Introduction to Computing

Lecture 12

Dr. Naveed Anwar Bhatti

Webpage: naveedanwarbhatti.github.io

The Scope of an Identifier

Scope

Definition:

- The scope of an identifier deals with **when the identifier is active** and **which parts of the program can see it**.

- The scope of an identifier begins with its most recent definition and continues until the end of the associated block.

Scope

Local Scope:

- The scope of identifiers declared inside a block.
- They exist from the time of the declaration until the end of the block

Example:

```
for (...)
{
    int x;
    …
    …
}
```

```
int function_1 (...)
{
    int x;
    …
    …
}
```

```
int main (...)
{
    int x;
    …
    …
}
```

Scope

Global Scope:

- The scope of an identifier that is declared outside of any function.
- They exist from the time of declaration until the end of the program and seen by any function

Example:

```
int x;

int main (...)
{
    …
    …
}
```

Scope

Name Precedence:

- If a function declares a local identifier that is the same name as a global identifier, the local identifier takes precedence inside the function

Example:

```
int x =2;

int function_1 (...)
{
    int x=0;
    …
    …
    cout << x;
}
```

Function Overloading

Before Function Overloading

Exercise:

- Write a C++ Program to Check Prime Number By Creating a Function

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cout << "Enter a positive  integer: ";
    cin >> n;

    if ((checkPrimeNumber(n)) == 1)
        cout << n << " is a prime number.";
    else
        cout << n << " is not a prime number.";

    return 0;
}
```

```
bool checkPrimeNumber(int n)
{
    bool flag = true;
    for (int i = 2; i <= n / 2; ++i)
    {
        if (n % i == 0)
        {
            flag = false;
            break;
        }
    }

    return flag;
}
```

Before Function Overloading

Exercise:

- Write a C++ Program to Display Prime Numbers Between Two Intervals Using Functions

```
int main()
{
    int n1, n2;

    cout << "Enter first positive integer: ";
    cin >> n1;

    cout << "Enter second positive integer: ";
    cin >> n2;

    cout << "Prime numbers between " << n1 << " and "
    << n2 << " are: ";

    PrimeNumbers(n1, n2);
}
```

```
void PrimeNumbers(int n1, int n2)
{
    for (int i = n1; i < n2; i++)
    {

        if (checkPrimeNumber(i)==1)
            cout << i << endl;

    }
}
```

Function Overloading

Definition:

- C++ allows you to specify more than one definition for a function name
- The definition of the function must differ from each other by **the types** and/or the **number of arguments** in the argument list.

Example:

```
int myFunction(int x)

int myFunction(int x, int y)

double myFunction(double x, double y)
```

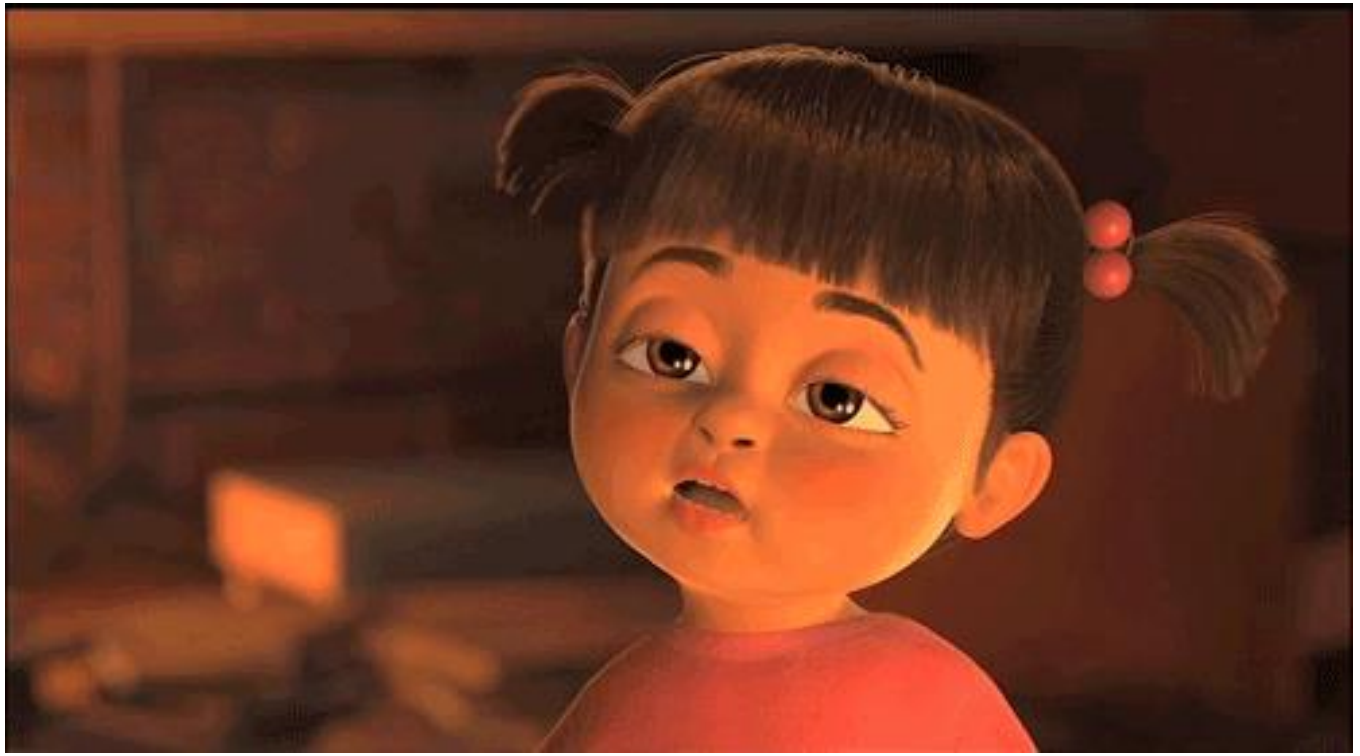
Function Overloading

Example:

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

int add(int x, int y)
{
    return x + y;
}

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

```
int main()
{
    int myNum1 = 2;
    int myNum2 = 4;
    float myNum3 = 2;
    float myNum4 = 2;

    cout << add(myNum1, myNum2) << endl;
    cout << add(myNum3, myNum4) << endl;


    return 0;
}
```

Thanks a lot



If you are taking a Nap, **wake up**.....Lecture Over