

Chapter 9 – More on Argument Passing

1. Pass by Value

Thus far, all items used in the argument list of functions have been *pass by value* arguments. Pass by value means that the values in the calling arguments are copied into the function parameters, i.e. each argument or parameter object has its own location in memory. Here is an example

Function Example: `int Strange(int a, int b, int c) {`
 `a = b + c;`
 `b = b * b;`
 `c = b;`
 `return a + b + c;`
 `}`

Function Call Example: `w = Strange(x, y, z);`

The variable objects x , y , and z are really three named memory locations. If $x = 3$, $y = 4$, and $z = 5$, this could be represented with the following illustration:

3	4	5
x	y	z

The function arguments a , b , and c also are three named memory locations that, when the values of x , y , and z are copied into a , b , and c during the call of the function `strange`, this will be the result:

3	4	5
x	y	z
3	4	5
a	b	c

Since a , b , and c are completely separate from x , y , and z , the values of a , b , and c can be altered without affecting the values in x , y , and z .

The statements:

```
a = b + c;
b = b * b;
c = b;
```

will result in the following values being stored in the 6 variable objects:

3	4	5
x	y	z
9	16	16
a	b	c

In addition, the statement: `return a + b + c;`

will result in the variable object *w* receiving the value 41.

41	3	4	5
w	x	y	z
9	16	16	
a	b	c	

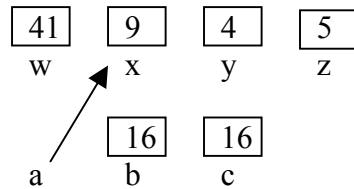
2. Pass by Reference

A pass by value argument always copies a value at a particular memory location to another memory location. This is not the only way that arguments can be passed in function calls. Instead of a value being passed, an argument can pass its address. This is called *pass by reference*. The effect of pass by reference is to have one memory location with multiple names. Any change to the memory location under one name is a change to the memory location under any other name.

In the function `strange`, if the argument `a` in the function header is declared to be a *pass by reference* variable, the calling argument `x` and the pass by reference parameter `a` would be names for the same memory location. Using the example program, this would have the following effects:

At the function call:

??	3	4	5
w	x	y	z
4	5		
a	b	c	

At the end of the function execution:

When the function is complete, $a = 9$, $b = 16$, $c = 16$, $y = 4$, and $z = 5$. This is the same result for these variables in the pass by value example. However, since the variables a and x are names for the same memory location, the changes made to a in the function `Strange` are changes made to x in the function `main`. Therefore, pass by reference is another way that information may be returned from a called function to a calling function.

To make an parameter pass by reference, the ampersand (&) is placed between the type or class and the parameter name in the function header. For example:

```
int Strange(int &a, int b, int c)
```

The call to `strange` is unaffected. For example:

```
w = Strange(x, y, z);
```

Here is an example program using pass by reference:

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

int Strange(int &a, int b, int c) {
    a = b + c;
    b = b * b;
    c = b;
    return a + b + c;
}

void main() {
    int w, x = 3, y = 4, z = 5;
    w = Strange(x, y, z);
    cout << w << ' ' << x << ' ' << y << ' ' << z << endl;
}
```

When run, the output of this program is

```
41 9 4 5
```

3. Passing Strings

Until now, passing C-strings as arguments has been avoided, but this is not because it is hard! There is simply no practical way to pass C-strings by value. All C-strings parameters are passed by reference. However, a C-string parameter being passed by reference is not written the same way as other formal parameters being passed by reference (i.e. the ampersand is not used). Instead, C-strings as formal parameters in function headers are indicated by use of an * before the parameter or a set of empty square brackets ([]) after the parameter. Here are two examples:

```
void GetName(char *name)
void GetId(char id[ ])
```

Here is an example program that uses a string as a function parameter.

```
#include <iostream>
#include <string>           // may need to be written <string.h>
using namespace std;

const int max = 60;

void GetFullName(char *name, int maxChar);

int main( ) {
    char yourName[max];
    GetFullName(yourName, max);
    cout << "\nHow are you today, " << yourName << "?" << endl;
}

void GetFullName(char *name, int maxChar) {
    cout << "Enter your name: ";
    cin.getline(name, maxChar, '\n');
}
```

If the user enters “Howdy Doody” when the program is run, the run will appear as follows:

```
Enter your name: Howdy Doody
How are you today, Howdy Doody?
```

Exercises

1. What is meant by *pass by value*?
2. What is meant by *pass by reference*?
3. How are formal parameters marked for *pass by reference* if they are not C-strings?
4. How are formal parameters marked for *pass by reference* if they are C-strings?
5. What is the output of the following program

```

int Strange(int a, int &b, int c) {
    a = b + c;
    b = b * b;
    c = b;
    return a + b + c;
}

void main() {
    int w, x = 1, y = 2, z = 3;
    w = Strange(x, y, z);
    cout << w << ' ' << x << ' ' << y << ' ' << z << endl;
}

```

Programming Assignment 9.1

Write a program that uses function *main* to call function *Strange* when function *Strange* has only pass by value arguments. Output all variables used in the function *main* before the call and after the call to function *Strange*. Before running the program, predict the output, then check your predictions against the actual results.

Programming Assignment 9.2

Write a program that uses function *main* to call function *Strange* when function *Strange* has only pass by reference arguments. Output all variables used in the function *main* before the call and after the call to function *Strange*. Before running the program, predict the output, then check your predictions against the actual results.

Programming Assignment 9.3

Modify the program of programming assignment #9.2 so that the function used is *void* and the value returned from the function is returned via a pass by reference parameter.

Programming Assignment 9.4

Write a program that requests the users first and last name of the program user, then outputs the following message.

To: <First Name> <Last Name>
From: Woretha Trie

Dear <First Name>,
As I am sure you know, <First Name>, Russian Golden Long Tailed Hamsters are in danger of losing their habitat to urban dwellers all over the world. Please contribute to our efforts to preserve this historic animal. After all, where would we be without their dramatic contribution to European history in the 14th century.

All prompts and input should be inside of the one function and all output should be within a second function. Function main should call both functions.