Pass by Reference

Instructor: Krishna Mahavadi
Pass by value

• Everything we have done so far is pass by value

• The value within the variable is passed to the subfunction

• The sub-function will take the value and store it in a variable within its own scope

• The variable with the value is discarded when it runs out of scope (in the sub-function)
Example – Passing variables by value

```cpp
#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;
}
```
How can we keep the change?

- One way is to return the newly calculated value to the calling function
- Then assign the value to the same variable, effectively updating it

- \( n = \text{update}(n) \)
- That would do the trick!
- However there is another way, a cleaner way
Passing variables by reference

- Instead of giving a copy of the value to the subfunction, we can give the ‘reference’ of the value.

- When we give the reference of the value, the subfunction would be able to change the value.

- Reference is another name referring to the location where the value is stored.
Example – Passing by reference

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

void update(int &)
{
    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;
```
Another example pass by reference

• If you need a function that swaps the values of two variables you can design the swap function to do this with pass by reference.

    void swap( int &x, int &y );
#include <iostream>
using namespace std;

void swap( int &, int &);

int main()
{
    int a = 10, b = 20;
    cout << "before swap:" << endl << "a: " << a << " b: " << b << endl;
    swap( a, b );
    cout << "after swap:" << endl << "a: " << a << " b: " << b << endl;
    return 0;
}

void swap( int &, int &y )
{
    int t = x;
    x = y;
    y = t;
}
Other Uses

• Passing by reference would also come in handy if you need to read in a set of inputs from the user and you don’t want to write a function to read in one at a time.

• Example: read dimensions of trapezoid

/*
 This calling function has to declare the variables:
 h – height, b1 – base 1 and b2 – base 2
 */
void getTrapezoidDimensions( int &h, int &b1, int &b2 ){
    cout << "Enter the height, base 1 and base 2: ";
    cin >> h >> b1 >> b2;
}