Function Overloading in C++

Difficulty Level : Easy • Last Updated : 16 Mar, 2023

Function overloading is a feature of object-oriented programming where two or more functions can have the same name but different parameters. When a function name is overloaded with different jobs it is called Function Overloading. In Function Overloading "Function" name should be the same and the arguments should be different. Function overloading can be considered as an example of a polymorphism feature in C++.

If multiple functions having same name but parameters of the functions should be different is known as Function Overloading.
If we have to perform only one operation and having same name of the functions increases the readability of the program.
Suppose you have to perform addition of the given numbers but there can be any number of arguments, if you write the function such as a(int,int) for two parameters, and b(int,int,int) for three parameters then it may be difficult for you to understand the behavior of the function because its name differs.

The parameters should follow any one or more than one of the following conditions for Function overloading:

- Parameters should have a different type

add(int a, int b)
add(double a, double b)

Below is the implementation of the above discussion:

C++

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


void add(int a, int b)
{
  cout << "sum = " << (a + b);
}

void add(double a, double b)
{
    cout << endl << "sum = " << (a + b);
}

// Driver code
int main()
{
    add(10, 2);
    add(5.3, 6.2);

    return 0;
}
```

Output

```
sum = 12
sum = 11.5
```

- Parameters should have a different number

add(int a, int b)

add(int a, int b, int c)

Below is the implementation of the above discussion:

C++

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

void add(int a, int b)
{
  cout << "sum = " << (a + b);
}

void add(int a, int b, int c)
{
    cout << endl << "sum = " << (a + b + c);
}

// Driver code
int main()
{
    add(10, 2);
    add(5, 6, 4);

    return 0;
}
```

Output

```
sum = 12
sum = 15
```

- Parameters should have a different sequence of parameters.

add(int a, double b)

add(double a, int b)

Below is the implementation of the above discussion:

C++

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

```
void add(int a, double b)
{
    cout<<"sum = "<<(a+b);
}

void  add(double a, int b)
{
    cout<<endl<<"sum = "<<(a+b);
}

// Driver code
int main()
{
    add(10,2.5);
    add(5.5,6);

      return 0;
}
```

Output

```
sum = 12.5
sum = 11.5
```

Following is a simple C++ example to demonstrate function overloading.

CPP

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

void print(int i) {
  cout << " Here is int " << i << endl;
}
void print(double  f) {
  cout << " Here is float " << f << endl;
}
void print(char const *c) {
  cout << " Here is char* " << c << endl;
}

int main() {
  print(10);
  print(10.10);
  print("ten");
  return 0;
}
```

Output

```
Here is int 10
Here is float 10.1
Here is char* ten
```

C++

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

void add(int a, int b)
{
    cout<<"sum ="<<(a+b);
}

void  add(int a, int b,int c)
{
    cout<<endl<<"sum ="<<(a+b+c);
}


main()
{

    add(10,2);
    add(5,6,4);
    return 0;
}
```

C++

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

void add(int a, double b)
{
    cout<<"sum ="<<(a+b);
}
void  add(double a, int b)
{
    cout<<endl<<"sum ="<<(a+b);
}

main()
{
    add(10,2.5);
    add(5.5,6);
    return 0;
           }
```

How does Function Overloading work?

- *Exact match*:- (Function name and Parameter)
- *If a not exact match is found:-*

  ->Char, Unsigned char, and short are promoted to an int.

  ->Float is promoted to double

- *If no match is found*:

  ->C++ tries to find a match through the standard conversion.

- *ELSE ERROR* 😕

1. Function overloading and return type
2. Functions that cannot be overloaded in C++
3. Function overloading and const keyword
4. Function Overloading vs Function Overriding in C++

Recent articles on function overloading in C++

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

325

Related Articles

1. error: call of overloaded 'function(x)' is ambiguous | Ambiguity in Function overloading in C++
2. Function Overloading vs Function Overriding in C++
3. Function Overloading and Return Type in C++
4. Function overloading and const keyword
5. Overloading function templates in C++
6. Advantages and Disadvantages of Function Overloading in C++
7. Overloading of function-call operator in C++