Constructors and destructors

***Constructors*:- A constructor is a member function of a class which initializes objects of a class. In C++, Constructor is automatically called when object (instance of class) create. It is special member function of the class.**

It is basically three types.

1. ***Default constructor.***
2. ***Parameter constructor.***
3. ***Copy constructor.***

1. ***Default constructor:-*** Default constructor is the constructor which doesn't take any argument. It has no parameters.

Example:-

```
#include <iostream>
using namespace std;
class construct
{
public:
   int a, b;
   construct()     // Default Constructor
   {
      cout<<"Enter two number";
      cin>>a>>b;
   }
};
int main()
```

```
{
   // Default constructor called automatically
   // when the object is created
   construct c;
   cout << "A: " << c.a << endl
        << "B: " << c.b;
   return 1;
}
```

Output:-

Enter two number. 4

5

A:4

B:5

Note:- Even if we do not define any constructor explicitly, the compiler will automatically provide a default constructor implicitly.

2. ***Parameter constructor*:-** It is possible to pass arguments to constructors. Typically, these arguments help initialize an object when it is created. To create a parameterized constructor, simply add parameters to it the way you would to any other function. When you define the constructor's body, use the parameters to initialize the object

Example:-

```
#include<iostream>

using namespace std;

class Car

 {
  public:
```

```
  string brand;
  string model;
  int year;
  Car(string x, string y, int z)      //Constructor with parameters
 {
    brand = x;
    model = y;
    year = z;
  }
};
int main() {
 Car carObj1("BMW", "X5", 1999);
 Car carObj2("Ford", "Mustang", 1969);
 cout << carObj1.brand << " " << carObj1.model << " " <<
carObj1.year << "\n";
 cout << carObj2.brand << " " << carObj2.model << " " <<
carObj2.year << "\n";
 return 0;
}
```

Output:-

```
BMW X5 1999
Ford Mustang 1969
```

3. ***Copy constructor:-*** **A copy constructor is a member function which initializes an object using another object of the same class. A copy constructor has the following general function prototype.**

```
ClassName (const ClassName &old_obj);
```

Example:-

```
#include<iostream>
using namespace std;
class Point
{
private:
   int x, y;
public:
   Point(int p, int q)
{
x = p;
y = q;
}
   // Copy constructor
   Point(const Point &p2)
 {
x = p2.x;
 y = p2.y;
}
   int getX()
 {
 return x;
}
```

```
    int getY()
 {
 return y;
}
};

int main()
{
    Point p1(10, 15); // Normal constructor is called here
    Point p2 = p1; // Copy constructor is called here
    cout << "p1.x = " << p1.getX() << ", p1.y = " << p1.getY();
    cout << "\np2.x = " << p2.getX() << ", p2.y = " << p2.getY();
    return 0;
}
```

Output :-

```
p1.x = 10, p1.y = 15
p2.x = 10, p2.y = 15
```

Destructors:- Destructor is a member function which destructs or deletes an object. It is denoted by (~).

Syntax:-

```
class class_name
```

```
{
    public:
      class_name();    //constructor.
     ~class_name();    //destructor.
}
```

Example:-

```
#include <iostream>
using namespace std;
class ABC
{
   public:
      ABC ()                    //constructor defined
      {
           cout << "Hey look I am in constructor" << endl;
      }
      ~ABC()                    //destructor defined
      {
          cout << "Hey look I am in destructor." << endl;
      }
};

int main()
{
    ABC cc1;                          //constructor is called
    cout << "function main is terminating...." << endl;
    return 0;
}
```

Output:-

Hey look I am in constructor

function main is terminating....

Hey look I am in destructor.

Created by Ajay Kumar Verma