. Another one is the string name. Right. One is ID.

Another one is a string name. Both are data members. Right. So now, for the first time, we are seeing the member function. This is called as a member function.

So inside the class, you have data member or member data. This is the member function. Right. So here it is, an insert. Right.

I. as an integer string as a n, so I am just copying right. This is like a getter and setter. So, you will get the value and set this into id and name right, and we have one more function. This is the first function, and we have one more function called display right. So, we have one more function called display. So, the display will display id and name right.

So, the display will display the cout id and name right so this is what we have written in the class. So, you have two member data or data member ID and name, and you have two member functions, which are insert and display now we will go to the next slide, so here you go. So, here you have the main program. We have created two objects. One object is s1 under student class. Another object is S2 under student class.



```
int main(void) {
    Student s1; //creating an object of Student
    Student s2; //creating an object of Student
    s1.insert(18, "Virat");
    s2.insert(45, "Rohit");
    s1.display();
    s2.display();
    return 0;
}
```

So we have instantiated two objects. So now I am calling this object s1 is invoking the member function called insert s1. You have a dot operator that is exactly like what we saw in the case of member data. Right. So here it is a member function.

The s1 is the object. You have a ‘.’ operator. And then you have a member function. So this object is invoking the member function. And you are passing two values.

One is 18. Another one is Virat. So what is happening? It will go to the member function.

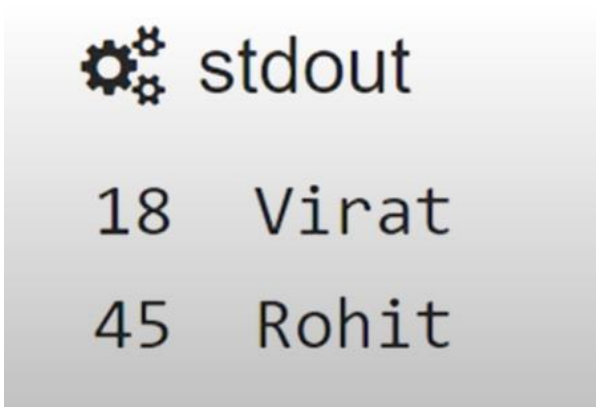
So this will go to the member function. What is the member function? Here. Right.

So you have passed. i is 18. and the string n, which is nothing but Virat n, is nothing but virat. So that means what will happen is id will be equal to 18, and the name will be Virat, right? This is what is happening in line number 9 and 10. So when you are calling this function, I will go to the next slide. When you are calling this function s1, the, under object s1. So this has id, which means s1.id will be 18, and s1.name will be what?

Virat, right? And another s2 object which is invoking again insert, right? So now id is 45 and n is, the string is Rohit. So now, if you look here, what will happen? The second class means second object s2, which is invoking insert, right?

Now i value will take 45, I will write it here for s2. the one I have written is for s1 right. So in the case of s2 you have I is 45 correct and the string n is Rohit this is what we are passing. So then what will happen your s2.id will become 45 and s2.name will become Rohit right under this object s2 your s2.id is 45 and s2.name is Rohit this is what exactly happening and now you are calling two different functions right two functions under s1 and s2 right s1.display. So, that means this will call display function. So, s1 so remember we know s1.id is 18 and s1.name is Virat. So, that will be printed so when s1 is invoking display at line number 12

You can see line number 14, see out ID and name. So that means 18 and Virat will be printing. So similarly, when you are invoking, when s2 is invoking display, so we know that under s2, it is 45 and Rohit. So these will be printed. So when we run the code, what will happen?



```
18  Virat
45  Rohit
```

Right, these two output we will get 18 Virat 45 Rohit. So when you call this function when s1 is calling display it will be printing 18 and Virat and when s2 is invoking display it will be printing 45 and Rohit. So this we will also test in our code right we will go to dev C++ and if you look at code So the same code that we have discussed I have written as a program when we execute compile and run. So you can see that the output 18 and Virat, 45 and Rohit you are getting as the output okay.

I hope it is clear for everyone. So in this program we have first time we introduced member functions as well. So two member functions you have seen. One is insert, another one is display. So now you are more or less clear how to create an object and how this will access a member data that we had seen in the last class and also we have run the code today, right?

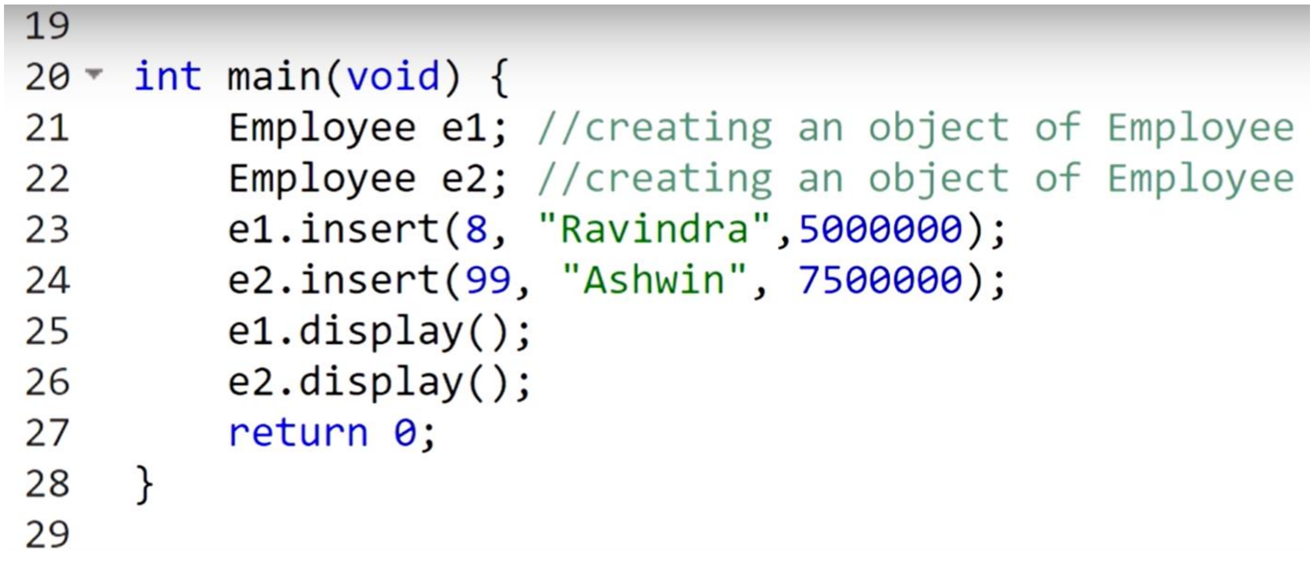
And in this program, we have seen how to write the member function and how the objects are invoking those member functions. So with this, we will go to the next program. So, which you can see storing and displaying certain information, right? For example, here employee information using function, okay? So, here the class is employee.

It is similar to the previous program, all right? So, here the class is employee and here you have two, sorry, in fact, three data members, all right? One is int ID, string name and float salary, okay? So, here also we are having the insert member function and display just the extension of the previous program. So, what I mean to say so here you can find in the case of insert function you have three arguments one is int string float.

So, you can do this is also like a getter and setter function. So, you get the value in the main program and set this into id name and salary. Similarly, we are going to display ID, name and salary, just the extension of the previous program. All right. So the main idea is you have to practice those who are learning first time object oriented.

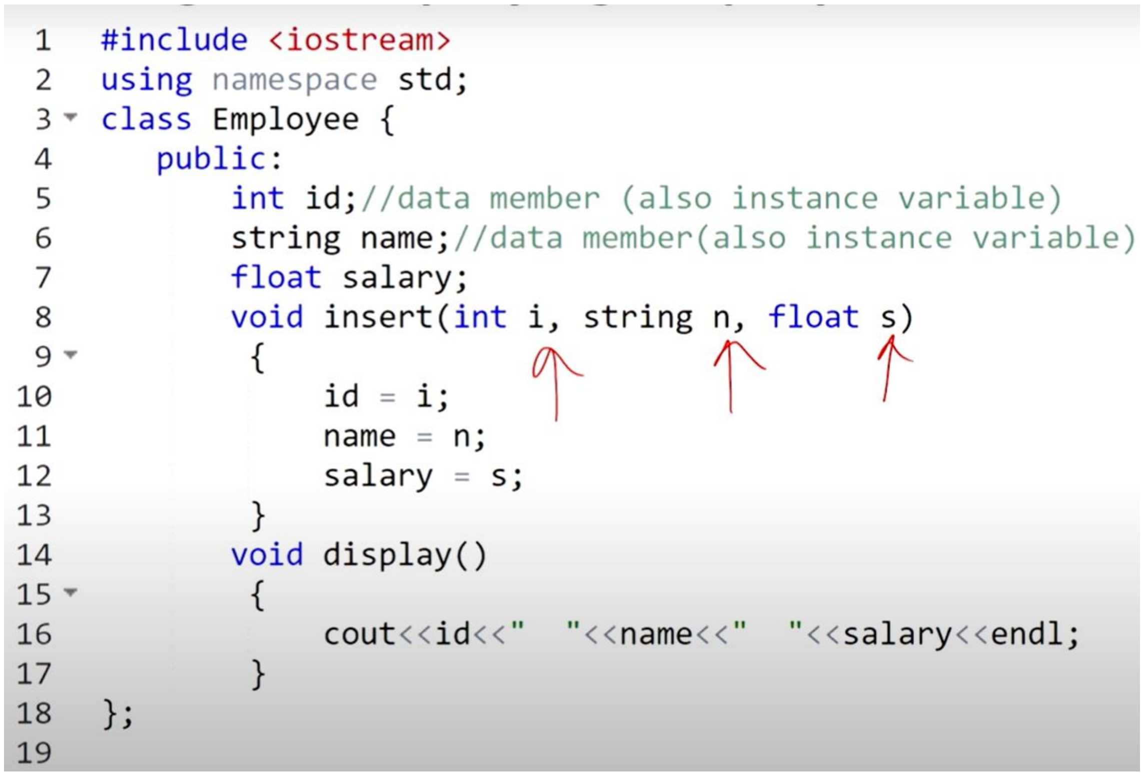
Assume that you are well versed with procedural oriented. So this is like a practicing the problems of the program. So now we have created two objects, right? e1 and e2, right? We have instantiated two objects, e1 and e2.

So here e1.insert I call that means the object e1 is invoking the insert function. So here you have 8 right Ravindra and 50 lakhs right. So the three arguments the three parameters three values we are passing 8 Ravindra let us say 50 lakhs. So what will happen? This insert, right, this e1 which is invoking insert.



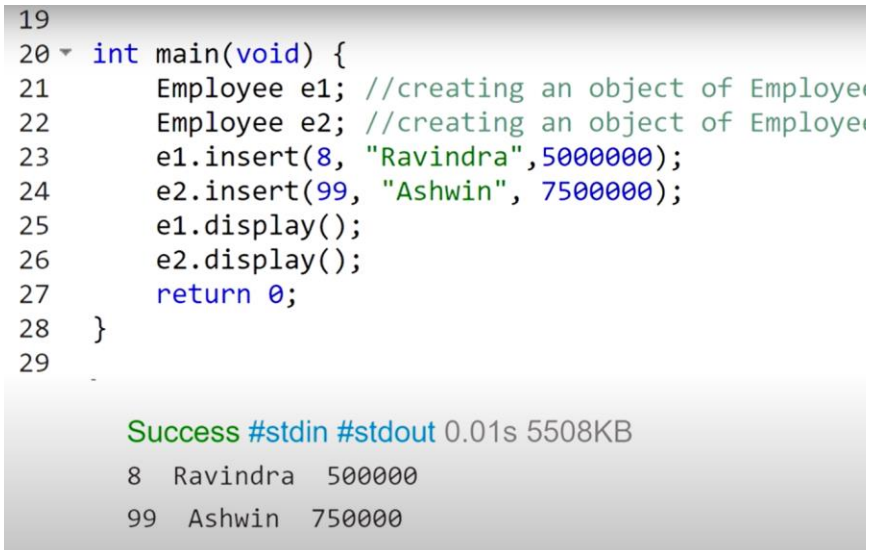
```
int main(void) {
    Employee e1; //creating an object of Employee
    Employee e2; //creating an object of Employee
    e1.insert(8, "Ravindra",5000000);
    e2.insert(99, "Ashwin", 7500000);
    e1.display();
    e2.display();
    return 0;
}
```

So, insert is over here. So, whatever value that we have passed, it will be respectively I, N and S. So, that means when you are assigning I into ID, N into name and S into salary. So, this e1.ID, whatever we have passed, right, what did we pass? 8. right and e1.name so name what did we pass Ravindra and e1.salary 50 lakhs right so when e1 calls this display function these three values will be printed similarly e2 is invoking insert so here you are passing 99 Ashwin and 75 lakhs right so those values when you are calling display Alright that means initially it is assigning when e2 is calling or invoking this insert all these values will be assigned to ID name and salary respectively.



```
#include <iostream>
using namespace std;
class Employee {
   public:
       int id;//data member (also instance variable)
       string name;//data member(also instance variable)
       float salary;
       void insert(int i, string n, float s)
        {
            id = i;
            name = n;
            salary = s;
        }
       void display()
        {
            cout<<id<<"  "<<name<<"  "<<salary<<endl;
        }
};
```

Alright, so then when E2 is invoking correct when E2 is invoking display all these will be printed all these values will be printed. So you can see line number 25 and 26 E1 is invoking display and E2 is also invoking display and you know that right whatever in line number 23 and 24 should be the output right.



```
int main(void) {
    Employee e1; //creating an object of Employe
    Employee e2; //creating an object of Employe
    e1.insert(8, "Ravindra",5000000);
    e2.insert(99, "Ashwin", 7500000);
    e1.display();
    e2.display();
    return 0;
}
```

```
Success #stdin #stdout 0.01s 5508KB
8  Ravindra  500000
99  Ashwin  750000
```

So you can see the output over here right. So this code we will run.

So we will go to Dev C++ and this will be under code 4. So the same right the same program and we if we compile and run I mean you can find that it is giving the output. So whatever output that we have got So you can see that as a output right. I hope it is clear how to use data member or member data and member functions.

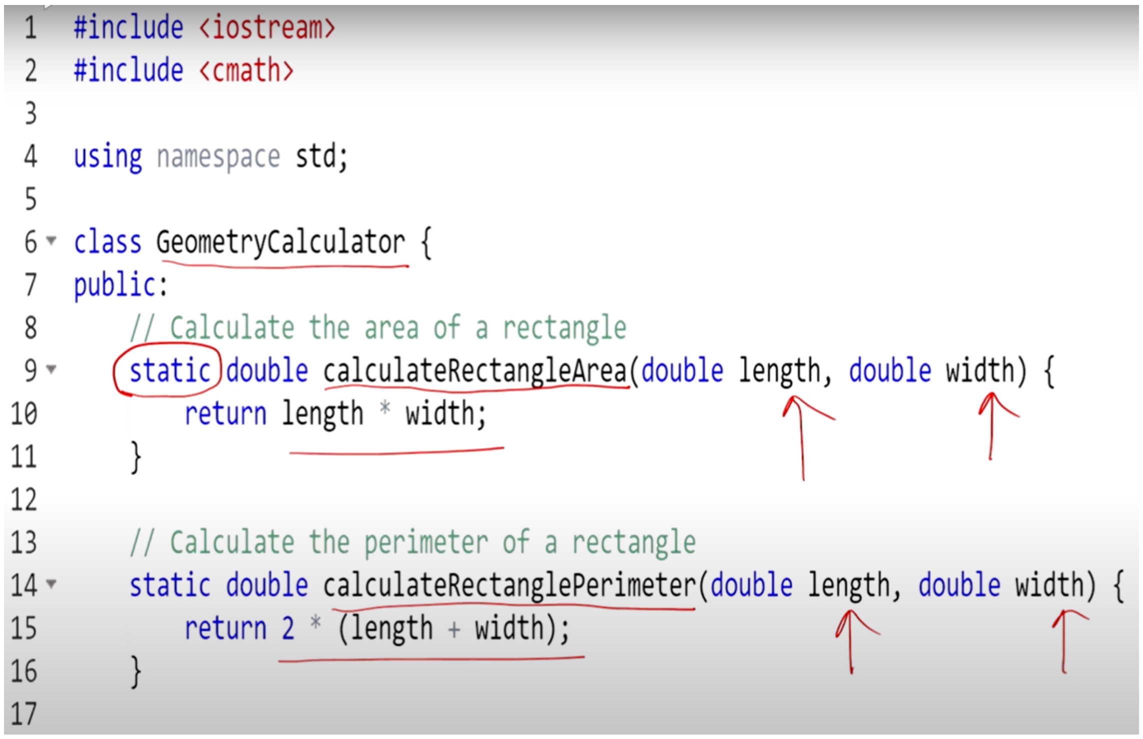
So these are all the examples that we had seen so far. So now you can think of problems like this right. So this is a simple program. Design a C++ class for basic geometry calculations. right you can have functions or methods for calculating area perimeter and other geometric properties of shapes right suppose you are given a program like this you have to write a program right so design a C++ class for basic geometric calculation so you will have a function so you have to write a function that means a member function you have to write so that has to calculate area perimeter and other geometric properties of shapes right

So this we will see how we are going to write. Suppose you are getting a program like this. So here the name of the class is geometry calculator. You can give any name. So what I have chosen?

Geometry calculator. This is the name of the class. So under this, so there are certain terms. Anyway, I will introduce. We know what is double, right?

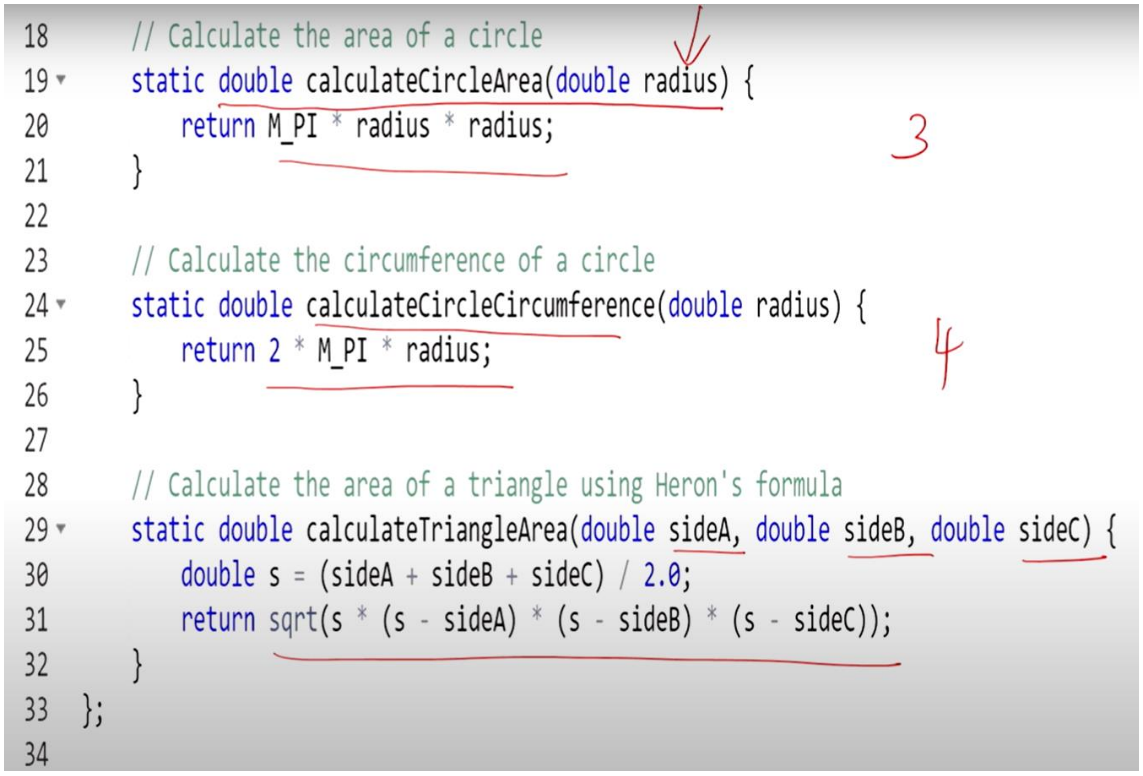
Static. This part I will come back slightly later. No need to worry right now. right. This will be covered.

So now if you look at you have a method, right. So you have the function. In the C++ we call it is a function and in Java you call it as a method, right. So whatever I am saying it will be understood at one point of time it is a method or function you will understand. So right now we are in C++.



```cpp
#include <iostream>
#include <cmath>

using namespace std;

class GeometryCalculator {
public:
    // Calculate the area of a rectangle
    static double calculateRectangleArea(double length, double width) {
        return length * width;
    }

    // Calculate the perimeter of a rectangle
    static double calculateRectanglePerimeter(double length, double width) {
        return 2 * (length + width);
    }

```

So I call this as a member function so here you have the member function the member function is calculateRectangleArea() right so we are passing $length$ and $width$ we are we will pass in the main right the arguments are double as well as double right double comma double right $length$ and $width$ so this will return $length \times width$. So, we know that if you are given a rectangle length into width is nothing but the area So, similarly we have calculateRectanglePerimeter of the rectangle first one is the area second member function is perimeter of the area again $length$ and $width$ as a parameter. So, we know the formula $2 \times (length + width)$. So, that has to return $2 \times (length + width)$ right.



```
    // Calculate the area of a circle
    static double calculateCircleArea(double radius) {
        return M_PI * radius * radius;
    }

    // Calculate the circumference of a circle
    static double calculateCircleCircumference(double radius) {
        return 2 * M_PI * radius;
    }

    // Calculate the area of a triangle using Heron's formula
    static double calculateTriangleArea(double sideA, double sideB, double sideC) {
        double s = (sideA + sideB + sideC) / 2.0;
        return sqrt(s * (s - sideA) * (s - sideB) * (s - sideC));
    }
};
```

So, we will go to the next slide I hope you understand up to here line number 17. So, next geometry is calculate circle area right. So, you have a function double calculate circle area you are passing only radius. So, we know it is $\pi.r^2$ return M_PI underscore pi. So, M_PI is under math.h right or cmath include cmath you can try either math.h or cmath simply cmath you are including

So, the C++ C math right now if you look at the area of the circle pi r square right m pi into radius into radius and calculate circle circumference 2 into pi r $2 \times \pi \times r$, $2 \times M\_PI \times radius$ right. So, that is another function right this is a fourth function. Alright, so this is 4, this is 3, previously we have this is 2 and this is 1. Alright, so we have seen 4 member functions. And the fifth one is calculate triangle area, area of the triangle.

Alright, so when you have $sideA$, $sideB$, and $sideC$, right, so 3 sides will be there for the triangle. So, here you have, so you are calculating S, S is $(sideA + sideB + sideC)/2$. right, $(a + b + c)/$ a and then you have the formulation $sqrt(s \times (s - sideA) \times (s - sideB) \times (s - sideC))$, okay. So, this will give you square root of, already we have included cmath, so that will take care for the square root, right, $sqrt(s \times (s - a) \times (s - b) \times (s - c))$, right, you might have studied. So, we have seen 5 member functions over here and then, so your main program will calculate all.

So, how it works? So, initially, so the usual variable names they are using, length, width, radius, $sideA$, $sideB$, $sideC$, they are all taking the double values. So, this is not very difficult, you might have studied in the basic programming. Procedural oriented programming. So now we are creating an object.

The object is G1. And the class is geometry calculator. This is what we had given. Right. So I created one object.

I instantiated one object. Now this object will invoke calculate rectangle area. So when it is invoking calculate rectangle area and passing length and width. already we have length and width 5.0 and 3.0 right so that means this function this g1 when it invokes this function calculate rectangle area will be called calculate rectangle area is the first function first member function so that means it will return $length \times width$, Right.

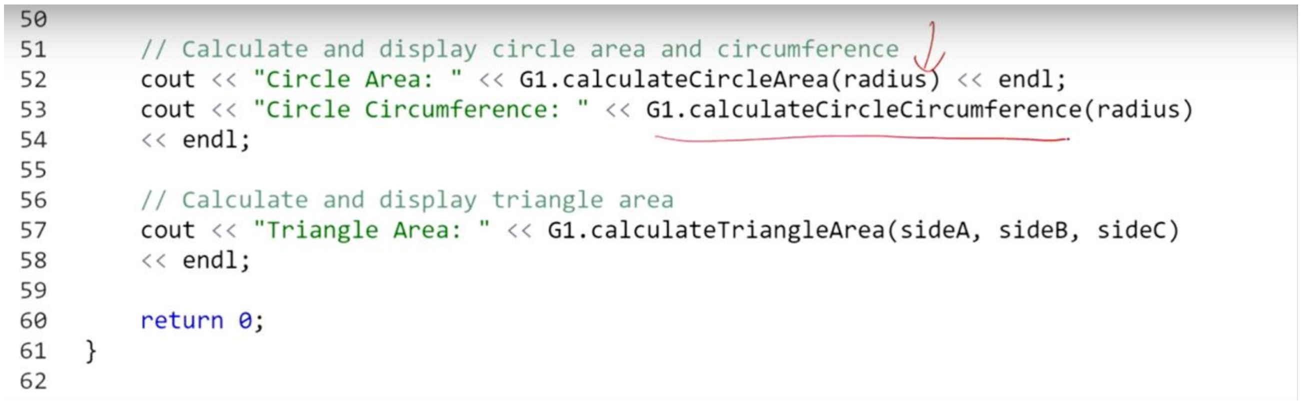
So what is the length we passed? We passed length is 5.0 and width is 3.0. So when you pass both, we have to get 15.0. Right. That will be the output.

Right. So it will print like this. See out rectangle area will be whatever $length \times width$ it is going to return. So will be printed. All right.

So this is what exactly happening. Right. It is returning length into width. Right. Similarly, see out rectangle perimeter.

The next second function. So, this will call rectangle perimeter the second function $2 \times (length + width)$. So, you can calculate substitute length and width add those two and multiply by 2 right this is the second function over G1 is invoking the second member function. And G1 will invoke the third member function called circle right area of the circle calculate circle area you are passing radius. So radius is already there.

Radius is how much? 4.0. So we are passing the value 4.0. It will be calculated when the object is invoking this calculate circle area by passing radius. We will get the resultant.



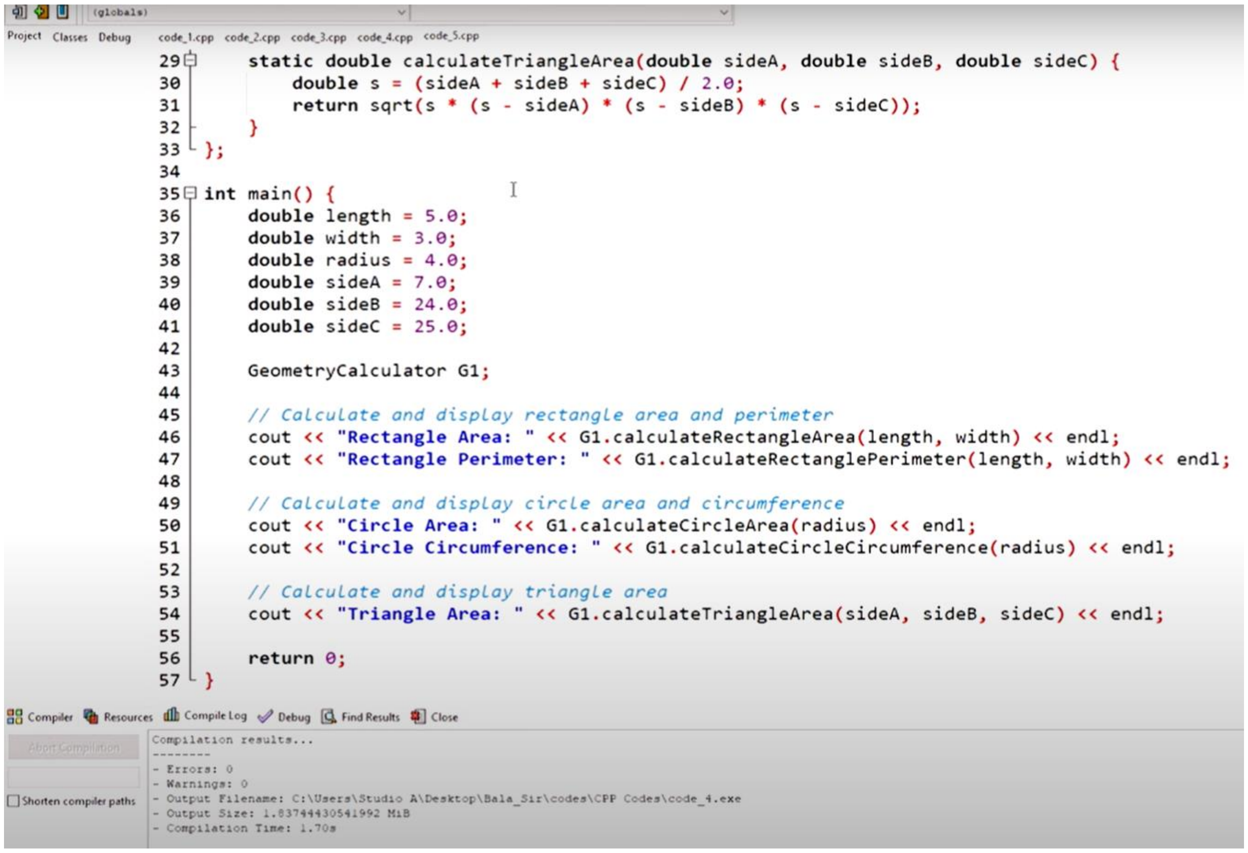
```
    // Calculate and display circle area and circumference
    cout << "Circle Area: " << G1.calculateCircleArea(radius) << endl;
    cout << "Circle Circumference: " << G1.calculateCircleCircumference(radius)
    << endl;

    // Calculate and display triangle area
    cout << "Triangle Area: " << G1.calculateTriangleArea(sideA, sideB, sideC)
    << endl;

    return 0;
}
```

Similarly the fourth function G1 is invoking calculateCircleCircumference(). It will call the fourth member function. Here you go, right? Just calculateCircleCircumference(), right? I hope you understood.

Now, the last function, the fifth one, G1 is invoking calculateTriangleArea(). You are passing $sideA$, $sideB$, and $sideC$, the values, all right? And then we know that, so we have to get the output, right? Because it is returning and then here we are writing all cout statements. The cout statements will print the

resultant or computed values. So if you look at right so we are getting the output like this. So you can verify whatever the values that you are giving if it is a rectangular area that particular formulation is being used in the first member function and it is being computed. Similarly perimeter area of the circle $\pi \times r^2$ circle circumference $2 \times \pi \times r$ and also we have the formulation for triangle area. So, this we got as a output so here if you look so we have 5 member functions only.

So, we have 5 member functions if you look at the class geometry calculator I do not think we have used any member data or data member we have not used any we have used 5 member functions. So, that also you can do that if you are using only member function and this is an example that is also allowed in C++ also in Java any object oriented programming. So, now what we can do we will run this code. So, geometry calculator if you look at line number 9 that is the first member function line number 14 which is the second member function 19, 24 and line number 29 we have fifth member function.

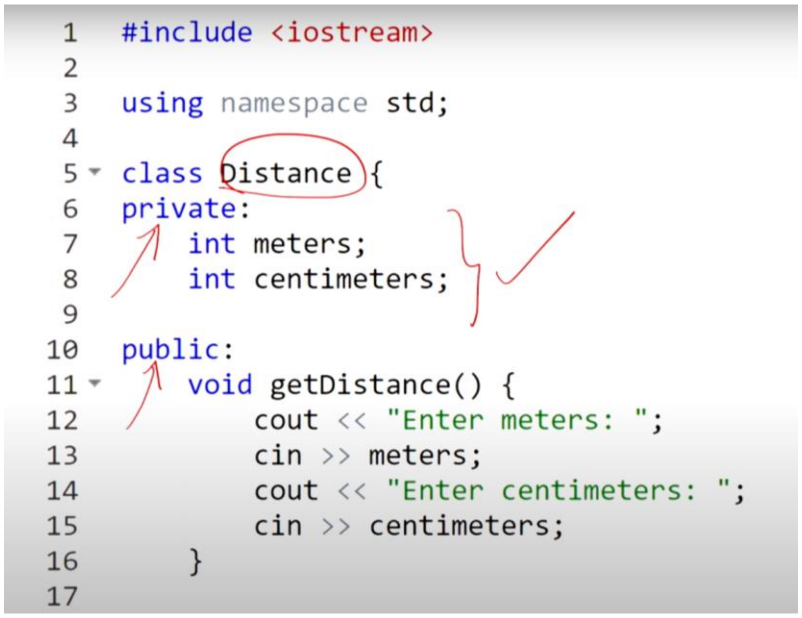


```cpp
29      static double calculateTriangleArea(double sideA, double sideB, double sideC) {
30          double s = (sideA + sideB + sideC) / 2.0;
31          return sqrt(s * (s - sideA) * (s - sideB) * (s - sideC));
32      }
33  };
34
35  int main() {
36      double length = 5.0;
37      double width = 3.0;
38      double radius = 4.0;
39      double sideA = 7.0;
40      double sideB = 24.0;
41      double sideC = 25.0;
42
43      GeometryCalculator G1;
44
45      // Calculate and display rectangle area and perimeter
46      cout << "Rectangle Area: " << G1.calculateRectangleArea(length, width) << endl;
47      cout << "Rectangle Perimeter: " << G1.calculateRectanglePerimeter(length, width) << endl;
48
49      // Calculate and display circle area and circumference
50      cout << "Circle Area: " << G1.calculateCircleArea(radius) << endl;
51      cout << "Circle Circumference: " << G1.calculateCircleCircumference(radius) << endl;
52
53      // Calculate and display triangle area
54      cout << "Triangle Area: " << G1.calculateTriangleArea(sideA, sideB, sideC) << endl;
55
56      return 0;
57  }
```

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Studio A\Desktop\Bala_Sir\codes\CPP Codes\code_4.exe
- Output Size: 1.83744430541992 MiB
- Compilation Time: 1.70s
```

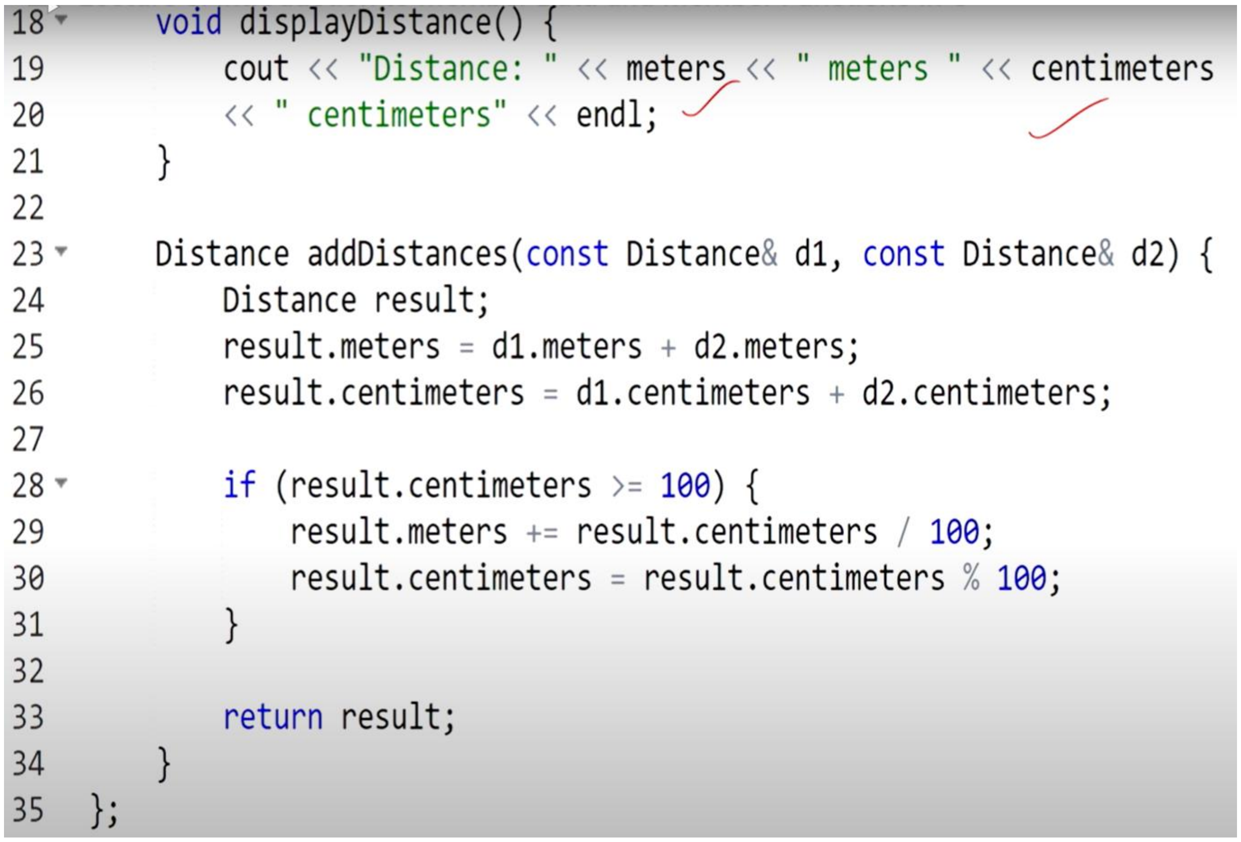
So, we have five member functions and you can see the main So, main you have define all the variables length, width, radius, $sideA$, $sideB$, and $sideC$ and then line number 46, 47, 50, 51, 54 we are printing the area of the rectangle, perimeter of the rectangle, area of the circle, circumference of the circle and triangle area. So G1 is invoking all these 5 member functions and when you run the code. You will get whatever output that we had seen you can able to see here okay. So this is how we can only use the member function. So this is an example.

It is a good example there is a possibility or you may ask the question can we use only functions so you can do that so we will see next example so here write a C++ program using class and objects to add two distances given in meter and centimeter so you are given two distances so one distance in some meter. Let us say 8 meter 52 centimeters another one is 9 meters 72 centimeters so when you add both When you are adding both. So what is the output that you are going to get? So here we use two member data, meters and centimeters. So now four or five programs we had seen when you are now familiar with the member data and member functions.



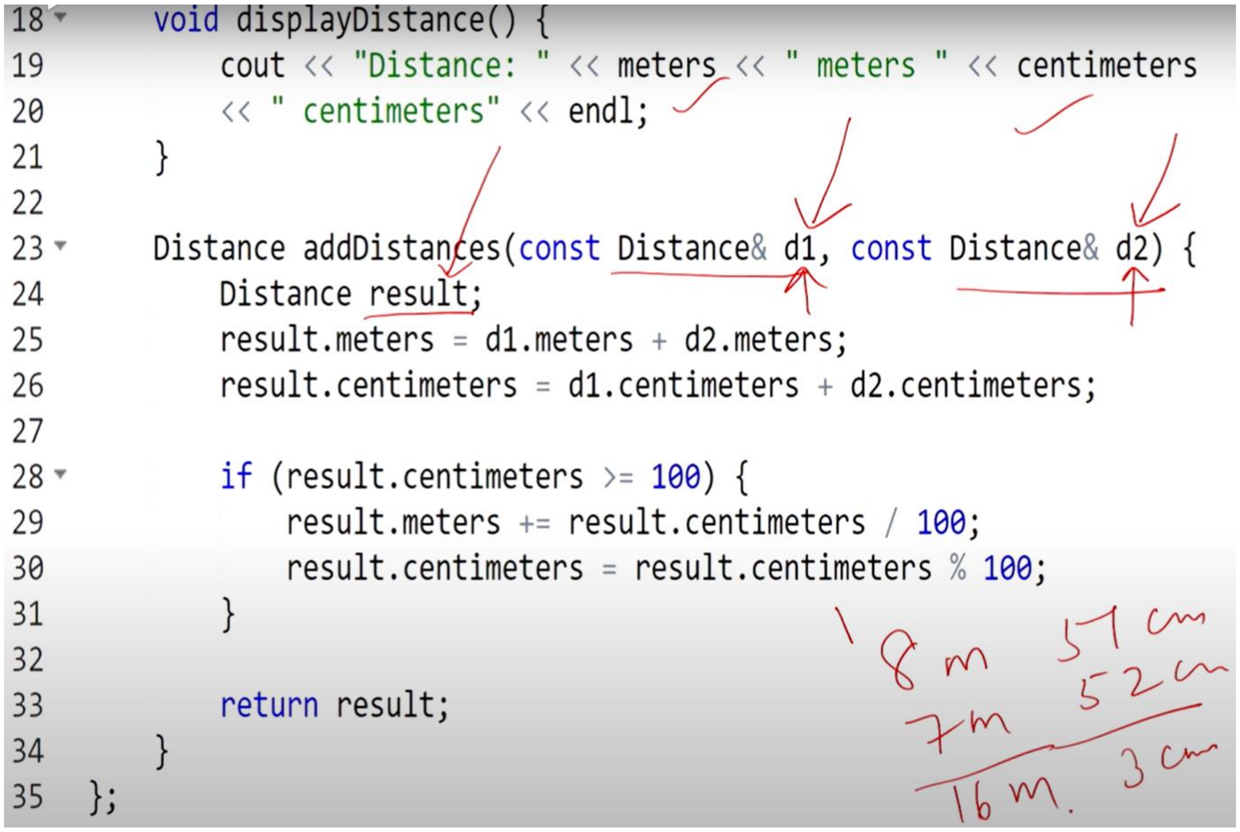
```cpp
#include <iostream>

using namespace std;

class Distance {
private:
    int meters;
    int centimeters;

public:
    void getDistance() {
        cout << "Enter meters: ";
        cin >> meters;
        cout << "Enter centimeters: ";
        cin >> centimeters;
    }

```

So the class is distance. Right now no need to worry about the access modifier, private and public. So, we will explain we will write this is the introductory class of classes and objects. So, no need to worry right now. So, we have two member data meters and centimeters and then you have a member function called get distance.



```
    void displayDistance() {
        cout << "Distance: " << meters << " meters " << centimeters
        << " centimeters" << endl;
    }

    Distance addDistances(const Distance& d1, const Distance& d2) {
        Distance result;
        result.meters = d1.meters + d2.meters;
        result.centimeters = d1.centimeters + d2.centimeters;

        if (result.centimeters >= 100) {
            result.meters += result.centimeters / 100;
            result.centimeters = result.centimeters % 100;
        }

        return result;
    }
};
```

You are taking the input from the users the user has to give meters and centimeters. So we create an object and that object will invoke getDistance() means you have to pass two values one is under meter another one is under centimeter. So now display will display meters and centimeters right display will display the distance of meters and centimeters. So now we can add two distances. You might have studied call by reference. In basic programming, all right anyway; when we talk about the methods or functions, I will introduce them here; in this case, what we are doing is based on the call-by reference address operator. So here you are having two distances d1 and d2 the references will be passed all right so when I try to print let us say I want to print c out d1



```
    void displayDistance() {
        cout << "Distance: " << meters << " meters " << centimeters
        << " centimeters" << endl;
    }

    Distance addDistances(const Distance& d1, const Distance& d2) {
        Distance result;
        result.meters = d1.meters + d2.meters;
        result.centimeters = d1.centimeters + d2.centimeters;

        if (result.centimeters >= 100) {
            result.meters += result.centimeters / 100;
            result.centimeters = result.centimeters % 100;
        }

        return result;
    }
};
```

So D1 will print if I just put cout<<d1 print the address of this d1 and when I put cout<<d2 that is nothing but address of d2. So now what we will do we will add these two distances. So distance result we will have one more right one more object result. So one object is d1 another object is d2 the third object is result. So we know that as I said 8 meters and let us say 51 centimeters.

Another one is 7 meters and 52 centimeters. So we know that if you add these two, if it goes beyond 100, right? 100 centimeters is nothing but 1 meter, right? So here in this case, what will happen? 1 or 3 we will get, correct?

So that means it will be 3 centimeters. 1 will be added with this. So 1 + 8 + 7, right? What is that? 16.

So we should get 16 meters and 3 centimeters if we add these two distances. So now logically you have to write. So first what we are doing, the d1.meters and d2.meters will be added. So that will be stored in result dot meters. So for example, 8 plus 7.

So 8 plus 7, 15 will be there up to line number 25. Then centimeters I am adding separately. 1 plus 2 is 3. 5 plus 5 is 10. 103.

103 centimeters. So, separately, I have added meters and centimeters. Now if it is going beyond 100. If it is going beyond 100. So we know that.

When it goes beyond 100, it means 1 should go because 100 centimeters is equal to 1 meter. So that will go over as a meter. So that is exactly what we have to do in logic.

All right. So if it is going beyond 100. So what we are doing? We are doing the integer division by 100.

So we are doing the integer division by 100. How about 3 by 100? How much? 1. So the 1 will be added with this 15 meters 1 will be added with this 15 meters 1 or 3 by 100 1 integer division both are integers integer division you will get 1.

So that means the result dot meters will be previously 15, 15 + 1. How much 16 this you will get, and then you are using a mod operator result dot centimeters mod 100. 103 mod 100 reminder is 3. So, 3 will be right will be displayed 3 is the output for centimeters. I hope you understood so for the result object result.meter is 16 and result.centimeters is 3. So here we are using the logic right and then this will return result because the return type is also the object. return type is also the object, yes, which is possible, right? We will extensively study this, so right now, in this introductory lecture, you have to worry about the objects and classes extensively we have done. There are some syntax anyway we will be covering in the due course. So when you run the code in the case of the main program, we have an object distance of 1 second and an object distance of 2, and you have a result.

So here line number 41 it is asking line number 40 it is asking enter the first distance distance 1 dot get distance. So you are sending something like exactly what we have done 8 meters 51 centimeter all right. So this we will give as a input because this is invoking when get distance is getting invoked so you can see that get distance what is happening you have to enter meters and centimeters correct so you have to enter meters and centimeters assume that I entered 8 and 51 understood 8 is a meter and 51 is a centimeter similarly 7 and

52 same example we took right understood meters right so now. This result object.

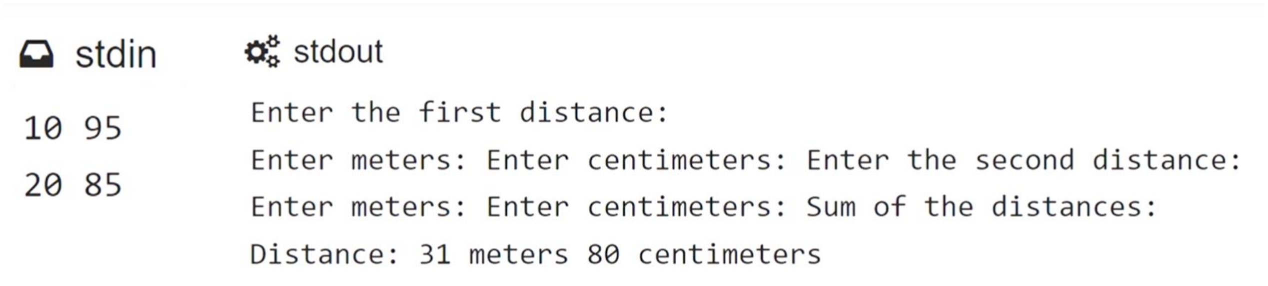
We have already defined over here. This result object. Will invoke addDistance. So you are passing. Distance 1 and distance 2.

Right. So distance 1. Distance 1. You are passing. Right.

So when you are passing distance 1. This is the object. So here you have a reference. Right. So distance 1.

address d1, which is equal to what we are passing distance 1. So this is the exact syntax. This you might have studied in the basic programming right procedural oriented. So you might have studied int address A equal to B. So the meaning is both are pointing to the same address both your variables A and B are pointing to the same address. in a similar way the d1 distance d1 object and distance 1 object both are pointing to the same address similarly for d2, and then you know the procedure, we are adding meters we are adding centimeters and then we have the logic over here suppose I pass these two values we also worked out we are going to get this as the output. So in the main program when

You have get distance, distance 2 will also have a get distance. These you are passing, right? And then when I add, right? By passing this addition of these two will be stored in the result object, right? This will be a result object. So that means result dot meters, result dot centimeters will be displayed. Once line number 48, this object result will invoke this particular function, right?



```
10 95
20 85
```

```
Enter the first distance:
Enter meters: Enter centimeters: Enter the second distance:
Enter meters: Enter centimeters: Sum of the distances:
Distance: 31 meters 80 centimeters
```

Alright so this will be your output. So this you are getting as a output correct if you are passing (10 95) and (20 85) you add this you should get 31 meters and 80 centimeters so this we will cross-check in your program also alright so here you can see when I run the code I have to get the output like this. So it is asking enter meters 34 meters I enteR yes it is asking centimeters fine that is for the first distance and the second distance enter meters enter centimeter and then you will get the final answer you can verify you see that what is the meters and centimeters So, you can get the output 45 + 67 naturally will go beyond 100 and the reminder when you put 1.

So, 34 + 23 + 1 you got 58 meters. So, the program is running fine with this I conclude this lecture and in the next lecture we will continue with classes and objects in Java. Thank you.