



Parameterized and Default Constructors In C++ | C++ Tutorials for Beginners #30



[Overview](#) [Q&A](#) [Downloads](#) [Announcements](#)

Parameterized and Default Constructors In C++ | C++ Tutorials for Beginners #30

In this tutorial, we will discuss parameterized and default constructors in C++

Parameterized and Default Constructors in C++

Parameterized constructors are those constructors that take one or more parameters. Default constructors are those constructors that take no parameters. The main things to note here are that constructors are written in the public section of the class and the constructors don't have a return type. An example program to demonstrate the concept of the constructor is shown below.

Parameterized Constructors Example Program 1

```
#include<iostream>
```

```
using namespace std;

class Complex
{
    int a, b;

public:
    Complex(int, int); // Constructor declaration

    void printNumber()
    {
        cout << "Your number is " << a << " + " << b << "i" << endl;
    }
};

Complex ::Complex(int x, int y) // ----> This is a parameterized constructor as it takes 2
{
    a = x;
    b = y;
    // cout<<"Hello world";
}
```

Parameters

Code Snippet 1: Parameterized Constructor Example Program 1

As shown in a code snippet 1,

- 1st “complex” class is defined which consists of private data members “a” and “b”.
- 2nd parameterized constructor of the “complex” class is declared which takes two parameters.
- 3rd function “printNumber” is defined which will print the values of the data members “a” and “b”.
- 4th parameterized constructor is defined which takes two parameters and assigns the values to the data members “a” and “b”. The main things to note here are that whenever a new object will be created this constructor will run.

The main program is shown in code snippet 2.

```
int main(){
    // Implicit call
    Complex a(4, 6);
    a.printNumber();

    // Explicit call
    Complex b = Complex(5, 7);
    b.printNumber();

    return 0;
}
```

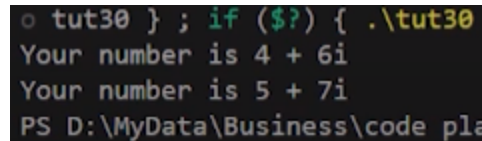
Code Snippet 2: Main Program

As shown in Code Snippet 2,

- 1st parameterized constructor is called implicitly with the object “a” and the values “4” and “6” are passed
- 2nd function “printNumber” is called which will print the values of data members
- 3rd parameterized constructor is called explicitly with the object “b” and the values “5” and “7” are passed

- 4th function “printNumber” is called again which will print the values of data members

The output for the following program is shown in figure 1.



```
o tut30 } ; if ($?) { .\tut30
Your number is 4 + 6i
Your number is 5 + 7i
PS D:\MyData\Business\code pla
```

Figure 1: Program Output 1

Parameterized Constructors Example Program 2

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

class Point{
    int x, y;
public:
    Point(int a, int b){
        x = a;
        y = b;
    }

    void displayPoint(){
        cout<<"The point is ("<<x<<", "<<y<<)"<<endl;
    }

};
```

Code Snippet 3: Parameterized Constructor Example Program 2

As shown in Code Snippet 3,

- 1st “point” class is defined which consists of private data members “x” and “y”.
- 2nd parameterized constructor of the “point” class is defined which takes two parameters and assigns the values to the private data members of the class.
- 3rd function “displayPoint” is defined which will print the values of the data members “x” and “y”.

The main program is shown in code snippet 4.

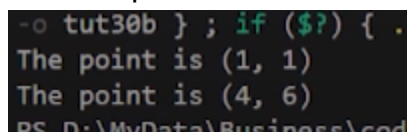
```
int main(){  
    Point p(1, 1);  
    p.displayPoint();  
  
    Point q(4, 6);  
    q.displayPoint();  
    return 0;  
}
```

Code Snippet 4: Main Program

As shown in Code Snippet 4,

- 1st parameterized constructor is called implicitly with the object “p” and the values “1” and “1” are passed
- 2nd function “displayPoint” is called which will print the values of data members
- 3rd parameterized constructor is called implicitly with the object “q” and the values “4” and “6” are passed
- 4th function “displayPoint” is called which will print the values of data members

The output for the following program is shown in figure 2.



```
-o tut30b } ; if ($?) { .  
The point is (1, 1)  
The point is (4, 6)  
PS D:\MyData\Business\cod
```

Figure 2: Program Output 2

[Previous](#)

[Next](#)



CodeWithHarry

Copyright © 2022 CodeWithHarry.com

