WESTERN SYDNEY
UNIVERSITY

Local & Global Variables

ELEC1006: ENGINEERING COMPUTING

Local Variables

WESTERN SYDNEY UNIVERSITY

- Variables defined inside a function are local to that function. They are hidden from the statements in other functions, which normally cannot access them.
- Because the variables defined in a function are hidden, other functions may have separate, distinct variables with the same name.

WESTERN SYDNEY UNIVERSITY

Example 1

```
1  // This program shows that variables defined in a function
2  // are hidden from other functions.
3  #include <iostream>
4  using namespace std;
5
6  void anotherFunction(); // Function prototype
7
8  int main()
9  {
10    int num = 1;   // Local variable
11
12    cout << "In main, num is " << num << endl;
13    anotherFunction();
14    cout << "Back in main, num is " << num << endl;
15    return 0;
16 }
17
18 //*********************************************
19 // Definition of anotherFunction              *
20 // It has a local variable, num, whose initial value  *
21 // is displayed.                              *
22 //*********************************************
23
24 void anotherFunction()
25 {
26    int num = 20;  // Local variable
27
28    cout << "In anotherFunction, num is " << num << endl;
29 }
```

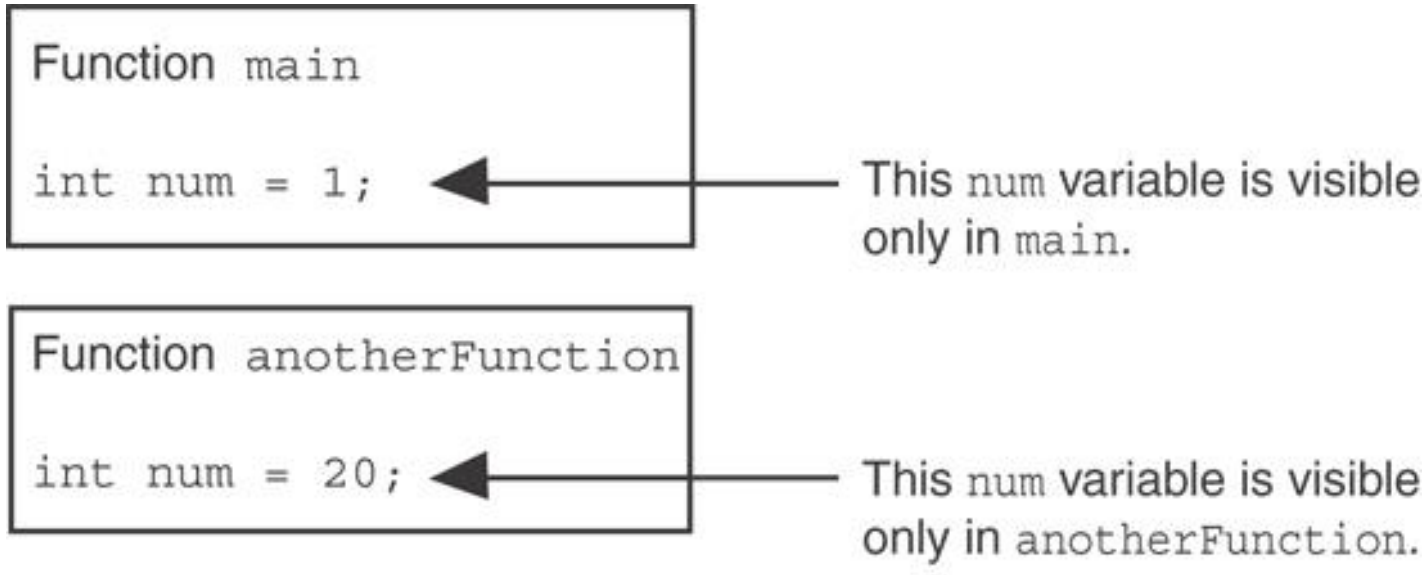
Program Output

```
In main, num is 1
In anotherFunction, num is 20
Back in main, num is 1
```

Example 1 continued …

WESTERN SYDNEY UNIVERSITY

- When the program is executing in `main`, the `num` variable defined in `main` is visible only in `main`.
- The `num` variable in `anotherFunction` is visible only in `anotherFunction` and is separate from the `num` used in `main`.



Lifetime of a Local Variable

WESTERN SYDNEY UNIVERSITY

- A function's local variables exist only while the function is executing. This is known as the *lifetime* of a local variable.
- When the function begins, its local variables and its parameter variables are created in memory, and when the function ends, the local variables and parameter variables are destroyed.
- This means that any value stored in a local variable in a called function is lost when the called function ends and execution returns to the calling function.

Global Variables & Global Constants

WESTERN SYDNEY UNIVERSITY

- A global variable is any variable defined outside all the functions in a program.
- The scope of a global variable is the portion of the program from the global variable definition to the end of the source code.
- This means that a global variable can be accessed by *all* functions that are defined after the global variable is defined.

Warning on using Global Variables

WESTERN SYDNEY UNIVERSITY

- **You should avoid using global variables because they make programs difficult to debug.**

- Any global that you create should be *global constants*.

Example 2

WESTERN SYDNEY UNIVERSITY

Global constants defined for values that do not change throughout the program's execution.

```
1  // This program calculates gross pay.
2  #include <iostream>
3  #include <iomanip>
4  using namespace std;
5
6  // Global constants
7  const double PAY_RATE = 22.55;     // Hourly pay rate
8  const double BASE_HOURS = 40.0;    // Max non-overtime hours
9  const double OT_MULTIPLIER = 1.5; // Overtime multiplier
10
11 // Function prototypes
12 double getBasePay(double);
13 double getOvertimePay(double);
14
15 int main()
16 {
17    double hours,              // Hours worked
18           basePay,            // Base pay
19           overtime = 0.0,     // Overtime pay
20           totalPay;           // Total pay
```

Example 2 continued …

WESTERN SYDNEY UNIVERSITY

The constants are then used for those values throughout the program.

```
29      // Get overtime pay, if any.
30      if (hours > BASE_HOURS)
31         overtime = getOvertimePay(hours);
```

```
56    // Determine base pay.
57    if (hoursWorked > BASE_HOURS)
58       basePay = BASE_HOURS * PAY_RATE;
59    else
60       basePay = hoursWorked * PAY_RATE;
```

```
75      // Determine overtime pay.
76      if (hoursWorked > BASE_HOURS)
77      {
78         overtimePay = (hoursWorked - BASE_HOURS) *
79                  PAY_RATE * OT_MULTIPLIER;
```

Initialising Local & Global Variables

WESTERN SYDNEY UNIVERSITY

- Local variables are not automatically initialized. They must be initialized by programmer.

- Global variables (not constants) are automatically initialized to `0` (numeric) or `NULL` (character) when the variable is defined.

More info

WESTERN SYDNEY UNIVERSITY

[1] cplusplus.com: Variables. Data Types. https://www.cplusplus.com/doc/oldtutorial/variables/

[2] learncpp.com: 6.3 – Local variables https://www.learncpp.com/cpp-tutorial/local-variables/

[3] learncpp.com: 6.4 – Introduction to global variables https://www.learncpp.com/cpp-tutorial/introduction-to-global-variables/