Local VS Global Variables

Review: Working Example

Write a C++ program that inputs two numbers from the user and prints the sum of those two numbers by calling the sum function.

Review

Function Prototype

Function Call

Function Definition

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

int addition(int num1, int num2);

main(){
    float number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition(number1, number2);
    cout << "Sum is: " << result;
}

int addition(int num1, int num2)
{
    int sum = num1 + num2;
    return sum;
}
```

Review

Function Prototype

Function Call

Value returning Function

2 Parameters

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

int addition(int num1, int num2);

main(){
    float number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition(number1, number2);
    cout << "Sum is: " << result;
}

int addition(int num1, int num2)
{
    int sum = num1 + num2;
    return sum;
}
```

Review

In the main function, there are different variables and in the addition function there are different variables.

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

int addition(int num1, int num2);

main(){
    int number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition(number1, number2);
    cout << "Sum is: " << result;
}

int addition(int num1, int num2)
{
    int sum = num1 + num2;
    return sum;
}
```

Function with no parameters

Instead of passing parameters, can we use same parameters? i.e.,

number1, number2 and result

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

int addition(int num1, int num2);

main(){
    int number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition(number1, number2);
    cout << "Sum is: " << result;
}

int addition(int num1, int num2)
{
    int sum = num1 + num2;
    return sum;
}
```

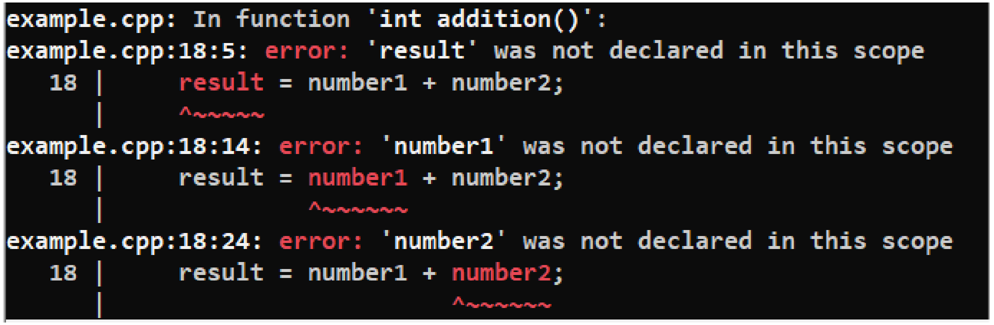
Function with no parameters

Instead of passing parameters, can we use same parameters?
i.e.,

number1, number2 and result

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

int addition();

main(){
    int number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition();
    cout << "Sum is: " << result;
}

int addition()
{
    result = number1 + number2;
    return result;
}
```

Error



```
example.cpp: In function 'int addition()':
example.cpp:18:5: error: 'result' was not declared in this scope
   18 |     result = number1 + number2;
      |     ^~~~~~
example.cpp:18:14: error: 'number1' was not declared in this scope
   18 |     result = number1 + number2;
      |              ^~~~~~~
example.cpp:18:24: error: 'number2' was not declared in this scope
   18 |     result = number1 + number2;
      |                        ^~~~~~~
```

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

int addition();

main(){
    int number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition();
    cout << "Sum is: " << result;
}

int addition()
{
    result = number1 + number2;
    return result;
}
```

Local Variables

Variables within a block { } remain accessible only within that block and not outside that block. These are called local variables of block.

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

int addition();

main(){
    int number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition();
    cout << "Sum is: " << result;
}

int addition()
{
    result = number1 + number2;
    return result;
}
```

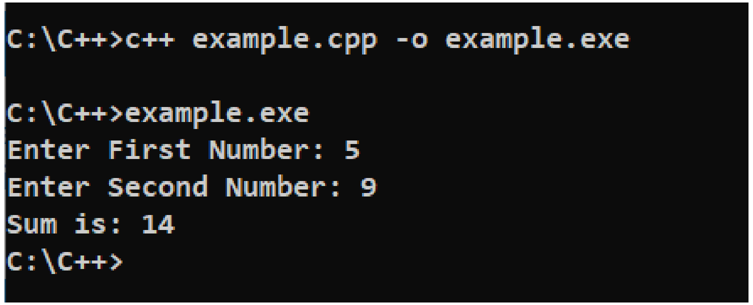
Solution: Global Variables

We can Declare Global Variables before the main function.

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

int addition();
int number1, number2, result;
main(){
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition();
    cout << "Sum is: " << result;
}

int addition()
{
    result = number1 + number2;
    return result;
}
```

Solution: Global Variables

We can Declare Global Variables before the main function.



```
C:\C++>c++ example.cpp -o example.exe

C:\C++>example.exe
Enter First Number: 5
Enter Second Number: 9
Sum is: 14
C:\C++>
```

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

int addition();
int number1, number2, result;
main(){
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition();
    cout << "Sum is: " << result;
}

int addition()
{
    result = number1 + number2;
    return result;
}
```

Local Variables

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

int addition(int num1, int num2);

main(){
    int number1, number2, result;
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition(number1, number2);
    cout << "Sum is: " << result;
}

int addition(int num1, int num2)
{
    int sum = num1 + num2;
    return sum;
}
```

✓ **Low Coupled**

Which one is better?

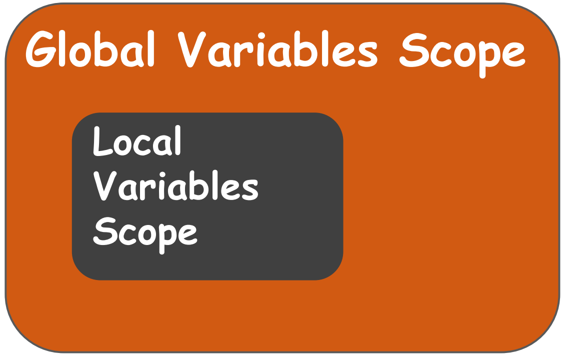
Global Variables

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

int addition();
int number1, number2, result;
main(){
    cout << "Enter First Number: ";
    cin >> number1;
    cout << "Enter Second Number: ";
    cin >> number2;
    result = addition();
    cout << "Sum is: " << result;
}

int addition()
{
    result = number1 + number2;
    return result;
}
```

High Coupled

Local Vs Global Variables

- Low Coupling is Good and Always Desired.
- In Some Cases, where multiple function need to share the same data we have to declare GLOBAL variables.



Learning Outcome

In this lecture, we learnt the difference between Local and Global Variables

Self Assessment

1. What will be the output of the program?

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

/* global variable declaration */
int g = 20;
main()
{
    /* local variable declaration */
    int g = 10;
    cout << "Value of g = " << g;
}
```

Self Assessment

2. What will be the **sequence of the output** of the program? How many **global variables**, **local variables of main**, **parameters of sum function** and **local variables of sum function** are there?

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

int a = 20;
int sum(int a, int b);
main ()
{
  int a = 10;
  int b = 20;
  int c = 0;
  cout << "value of a in main() = " << a << endl;
  c = sum( a, b);
  cout << "value of c in main() = " << c << endl;
}
/* function to add two integers */
int sum(int a, int b)
{
   cout << "value of a in sum() = " << a << endl;
   cout << "value of b in sum() = " << b << endl;
   return a + b;
}
```

Self Assessment

Blood types are named according to three factors:

1. presence of antigen A
2. presence of antigen B, and
3. presence of Rh factor.

If antigen A is found, the blood type includes the letter "A". If antigen B is found, the blood type includes the letter "B". If both antigens are found, the blood type includes the letter “AB”. If neither antigen A nor antigen B are found, the blood type includes the letter "O".
if the Rh factor is present, the blood type ends with "+"; otherwise, it ends with "-".

Self Assessment

First write antigenChecker function than take two Boolean input that represent either of each antigenA or antigenB is present. This function shall return the antigen type. According to following criteria.

If antigen A is found, the blood type includes the letter "A". If antigen B is found, the blood type includes the letter "B". If both antigens are found, the blood type includes the letter “AB”. If neither antigen A nor antigen B are found, the blood type includes the letter "O".

Self Assessment

First write antigenChecker function than take two Boolean input that represent either of each antigenA or antigenB is present. This function shall return the antigen type. According to following criteria.

What will be the header of function ?

Self Assessment

First write antigenChecker function than take two Boolean input that represent either of each antigenA or antigenB is present. This function shall return the antigen type. According to following criteria.

What will be the header of function ?

```
string antigenChecker(bool antigenA, bool antigenB)
```

Self Assessment

First write antigenChecker function than take two Boolean input that represent either of each antigenA or antigenB is present. This function shall return the antigen type. According to following criteria.

```
string antigenChecker(bool antigenA, bool antigenB)
{
        //What will be the BODY of function ?

}
```

Solution: antigenChecker

Let's write the

string antigenChecker(bool antigenA, bool antigenB)

function first.

```
string antigenChecker(bool antigenA, bool antigenB)
{
    if(antigenA == true && antigenB == false){
        return "A";
    }
    else if(antigenA == false && antigenB == true){
        return "B";
    }
    if(antigenA == true && antigenB == true){
        return "AB";
    }
    else{
        return "O";
    }
}
```

Self Assessment

Now your task is to write a function that decided the rh factor of the blood based on following criteria.

if the Rh factor is present, the blood type ends with "+"; otherwise, it ends with "-".

What will be the function header ?

Self Assessment

Now your task is to write a function that decided the rh factor of the blood based on following criteria.

if the Rh factor is present, the blood type ends with "+"; otherwise, it ends with "-".

What will be the function header ?

```
char rhChecker(bool rH)
```

Self Assessment

Now your task is to write a function that decided the rh factor of the blood based on following criteria.

if the Rh factor is present, the blood type ends with "+"; otherwise, it ends with "-".

```
char rhChecker(bool rH)
{
  // What will be the body of function
}
```

Solution: rhChecker

```
char rhChecker(bool rH)
{
    if(rH == true)
    {
        return '+';
    }
    else
    {
        return '-';
    }
}
```

Self Assessment

Now write a bloodType function that take three argument as follow and print the blood group on the screen

```
void bloodType(bool antigenA, bool antigenB, bool rH)
```

This function should use following two functions that we alrready defined.

1. string antigenChecker(bool antigenA, bool antigenB)
2. char rhChecker(bool rH)

Self Assessment

Note:
Take 1 as input from the user if antigen A is present and 0 otherwise. Same is the case for antigen B and rH factor.

Test Cases:

Input	Output
bloodType(1, 1, 0)	AB-
bloodType(1, 0, 1)	A+
bloodType(0, 0, 0)	O-

Solution: bloodType

```
void bloodType(bool antigenA, bool antigenB, bool rH)
{
    string name;
    char sign;
    name = antigenChecker(antigenA, antigenB);
    sign = rhChecker(rH);
    cout << "Blood Type is: " << name << sign;
}
```

Solution: main

Now, Let's write the main function.

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

void bloodType(bool antigenA, bool antigenB, bool rH);
string antigenChecker(bool antigenA, bool antigenB);
char rhChecker(bool rH);

main()
{
    bool a, b, r;
    cout << "is Antigen A present?: ";
    cin >> a;
    cout << "is Antigen B present?: ";
    cin >> b;
    cout << "is rH factor present?: ";
    cin >> r;
    bloodType(a, b, r);
}
```

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

void bloodType(bool antigenA, bool antigenB, bool rH);
string antigenChecker(bool antigenA, bool antigenB);
char rhChecker(bool rH);

main()
{
    bool a, b, r;
    cout << "is Antigen A present?: ";
    cin >> a;
    cout << "is Antigen B present?: ";
    cin >> b;
    cout << "is rH factor present?: ";
    cin >> r;
    bloodType(a, b, r);
}
```

```cpp
string antigenChecker(bool antigenA, bool antigenB)
{
    if(antigenA == true && antigenB == false){
        return "A";
    }
    else if(antigenA == false && antigenB == true){
        return "B";
    }
    if(antigenA == true && antigenB == true){
        return "AB";
    }
    else{
        return "O";
    }
}
```

```cpp
char rhChecker(bool rH)
{
    if(rH == true){
        return '+';
    }
    else{
        return '-';
    }
}
```

```cpp
void bloodType(bool antigenA, bool antigenB, bool rH){
    string name;
    char sign;
    name = antigenChecker(antigenA, antigenB);
    sign = rhChecker(rH);
    cout << "Blood Type is: " << name << sign;
}
```