**Constructors**

A **constructor** is a special member function in C++ used to initialize objects of a class. It is automatically invoked when an object is created.

**Key Features of Constructors**

1. **Same Name as the Class**: The constructor has the same name as the class.
2. **No Return Type**: Constructors do not have a return type, not even void.
3. **Automatically Invoked**: Called automatically when an object is instantiated.
4. **Can Be Overloaded**: You can define multiple constructors with different parameter lists.

**Types of Constructors**

1. **Default Constructor**
2. **Parameterized Constructor**

**1. Default Constructor**

A constructor with no parameters is called a **default constructor**. It initializes members with default values.

#include <iostream>

using namespace std;

class DefaultExample {

int x;

public:

DefaultExample() { // Default Constructor

x = 0;

cout << "Default Constructor Called: x = " << x << endl;

}

};

int main() {

DefaultExample obj; // Default constructor is automatically invoked

return 0;

}

**2. Parameterized Constructor**

A constructor that takes parameters to initialize the object with specific values.

#include <iostream>

using namespace std;

class ParameterExample {

int x, y;

public:

ParameterExample(int a, int b) { // Parameterized Constructor

x = a;

y = b;

cout << "Parameterized Constructor Called: x = " << x << ", y = " << y << endl;

}

};

int main() {

ParameterExample obj(10, 20); // Parameterized constructor is invoked

return 0;

}

**Constructor Overloading**

You can have multiple constructors in a class with different parameters.

#include <iostream>

using namespace std;

class OverloadExample {

int x;

public:

OverloadExample() { // Default Constructor

x = 0;

}

OverloadExample(int a) { // Parameterized Constructor

x = a;

}

void display() {

cout << "Value of x: " << x << endl;

}

};

int main() {

OverloadExample obj1; // Default constructor

OverloadExample obj2(50); // Parameterized constructor

obj1.display();

obj2.display();

return 0;

}

In C++, constructors are typically defined inside the class, but it’s also possible to define the **constructor outside the class** using the scope resolution operator (::). Here's an example:

#include <iostream>

using namespace std;

class MyClass {

private:

string name;

public:

// Declaration of constructor outside the class

MyClass(string n);

void display() {

cout << "Hello, " << name << "!" << endl;

}

};

// Definition of the constructor outside the class

MyClass::MyClass(string n) {

name = n;

}

int main() {

// Create an object of MyClass and initialize with a name

MyClass obj("Buji");

// Call the display method

obj.display(); // Output: Hello, Buji!

return 0;

}