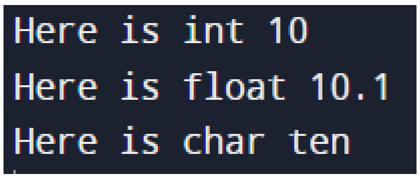
1. Write a C++ Program illustrating function overloading feature.

```
#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;
}
```



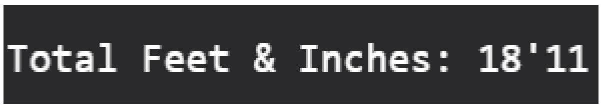
```
Here is int 10
Here is float 10.1
Here is char ten
```

3. Write a C++ Program using the Function Overloading binary + with class objects as argument.

```
#include <iostream>

using namespace std;

class Distance {
public:
        // Member Object
```

```
    int feet, inch;
    // No Parameter Constructor
    Distance()
    {
        this->feet = 0;
        this->inch = 0;
    }

    // Constructor to initialize the object's value
    // Parameterized Constructor
    Distance(int f, int i)
    {
        this->feet = f;
        this->inch = i;
    }

    // Overloading (+) operator to perform addition of
    // two distance object
    Distance operator+(Distance& d2) // Call by reference
    {
        // Create an object to return
        Distance d3;

        // Perform addition of feet and inches
        d3.feet = this->feet + d2.feet;
        d3.inch = this->inch + d2.inch;

        // Return the resulting object
        return d3;
    }
};
```

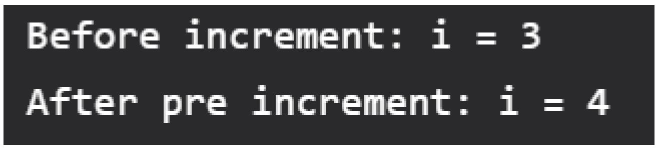
```
// Driver Code
int main()
{
        // Declaring and Initializing first object
        Distance d1(8, 9);

        // Declaring and Initializing second object
        Distance d2(10, 2);

        // Declaring third object
        Distance d3;

        // Use overloaded operator
        d3 = d1 + d2;

        // Display the result
        cout << "\nTotal Feet & Inches: " << d3.feet << "'" << d3.inch;
        return 0;
}
```



```
Total Feet & Inches: 18'11
```

5. Write a C++ Program of function overloading pre ++ and post ++ operator in the same program.

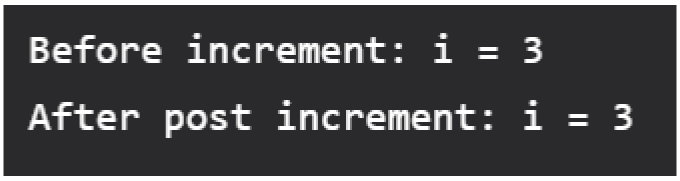
```
#include <bits/stdc++.h>
using namespace std;

class Integer {
private:
        int i;

public:
	// Parameterised constructor
	Integer(int i = 0)
	{
		this->i = i;
	}

	// Overloading the prefix operator
	Integer operator++()
	{
		Integer temp;
		temp.i = ++i;
		return temp;
	}

	// Function to display the value of i
	void display()
	{
		cout << "i = " << i << endl;
	}
};

// Driver function
int main()
{
	Integer i1(3);

	cout << "Before increment: ";
	i1.display();

	// Using the pre-increment operator
	Integer i2 = ++i1;
	cout << "After pre increment: ";
	i2.display();
}
```



```
Before increment: i = 3
After pre increment: i = 4
```

```
// postfix increment operator overloading


#include <bits/stdc++.h>
using namespace std;


class Integer {
private:
	int i;


public:
	// Parameterised constructor
	Integer(int i = 0)
	{
		this->i = i;
	}


	// Overloading the postfix operator
	Integer operator++(int)
	{
		Integer temp;
		temp.i = i++;
		return temp;
	}
```

```
        // Function to display the value of i
        void display()
        {
                cout << "i = " << i << endl;
        }
};

// Driver function
int main()
{
        Integer i1(3);

        cout << "Before increment: ";
        i1.display();

        // Using the post-increment operator
        Integer i2 = i1++;

        cout << "After post increment: ";
        i2.display();
}
```



```
Before increment: i = 3
After post increment: i = 3
```