***OPERATOR OVERLOADING IN C++***

C++ allows you to specify more than one definition for a **function** name or an **operator** in the same scope, which is called **function overloading** and **operator overloading** respectively.

An overloaded declaration is a declaration that is declared with the same name as a previously declared declaration in the same scope, except that both declarations have different arguments and obviously different definition (implementation).

When you call an overloaded **function** or **operator**, the compiler determines the most appropriate definition to use, by comparing the argument types you have used to call the function or operator with the parameter types specified in the definitions. The process of selecting the most appropriate overloaded function or operator is called **overload resolution**.

Function Overloading in C++

You can have multiple definitions for the same function name in the same scope. The definition of the function must differ from each other by the types and/or the number of arguments in the argument list. You cannot overload function declarations that differ only by return type.

Following is the example where same function **print()** is being used to print different data types

CODE;

#include <iostream>

**using** **namespace** std**;**

class complex

**{**

int real **,** imag**;**

public**:**

complex**()**

**{**

real**=**0**;**imag**=**0**;**

**}**

complex **(**int r**,** int i**)**

**{**

real**=**r**;**

imag**=**i**;**

**}**

void print **()**

**{**

cout**<<**real **<<**"\t+\t"**<<**imag**<<**" i"**<<**endl**;**

**}**

//operator overloading function

complex **operator** **+(**complex c**)**

**{**

complex temp**;**

temp**.**real**=**real**+**c**.**real**;**

temp**.**imag**=**imag**+**c**.**imag**;**

**return** temp**;**

**}**

**};**

int main**()**

**{**

complex c1 **(**5**,**9**);**

complex c2 **(**3**,**6**);**

complex c3**(**c2**);**

complex c4**;**

c4**=**c1**+**c2**+**c3**;**

c4**.**print**();**

int x**=**5**;**

int y**=**4**;**

int z**=**x**+**y**;**

**return** 0**;**

**}**