C++ function overloading programs: What is function overloading?

Function overloading or **method overloading** is a feature found in various programming languages that allows creating several methods with the same name which differ from each other in the type of the input and the output of the function.
It is simply defined as the ability of one function to perform different tasks.

Function overloading means two or more functions can have the same name but either the number of arguments or the data type of arguments has to be different, also it is important to remember that return value has no role because function will return a value when it is called and at compile time we will not be able to determine which function to call.
In the first example, we make two functions one for adding two integers and other for adding two floats but they have same name and in the second program we make two functions with identical names but pass them different number of arguments.
Function overloading is also known as **compile time polymorphism**.

Function Overriding: If we inherit a class into the derived class and provide a definition for one of the base class's function again inside the derived class, then that function is said to be overridden, and this mechanism is called Function Overriding

Function overloading vs. function overriding:

Overloading a method (or function) in C++ is the ability for functions of the same name to be defined as long as these methods have different signatures (different set of parameters). Method overriding is the ability of the inherited class rewriting the virtual method of the base class.

a) In overloading, there is a relationship between methods available in the same class whereas in overriding, there is relationship between a superclass method and subclass method.

(b) Overloading does not block inheritance from the superclass whereas overriding blocks inheritance from the superclass.

(c) In overloading, separate methods share the same name whereas in overriding, subclass method replaces the superclass.

(d) Overloading must have different method signatures whereas overriding must have same signature.

More Points to remember in overloading & overriding:

1. Function Overloading is when multiple functions with same name exist in a class. Function Overriding is when function have same prototype in base class as well as derived class.

2. Function Overloading can occur without inheritance. Function Overriding occurs when one class is inherited from another class.

3. Overloaded functions must differ in either number of parameters or type of parameters should be different. In Overridden function parameters must be same.

Example 1: Function Overloading:

```
#include <iostream.h>

/* Function arguments are of different data type */

long add(long, long);
float add(float, float);

void main()
{
  long a, b, x;
  float c, d, y;

  cout << "Enter two integers\n";
  cin >> a >> b;
   x = add(a, b);
   cout << "Sum of integers: " << x ;
   cout << "Enter two Fractional numbers\n";
  cin >> c >> d;
   y = add(c, d);
   cout << "Sum of Floats: " << y ;
}

long add(long x, long y)
{
  long sum;
   sum = x + y;
   return sum;
}
 float add(float x, float y)
{
  float sum;
   sum = x + y;
   return sum;
}
```

Example 2: Function Overloading:

```
#include <iostream.h>

 /* Number of arguments are different */

void display(char []);  // print the string passed as argument
void display(char [], char []);

void main()
{
  char first[] = "C programming";
```

```
  char second[] = "C++ programming";

  display(first);
  display(first, second);

}

void display(char s[])
{
  cout << s;
}

void display (char s[], char t[])
{
  cout << s << "\n" << t ;
}
```

Output of program:

C programming

C programming

C++ programming

Function Overriding Example:

```
class Base
{
 public:
 void show()
 {
  cout << "Base class";
 }
};
class Derived: public Base
{
 public:
 void show()
 {
  cout << "Derived Class";
 }
}
```

In this example, function **show()** is overridden in the derived class