

FUNDAMENTALS OF OBJECT ORIENTED PROGRAMMING

Lecture06

Lecture 6: Classes and Objects in C++

So, welcome to Lecture 6: Classes and Objects. So, we briefly defined how we can create classes in C++, alright. So, we have also studied that it is nothing but the blueprint for creating objects, alright. So, once you create a class, right, assume that you are creating a class, so it encapsulates the data, alright.

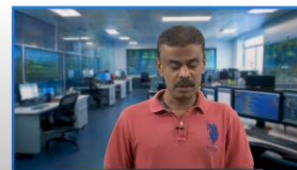
So, in fact, data. That means attributes and the behavior, which is called functions or methods. Alright. So, in set notation. Alright.

Defining Classes in C++

- Classes are blueprints for creating objects. They encapsulate data (attributes) and behavior (methods). Formally, a class C can be defined as a tuple: $C = (A, M)$ where:
 - ► A is the set of attributes
 - ► M is the set of methods

In C++, a class can be defined using the class keyword. The syntax is as follows:

```
class ClassName {  
    // Attributes (or Member Variables)  
    // Methods (or Member Functions)  
};
```



So, we can define this as a tuple. Alright. So, here it is nothing but a tuple. C is equal to A, M , where A we call it as a set of attributes. Suppose you are using the data.

Right. You call it as member data. And M is nothing but a member function. Or you call it as methods. So, we have already talked about how the syntax works in.

C++, so when I define a class, right, so here you go, you will write the keyword called class, right. So, we will have a class keyword and then the name of the class. You are opening the braces; the code block is starting over here, and you have the member data called attributes, and you will have the functions. You can have several attributes; you can use arrays also, all right. And you can have several methods, several functions, right? You call it as member data or member variable or member function. And then you will have the end parenthesis with a semicolon. In the case of C++, you should have the semicolon.

This is how we can define a class in C++. So, what is the motivation behind, all right, for class design in C++? So, when we are writing object-oriented programming, as we had seen, it is mainly based on the objects and classes, right, which is a very powerful paradigm. So, this can solve a wide variety of problems. So, you can have the list of problems over here.

- Object-oriented programming (OOP) using objects and classes is a powerful paradigm that can solve a wide range of problems. Below is a list of problem types and scenarios where OOP concepts (objects, classes, inheritance, polymorphism, encapsulation, etc.) can be effectively applied:
- **Library Management System:** Books, users, loans, and staff members can be modeled as classes, with attributes like book titles, due dates, and behaviors like borrowing or returning books.
- **Banking System:** Classes can represent customers, accounts, and transactions. Methods can include deposit, withdrawal, and balance inquiry.



So, the list of problems when you are having a class, you can have an object. So, class and from the class, you can create the object, and we can use some of the object-oriented programming concepts like inheritance, polymorphism,

encapsulation, etc. So, all these can be effectively applied. We can apply. So, now we have seen

Right, in the few classes back, we had seen the tangible objects and intangible objects. So, for example, right, so intangible objects, assume that I want to create a class or I have to solve the problem using object-oriented programming for library management systems. So, when you have library management systems, you can think of, right, books, users, right, loans, Staff members. Right.

So these are your models. I mean, you are modeling these as classes. Right. You are modeling these as classes. Like, then we have attributes.

For example, book title. Right. Due dates. And behaviors like borrowing or returning the books. Right.

So the attributes. I define book title. You call it as a member data. Then we can have what are all the due dates. And then, if I want to write certain functions.

Like borrowing or returning the books. So this is how I can think of the blueprint for the library management system. Similarly, the banking system. So classes can represent customers, accounts, transactions. Methods can include deposit, withdrawal, and balance inquiry.

So, when intangible objects like banking systems are involved, how can I have a class? From the class, I mean I can represent the customer's account and transactions, and then I will have a set of member functions, all right. So, like deposit, withdrawal, balance inquiry, etc. So, this is how I can think of writing object-oriented code for these kinds of problems. Similarly, suppose you are interested in, let us say, gaming. So, I can think of classes for pieces like pawn, king, rook, queen.

Because they have specific movements and behaviors. And also, I can think of the board. The board can be an object. So, that can have all the pieces. So, when you are writing any games, right?

- **Chess Game:** Classes for pieces like pawn, king, or queen, each with specific movements and behaviors. The board can be an object that holds the pieces.
- **Traffic Simulation:** Vehicles, roads, and traffic lights can be objects. Classes can define the behavior of each component, and methods simulate movement and traffic flow.
- **Linked Lists, Stacks, Queues, Trees:** Classes can represent nodes and the structure itself, with methods to add, remove, or manipulate data.
- **Neural Networks:** Layers, neurons, and connections can be modeled as classes, with methods for forward propagation, backpropagation, and training.



Chess is one of the games. So, if you want to have other board games or any other games, you can think of them in an object-oriented way. Similarly, traffic simulations, right? So, we can think of vehicles, roads, traffic lights. So, these can be objects, right?

So, when I am talking about the classes of this problem, right? So, the classes can define the behavior of every component of this traffic simulation. And the methods simulate movement and traffic flow. So, when I am thinking about traffic simulation, I can write a program using object-oriented programming. Similarly, when you are going for data structures like linked lists, stacks, queues, and trees, right?

So, once you are done with, let us say, C++ and Java, you can think of the next one as data structures and advanced data structures. So, I can define a class node, right? Class linked list, class stack, class queue, class binary search trees, class AVL trees, right? So, these are all nothing but classes. And you can have, right, several methods, right?

Insertion, right, or addition of the node, removal of the node, or I want to manipulate certain data. So, these are all coming under the methods, right? So, you can think of whenever I have a linked list or any data structure. So, the data structure can be a class and then the several operations. You can have several operations.

Those operations are nothing but the methods or member functions, right? Similarly, we are in the artificial intelligence paradigm, right? In neural networks,

we can use object-oriented programming like layers, neurons, the connections, right? So, these can all be considered as classes.

And then, when I am talking about the methods, you have forward propagation, backward propagation, and you have to train the model, correct? So, when I want to have object-oriented concepts in neural networks, you can think of it. So, The main idea of all these problems that I have suggested, right? So, you have a class.

So from the class, you can have the objects, right? So once you have the objects, let us assume inside the class, you can have a set of member data. You can use any data. It may be an integer, double, float, or even an array of integers, right? You can think of these as all the member data and several member functions.

When I am talking about the data structure, I have to write a method for insertion, deletion, update, pop operation in stacks, peek operation in stack, and push operation in stack. So all these operations, when I am talking about the operations, they are nothing but the methods. So this is how I can think of using object-oriented programming, how I can solve these kinds of problems. So similar to class design in C++, we have class design in Java. So when we talk about the class design in Java, here you go.

So you have the name of the class. So here you have the name of the class, the class name. And then you have attributes and methods, right? Attributes are nothing but data members. So in Java, you call them data members.

And then you can have a set of methods. The methods are nothing but the behaviors, operations, or the functions that you call in C++. You call this in Java. So the class is the keyword, and you can have a name of the class. You can have an attribute.

You can have a method. So the only thing after braces is you should not have a semicolon. Okay. So this is how we can define a class in Java. So now you know how to create a class, how to define an object, right?

Or how to declare an object, or I can say, how to instantiate an object, right? Right? Instantiate an object or instantiate the objects. Right? So these you know that.

So with this, we will see some of the problems. Okay? So now write a C++ program to multiply two complex numbers using class and objects. Right? So you have to multiply two complex numbers.

Problem

Write a C++ program to multiply two complex numbers using class and objects

$$\begin{aligned}C_1 &= a_1 + ib_1 \\C_2 &= a_2 + ib_2 \\C_1 \times C_2 &= (a_1 + ib_1) \times (a_2 + ib_2) \\&= (a_1 a_2 - b_1 b_2) + i(b_1 a_2 + b_2 a_1)\end{aligned}$$



So how do you define a complex number? So C is equal to let us say c_1 . $a_1 + ib_1$. So, this is one complex number. And c_2 is equal to $a_2 + ib_2$, right?

When I multiply $c_1 \times c_2$, it is nothing but $a_1 + ib_1$ multiplied by $a_2 + ib_2$, correct? So, which is nothing but $a_1 \times a_2$. $i^2 = -1$. Therefore, I will get minus $b_1 \times b_2$, correct? Plus I into...

$b_1.a_2 + a_1 .b_2$ or I can write $b_2. a_1$, right? So, the multiplication when I am doing the product, right? So, I have to get this as the output. So, these two numbers will be the input. So, it is a pair.

The pair will be the input, and the program has to multiply these two complex numbers, and I have to get the output like this. So, now the user will give you the numbers $a_1.b_1$, the pair $a_2.b_2$, the pair, and the resultant I have to get this. So, how are we going to solve this using object-oriented programming? So, let us consider this program. Alright.

```

1  #include <iostream>
2  using namespace std;
3
4  // Define a class to represent a complex number
5  class Complex {
6      float real;
7      float imag;
8
9      public:
10         // Function to set the values of real and imaginary parts
11         void setValues(float r, float i) {
12             real = r;
13             imag = i;
14         }
15

```

Handwritten notes: $C = a + ib$ with arrows pointing to 'a' and 'b'. A circled '1' is next to the `setValues` function.



So, let us have the class Complex. Right. I am creating a class Complex. The idea is we have to multiply these two complex numbers. Right.

I am creating a complex. Class complex. So, when I write C equals a+ib. Right. Here, A is a real value, and B is i x b, which we can call a complex.

Imaginary value. Alright. So, imaginary value. Therefore, we have float real and float imaginary as member data. So, you have two member data.

The member data are float real and float imaginary. Next, you have a set of member functions, all right. You have a member function called set values. Usually, we see these getter and setter functions. So, set values.

Two parameters: float R and float I, right? So, whatever you are setting the value or taking the value from the user, real and imaginary, right? So, here r will be assigned to real, and I will be assigned to imaginary, right? Sooner or later, we will talk about the access modifiers, private and public. So, right now, it is still a keyword.

I have not explained. So, once we talk about it, we will again see the complete program. So, right now, there is no need to worry about the private and public. So, that is the reason here, r will be assigned to real, and I will be assigned to imaginary. So, I can access this inside the class.


That is the meaning. So, when I am taking the input from the user, right, r and i, so based on this function, when I am passing these values in the main, so now r and i will have the value, so that will be assigned to real and imaginary. So, this

is one function. And then, you have another function called multiply function. So, here we use the keyword static.

```
16 // Static function to multiply two complex numbers using two arguments
17 static Complex multiply(const Complex &c1, const Complex &c2) {
18     Complex result;
19     result.real = c1.real * c2.real - c1.imag * c2.imag;
20     result.imag = c1.real * c2.imag + c1.imag * c2.real;
21     return result;
22 }
23
24 // Function to display the complex number
25 void display() {
26     if (imag >= 0)
27         cout << real << " + " << imag << "i" << endl;
28     else
29         cout << real << " - " << -imag << "i" << endl;
30 }
31 };
32
```

int x c1

*** Return Type is Object of Class Complex**



I will explain about static. Maybe we will see one or two examples. How we are defining the keyword static. So, now complex multiply. So, here you have the multiply function.

Whose return type is the object? Your return type is the class complex. So, I will take complex address c1. And another one is complex address c2, right? So, suppose I have, if you would have done the reference operator int address c1, right?

Assume that I have an int, let us say c2. Equal to 10, right. So, when I write int c2 equal to 10, we know that, let us say, this will be stored in some address. Assume that the address is having, let us assume this is your memory location, right. So, this is c2 taking the value 10. So now, if I put int address c1 which is equal to c2, right?

So now, if you look, what is the address of c2? The address of c2 is 4002. The address of c2 is 4002, correct? So now, when I write like this, int address c1 equal to c2, the meaning is address c1 is also pointing to the location 4002, right? So that is the meaning of this.

Because we are using this syntax over here, all right. So, maybe the syntax goes like this, complex when we are calling the function. So, you will give like this, some let us say c11, all right. So, the meaning is address c1 and address c11 are both the same, right. When you are doing like this, address c1 and address c11 are both the same, address of c11.

And the address of c11, both are the same, right? So, with this in mind, we will again come back when we are calling this particular function in line number 17, right? So, this concept will be useful to you. So, now as we worked out, right? How to multiply two complex numbers, right?

Maybe one more time, I will do it. $a_1 + ib_1$, because here we have a set of expressions, right? Into $a_2 + ib_2$, right? So, we know that it is a_1 , a_2 minus b_1 , b_2 ; i squared is minus 1, right? This will be your real plus i into $a_1.b_2 + a_2.b_1$, right?

So, now if you look, this is the resultant, right? The resultant complex number, let us say C3. So, C3 will be, it will be like $a + ib$ form, right? C3 will be some X plus iY form, right? What is X?

That is real. So, X will be; let us say C3 is equal to X plus iY, alright. In that case, $X = a_1a_2 - b_1b_2$, and $Y = a_1b_2 + a_2b_1$, alright. So, this you call it as result dot real. X is nothing but your result dot real.

So, you can see that C1 dot real into C2 dot real, which is nothing but $a_1 \times a_2$. And then you have minus; you have $-b_1 \times b_2$, C1 dot imaginary into C2 dot imaginary, $b_1.b_2$, and imaginary, $c_1.real$, a_1 into $c_2.imaginary$, b_2 , right. Similarly, $c_1.imaginary \times c_2.real$, right? So, that you have evaluated this. And then return result. Alright.

So, the result is a complex number. You can see the return type is complex. Alright. So, this is one main function. So, you have another member function over here.

And the last member function is displaying. Alright. You are displaying real and imaginary. Alright. So, if imaginary is greater than or equal to 0, you display $a + ib$.


```

33 int main() {
34     // Declare two objects for the complex numbers
35     Complex c1, c2, result;
36
37     // Set values for two complex numbers using two arguments
38     float real1, imag1, real2, imag2;
39
40     cout << "Enter real and imaginary part of first complex number: ";
41     cin >> real1 >> imag1;
42     c1.setValues(real1, imag1);
43
44     cout << "Enter real and imaginary part of second complex number: ";
45     cin >> real2 >> imag2;
46     c2.setValues(real2, imag2);

```

Handwritten notes and annotations on the code:

- Red circles around `c1`, `c2`, and `result` in line 35.
- Red arrows pointing from `real1` and `imag1` in line 41 to `c1.setValues` in line 42.
- Red arrows pointing from `real2` and `imag2` in line 45 to `c2.setValues` in line 46.
- Red handwritten text: $result = c1 \times c2$ with an arrow pointing down to $a_1 + ib_1$.
- Red handwritten text: $2 + 3i$ with an arrow pointing to line 42.
- Red handwritten text: $(4) + (5)i$ with arrows pointing to line 46.



Otherwise, $A - iB$. So, that is the reason. That is why it is being taken like this. Right. So, basically, you are displaying $A + iB$ or $X + iY$ form, okay.

So, up to here, we have written the class, correct. So, this is how we define the class called complex. So, you have real and imaginary data members or member data, and then you have member functions called set values, multiply, and then display. So, now, come to the main program.

So, in the main program, we are creating two objects, C_1 and C_2 , and then you need another object called result, right. So, you obviously require result to be equal to $c_1 \times c_2$, where c_1 and c_2 are complex numbers. So, result is the third object, right. So now, You will have $c_1 = a_1 + ib_1$.

So, you need real 1 and imaginary 1. And $c_2 = a_2 + ib_2$. So, real 2 and imaginary 2. So, take the input from the user. For example, I write $2 + 3i$.

And then another one is $4 + 5i$. Right? In that case, I have to write C in real one, C in imaginary one, alright, and then you will have `c1.setValues()`. So now you are calling the object `c1`, which you have already defined over here.


So, the object `c1` is invoking the member function. The member function called set values using the dot operator. So, you have the object, the object is invoking using the dot operator and one of the member functions called set values. You are passing real one and imaginary one. You are passing real one and imaginary one.

```

47
48 // Multiply the two complex numbers using two arguments
49 result = Complex::multiply(c1, c2);
50
51 // Display the result
52 cout << "Result of multiplication: ";
53 result.display();
54
55 return 0;
56 }
57

```

$(2+3i) \times (4+5i)$ $(8-15) + i(12+10)$
 $-7 + 22i$
 Result of multiplication: $-7 + 22i$



So, let us say what is happening in the setValues(), right? So, here you have set values. Assume that I am passing 2 and 3, all right? So, your real is 2 and imaginary will be 3, right? For the object c1, your real is 2 and imaginary is 3.

So, this is the meaning. And then, similarly, c2 is invoking, right? You are taking 2 more inputs, real 2 and imaginary 2, right? So, this is your real 2, this is your imaginary 2, right? So, now c2 is invoking the set values member function.

Right? You have a dot operator and then you are passing real 2 and imaginary 2. Assume that I am passing 4 and 5, correct? So, when I am passing 4 and 5, what will I get? My real will be 4, imaginary will be 5, right?

c2.real will be 4, c2.imaginary will be 5. So, this is the meaning, alright? So, finally, So, here under the class complex, you have the scope resolution operator and then you are having multiply c1, c2. So, this means I can write this function outside the class.

So, when I am writing this in the main, I can do that. So, that means I am calling this particular function, multiply. I am calling this function, multiply. Correct? With passing two arguments, c1 and c2.

Alright? So, when I am calling multiply, let us see what is happening. So, here you go, multiply. So, complex address c1 is equal to c1. c1 that you are passing from here.

Right? c1.real, C1 dot imaginary. Similarly, complex c2 is equal to c2. Alright? So, that means address c1 and address c2.

So, here c1 Your c1 over here, right? You have an object c1 in the main, right? So, you have object c1 in the main. Object c2 is also in the main, correct?

So, which is you are creating an object with some address value, right? So, for example, I create c1 whose address is, let us say, 4002, the address of c1, and I create c2. Whose address is 4006. So, once I have this, when I am calling this, your address c1 equals c1. So that means, c1 when I define inside the function, c2 I define inside the function, both will have the same address as what you have defined outside the class, that is the meaning.

So, now the rest is fine. c1, c2, you pass that means c1.real, c2 imaginary. c1 imaginary, right, and c2 real, all you have passed, and then this computation will be done. So, once this computation is done, we know that, right. So, the complex multiplication is happening, and then you are displaying the result. So, you are getting a result, right.

Your result is the object whose class is complex, right. So, therefore, you have a complex class. This is called the scope resolution operator. And then you are calling the multiply function by passing two arguments, c1 and c2, right? So, multiplication will be done when you are calling this function, and the result is invoking.

```


47
48 // Multiply the two complex numbers using two arguments
49 result = Complex::multiply(c1, c2);
50
51 // Display the result
52 cout << "Result of multiplication: ";
53 result.display();
54
55 return 0;
56 }
57

```

$(2+3i) * (4+5i)$ $(8-15) + i(12+10)$
 $-7 + 22i$

2 3
 4 5

Result of multiplication: $-7 + 22i$



So, the result of multiplication, this is invoking display, right? So that means, result dot real and result dot imaginary will be displayed, right? So, this is what exactly is happening, right? So, result dot real and result dot imaginary will be displayed, right? For example, assume that we are giving 2, 3, right, $2 + 3i$, the same, right, $2 + 3i$ as one input, and another input is $4 + 5i$, correct.

So, when I multiply these two, what will I get? So, I should get minus 7 + 22i. This is obvious, right? $2 \times 4 = 8$, $3 \times 5 = 15$, i squared minus 15 plus i into 3 into 4 is 12, and 5 into 2 is 10.

So, $8 - 15 = -7$, $12i + 10i = 22i$. So, I should get this as output. In fact, this is giving the correct output, right? So, this is how we are doing the complex multiplication using class and objects, all right.

So, what exactly have we done? So, we have created a class, right? So, the class is complex, the name of the class is complex, and then you have member data. So, real and imaginary are member data, and then you have a set of member functions called set values, all right, and then you have multiply, which is the main function, right? The main member function is called multiplication. And then displaying, right?

So, you can display maybe the number 1, c1, complex 1, $a_1 + ib_1$, complex 2, $a_2 + ib_2$. So, this you can display, right? And then result. So, once you are doing

the complex multiplication, right? So, once you are doing a complex multiplication, the resultant can also be displayed using this.

So, that means whenever you are creating an object, you have member data and member functions. So here, we have used two member data and then three member functions. So up to here, line number 31, you have a class. So once you have a class, when you go to the main program, you are creating objects. So $c1$, $c2$, and $result$ are all objects.

So you know how to create a class, and now you are instantiating an object or instantiating objects. So the objects are nothing but $c1$, $c2$, and $result$. So you have the complex class and then the usual float. Real 1, imaginary 1, real 2, imaginary 2. That is the usual data type.

So that you have used in your procedural-oriented programming. So then you are taking the input from the user. So when you are taking the input from the user, the set values will be set. Where $c1$ is invoking set value, $c2$ is invoking set value, and it is nothing but you have $a_1 + ib_1$ and $a_2 + ib_2$, and then you are calling the multiplication function. So here, I am using the reference operator.

So the reference operator $c1$, when you define the object in the main, $c1$, you are defining the object inside the class. So both are pointing to the same location, alright. So this is how we have seen the concept of that integer, alright. Integer address a or integer address $c1$ equal to sub $c11$. So both $c1$ and $c11$ are pointing to the same address, correct.

And then when it is multiplying invoking, alright. In fact, the multiply when you are calling this function with $c1$ and $c2$ passing the arguments. So, these two will be multiplied. $c1$ and $c2$ will be multiplied and the resultant will be over here, right. You are calling the multiplication member function.

So it will throw the result. So $result$ is of the form X plus iY , complex number. So that we are displaying. So we have given the sample test $2 + 3i$ and $4 + 5i$. So and then we got the exact output, right.

So this is how we can have a class and then create an object. And you can solve, as I said, several problems. This is one of the mathematical problems, right? Similarly, you can think of, right, the banking system. In fact, you can think of neural networks.

Or we will go, right, how to solve the problems like data structures, right? Create a class, let us say, linked list. Or even the arrays, right? So create a class arrays. And then you can do set of operations like insertion.

In arrays, deletion in arrays, updating the arrays, or you can write the program like searching, you can write the function like searching, member function like searching, sorting, right? So, the main idea is, with the help of object-oriented programming, right? So, you all might have studied procedural-oriented programming, you can think of, right? Creating a class. Defining a class, creating an object, and then you can invoke many functions. So here, we have seen several keywords. So, one of the keywords you are seeing over here, we will talk about private and public slightly later, and here, static. So, in the next class, we will see a few examples about static.

And then, how the static works without a static variable or with a static variable, all the results are being changed. So, that we will see in the next class. Thank you.