Scope of Variables

Concept of Scope

Instructor: Andy Abreu

Concept of Scope

- Scope – noun
  - extent or range of view
    - www.dictionary.com
- Where we declare a variable in our C++ program, inherently determine where this variable would be accessible.

Declaring variables in a block

- A block of code is defined as code inside { }
- If statements, while loop, for loop, are usually follow by a block of code.
- If the variable is declared in the block, it can't be accessed anywhere else.

Example – in block declaration

```
#include <iostream>
using namespace std;
int main()
{
   for ( int i = 1 ; i <= 10 ; ++i )
   cout << i << endl; //in block (Print 1...10)
   cout << i << endl; //out of block (Compile error)
   return 0;
}
```

Declaring variables in a function

- A variable defined in a function is contained with in the function block of code.
- This means that you can not share variables between functions.
- Each function needs to be “self sufficient” in terms of declaring variables that it needs.

  OR
- The variable must be pass into the function as a parameter. (See next slide)

Passing variables as arguments

- When we pass variable as an argument from main function to a sub function…
- Only the value of the variable is passed through
- That is why we can give the variable a different name
- That is also why whatever changes we do to this new, ‘different’ variable wouldn’t affect main program.

Example – Passing variables by value

```
#include <iostream>
using namespace std;
void update(int);
int main()
{
    int n = 100;
    cout << "Value of n: "<< n << endl;
    update(n);
    cout << "Value of new n: "<< n << endl;
    return 0;
}

void update(int n)
{
      cout << "IN UPDATE: Value of n: "<< n << endl;
      n = 0;
      cout << "IN UPDATE: Value of n again: "<< n << endl;
}
```