Constructor

- It is a member function of class that is used to intialize the object.
- Due to following reasons, constructor is special function of a class:

1. It's name is same as class name.
2. It doesn't have any return type
3. It is designed to call implicitly
4. In life time of the object it gets called only once.

> Note : Constructor gets called once per object.

- In C++, on object, pointer or reference, we can not call constructor explicitly. It is designed to call implicitly.

```
int main( void )
{
    Complex c1; //Ok : Complex::Complex( )
    //c1.Complex( );    //Not OK

    Complex *ptr = &c1; //OK
    //ptr->Complex();   //Not Ok

    Complex &c2 = c1;   //OK
    //c2.Complex( );    //Not OK
    return 0;
}
```

- We can not declare constructor static, constant, volatile or virtual. We can declare constructor "inline" only.
- We can use any access specifier on constructor.
- If constructor is public then we can create object of the class inside member function as well as non member function.
- If constructor is private then we can create object of the class inside member function and friend function only.
- Constructor calling sequence is depends on order of object declaration.
- Compiler do no call constructor on pointer and reference. Compiler call constructor on only object.

```
class Complex
{
private:
    int real;
    int imag;
public:
    Complex( void );
};
//Global definition of constructor
Complex::Complex( void )
```

```
{
    this->real = 0;
    this->imag = 0;
}
```

Types of constructor(3)

```
1. Parameterless constructor
2. Parameterized constructor
3. Default constructor
```

Parameterless constructor

- It is also called as zero argument construtor / User Defined Default constructor.
- A constructor which do not take any parameter is called Parameterless constructor.

```
Complex( )
{
    this->real = 0;
    this->imag = 0;
}
```

- If we create object without passing argument (i.e zero argument) then parameterless constructor gets called.

```
Complex c1, c2, c3;
```

- In above statement, compiler will call parameterless constructor on c1, c2 and c3.

Parameterized constructor

- A constructor which take parameter is called paramerized constructor.
- If name of data member and local variable is same then use of this keyword to access data member is mandatory.

```
Complex( int real, int imag )
{
    this->real = real;
    this->imag = imag;
}
```

- If create object by passing argument then parameterized constructor gets called.

```
Complex c1( 10 ), c2, c3(10,20);
```

- In above statement constructor calling sequence is: c1 : single Parameter ctor c2 : parameterless ctor c3 : two Parameter ctor

Default constructor

- If we do not define constructor inside class then compiler provides one constructor for the class by default. It is called default constructor.
- Compiler do not generate default parameterized constructor i.e Default constructor is parameterless.
- If we want to create object by passing argument then we must define constructor inside class.
- For the C++ application developer implementation of default constructor is as follows:

```
Complex( void )
{
    //Empty body
}
```

- Compiler provided default constructor do not initialize data member declared by the programmer. It is used to initialize data members declared by the compiler.
- eg. v-ptr, vb-ptr etc.
- In C++98 and C++03, we can not call constructor from another constructor. It constructor chaining is not allowed.
- In C++11, we can call constructor from another constructor. It is called constructor delegation.