# Programming with C and C++

*CSC-101 (Lecture 33 and 34)*

**Dr. R. Balasubramanian**
**Professor**
**Department of Computer Science and Engineering**
**Mehta Family School of Data Science and Artificial Intelligence**
**Indian Institute of Technology Roorkee**
**Roorkee 247 667**

***bala@cs.iitr.ac.in***
***https://faculty.iitr.ac.in/cs/bala/***

# Parameterized Constructor

https://ideone.com/p6kzuK

```
1  #include <iostream>
2  using namespace std;
3  class Employee {
4     public:
5         int id;//data member
6         string name;//data member
7         float salary;
8         Employee(int i, string n, float s)
9          {
10             id = i;
11             name = n;
12             salary = s;
13         }
14        void display()
15         {
16             cout<<id<<"  "<<name<<"  "<<salary<<endl;
17         }
18 };
```

# Parameterized Constructor

```
19  int main(void) {
20      Employee e1 =Employee(18, "Virat", 100000);
21      Employee e2=Employee(45, "Rohit", 90000);
22      e1.display();
23      e2.display();
24      return 0;
25  }
```

stdout

```
18  Virat  100000
45  Rohit  90000
```

# Same Program in a different way

```
19  int main(void) {
20      Employee e1(18, "Virat", 100000);
21      Employee e2(45, "Rohit", 90000);
22      e1.display();
23      e2.display();
24      return 0;
25  }
26
```

stdout

```
18   Virat   100000
45   Rohit   90000
```

https://ideone.com/u1tVq4

# Parameterized Constructor

- Design a class Circle with a parameterized constructor that takes the radius as an argument and calculates the area of the circle in C++.

# Parameterized Constructor

https://ideone.com/cjnC7X

```
1  #include <iostream>
2  #include <cmath>
3
4  class Circle {
5  private:
6      double radius;
7
8  public:
9      // Parameterized constructor to initialize the radius
10     Circle(double r) : radius(r) {}
11
12     // Method to calculate the area of the circle
13     double calculateArea() {
14         return 3.14159265359 * pow(radius, 2);
15     }
16 };
17
```

# Parameterized Constructor

```
18  int main() {
19      double radius;
20      std::cout << "Enter the radius of the circle: ";
21      std::cin >> radius;
22
23      // Create a Circle object with the provided radius
24      Circle myCircle(radius);
25
26      // Calculate and display the area of the circle
27      double area = myCircle.calculateArea();
28      std::cout << "The area of the circle with radius " << radius << " is: "
29      << area << std::endl;
30
31      return 0;
32  }
33
```

stdin

10

Enter the radius of the circle:
The area of the circle with radius 10 is: 314.159

# Define the member function outside the class

https://ideone.com/BRc3XM

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

// Declare a class
class MyClass {
public:
    // Function declaration within the class
    void printMessage();
};

// Define the member function outside the class
void MyClass::printMessage() {
    cout << "Hello from MyClass!" << endl;
}

```

```
16  int main() {
17      MyClass myObj;
18      // Call the member function
19      myObj.printMessage();
20
21      return 0;
22  }
23
```

stdout

```
Hello from MyClass!
```

- Create a complex number class and add two complex numbers by taking one number from user and another one should be passed using constructor in C++.

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

class Complex {
private:
    double real;
    double imaginary;

public:
    // Constructor to initialize a complex number
    Complex(double r, double i) {
        real = r;
        imaginary = i;
    }

```

https://ideone.com/otMfkL

```
16      // Method to add two complex numbers
17      Complex add(Complex c1, Complex c2) {
18          double sumReal = c1.real + c2.real;
19          double sumImaginary = c1.imaginary + c2.imaginary;
20          return Complex(sumReal, sumImaginary);
21      }
22
23      // Method to display the complex number
24      void display() {
25          cout << real << " + " << imaginary << "i" << endl;
26      }
27  };
28
```

```
29  int main() {
30      double userReal, userImaginary;
31
32      cout << "Enter a complex number (real part): ";
33      cin >> userReal;
34      cout << "Enter the imaginary part: ";
35      cin >> userImaginary;
36
37      // Create a complex number using the user's input
38      Complex userComplex(userReal, userImaginary);
39
40      // Create another complex number using the constructor
41      Complex constructorComplex(3.5, 2.0); // Example values
42
```

```
43      // Add the two complex numbers
44      Complex result = result.add(userComplex,constructorComplex);
45
46      cout << "User's Complex Number: ";
47      userComplex.display();
48
49      cout << "Complex Number from Constructor: ";
50      constructorComplex.display();
51
52      cout << "Sum of Complex Numbers: ";
53      result.display();
54
55      return 0;
56  }
```

stdin copy

```
5.0 2.0
```

stdout copy

```
Enter a complex number (real part): Enter the imaginary part: User's Complex Number:
5 + 2i
Complex Number from Constructor: 3.5 + 2i
Sum of Complex Numbers: 8.5 + 4i
```

# Another way

https://ideone.com/9uDswj

```
1   #include <iostream>
2   using namespace std;
3
4   class Complex {
5   private:
6       double real;
7       double imaginary;
8
9   public:
10      // Constructor to initialize a complex number
11      Complex(double r, double i) {
12          real = r;
13          imaginary = i;
14      }
15
```

```
16      // Method to add two complex numbers
17      Complex add(Complex& other) {
18          double sumReal = real + other.real;
19          double sumImaginary = imaginary + other.imaginary;
20          return Complex(sumReal, sumImaginary);
21      }
22
23      // Method to display the complex number
24      void display() {
25          cout << real << " + " << imaginary << "i" << endl;
26      }
27  };
28
```

```
29  int main() {
30      double userReal, userImaginary;
31
32      cout << "Enter a complex number (real part): ";
33      cin >> userReal;
34      cout << "Enter the imaginary part: ";
35      cin >> userImaginary;
36
37      // Create a complex number using the user's input
38      Complex userComplex(userReal, userImaginary);
39
40      // Create another complex number using the constructor
41      Complex constructorComplex(3.5, 2.0); // Example values
42
```

```
43      // Add the two complex numbers
44      Complex result = userComplex.add(constructorComplex);
45
46      cout << "User's Complex Number: ";
47      userComplex.display();
48
49      cout << "Complex Number from Constructor: ";
50      constructorComplex.display();
51
52      cout << "Sum of Complex Numbers: ";
53      result.display();
54
55      return 0;
56  }
57
```

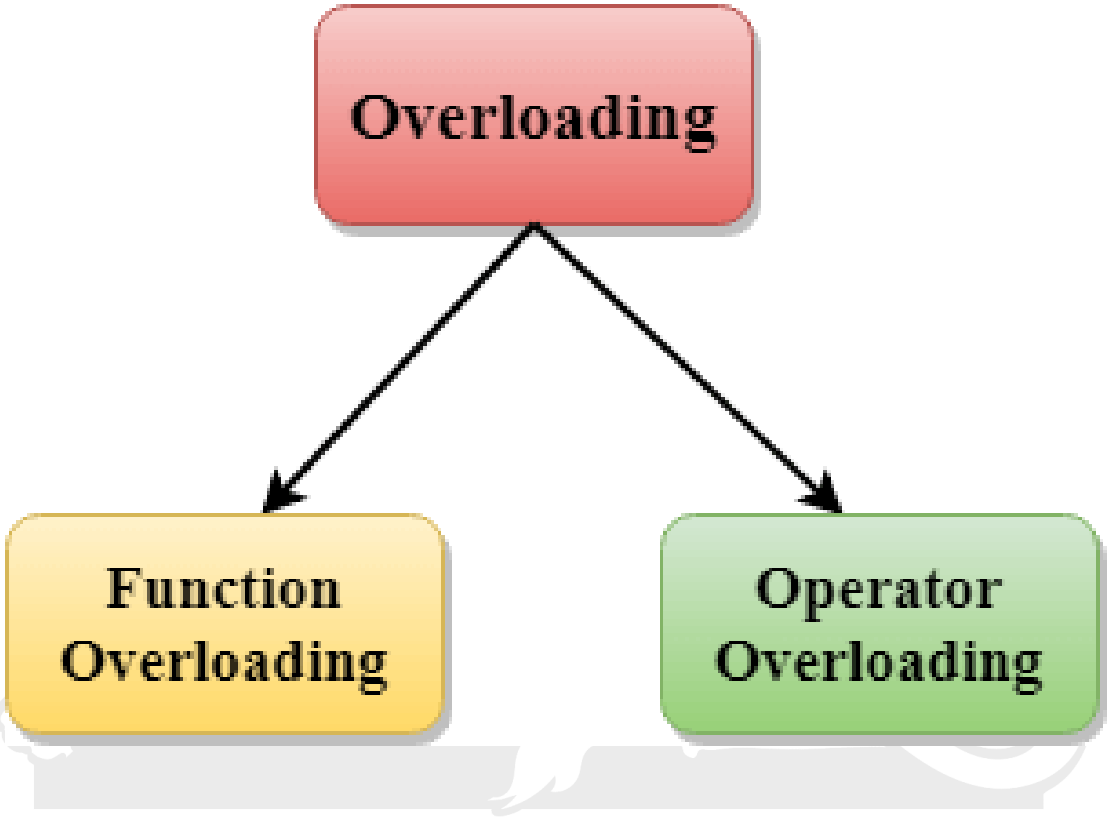
stdin copy

```
5 2
```

stdout copy

```
Enter a complex number (real part): Enter the imaginary part: User's Complex Number:
5 + 2i
Complex Number from Constructor: 3.5 + 2i
Sum of Complex Numbers: 8.5 + 4i
```

# C++ Overloading (Function and Operator)



# Function Overloading

https://ideone.com/dGL3En

```
1  #include <iostream>
2  using namespace std;
3  class Cal {
4      public:
5  static int add(int a,int b){
6          return a + b;
7      }
8  static int add(int a, int b, int c)
9      {
10         return a + b + c;
11     }
12 };
13
```

```
14  int main(void) {
15      Cal C;  //  class object declaration.
16      cout<<C.add(10, 20)<<endl;
17      cout<<C.add(12, 20, 23);
18     return 0;
19  }
20
```

stdout

```
30
55
```

# Operator Overloading

```
1  #include <iostream>
2  using namespace std;
3  class Test
4  {
5     private:
6        int num;
7     public:
8        Test()
9        {num=8;}
10       void operator ++(){
11          num = num+2;
12       }
13       void Print() {
14          cout<<"The Count is: "<<num;
15       }
16 };
17
```

https://ideone.com/Db2VPH

```
18  int main()
19  {
20      Test tt;
21      ++tt;  // calling of a function "void operator ++()"
22      tt.Print();
23      return 0;
24  }
25
```

stdout

```
The Count is: 10
```

# Constructor Overloading

https://ideone.com/4Bxwz9

```
1  // C++ program to demonstrate constructor overloading
2  #include <iostream>
3  using namespace std;
4
5  class Person { // create person class
6      private:
7          int age; // data member
8      public:
9      // 1. Constructor with no arguments
10     Person()
11     {
12         age = 17; // when object is created the age will be 17
13     }
14     // 2. Constructor with an argument
15     Person(int a)
16     { // when parameterised Constructor is called with a value the
17         // age passed will be initialised
18         age = a;
19     }
20
```

```
21      intgetAge()
22      { // getter to return the age
23      return age;
24      }
25  };
26
27  int main()
28  {
29      // called the object of person class in differnt way
30      Person person1, person2(49);
31      cout<< "Person1 Age = " << person1.getAge() <<endl;
32      cout<< "Person2 Age = " << person2.getAge() <<endl;
33      return 0;
34  }
35
```

stdout

```
Person1 Age = 17
Person2 Age = 49
```

# Thank You!