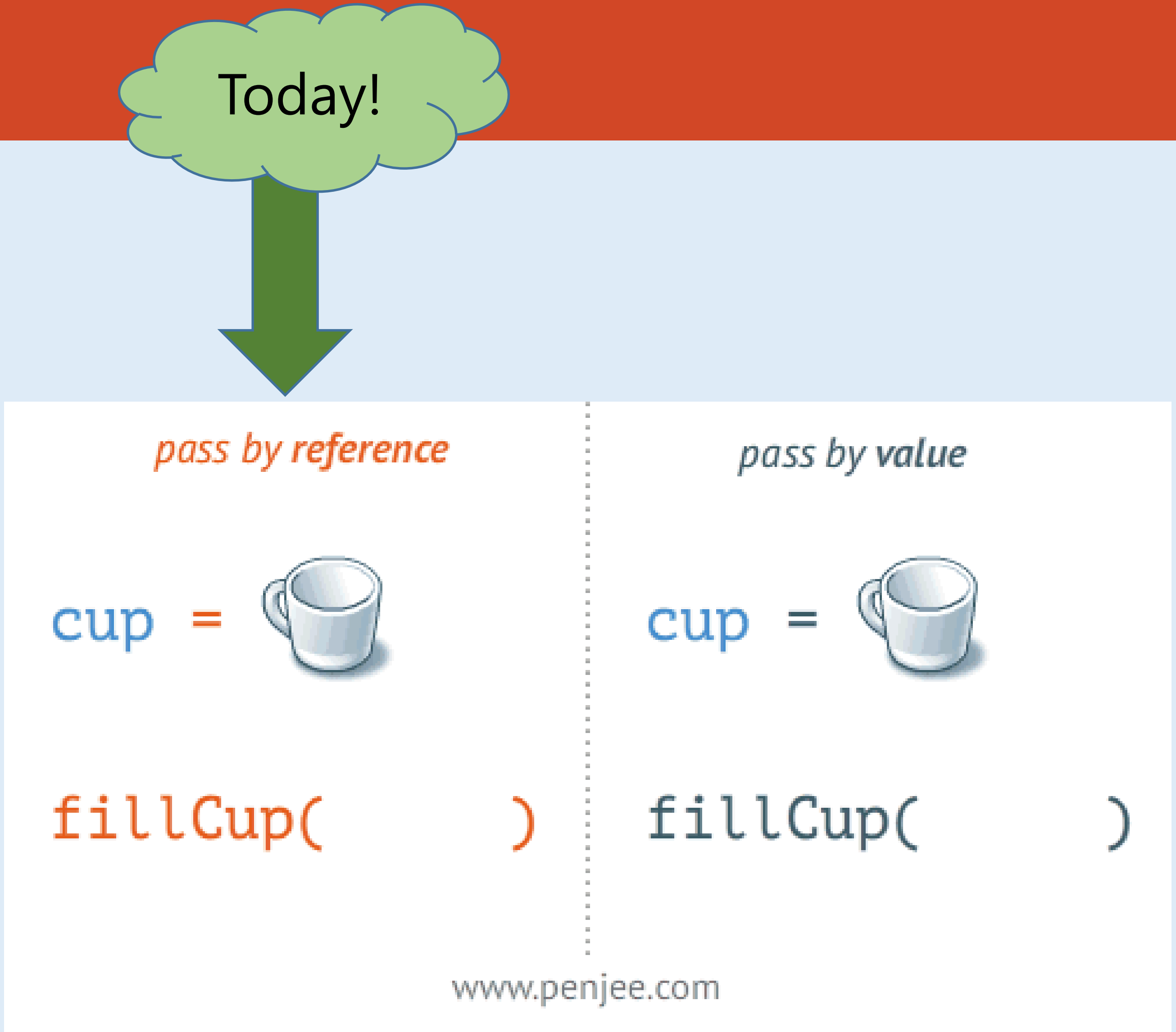
Lab #8

Functions (3)

Structured Programming 2017/2018

Today’s Lab



Today!

By Value VS By Reference

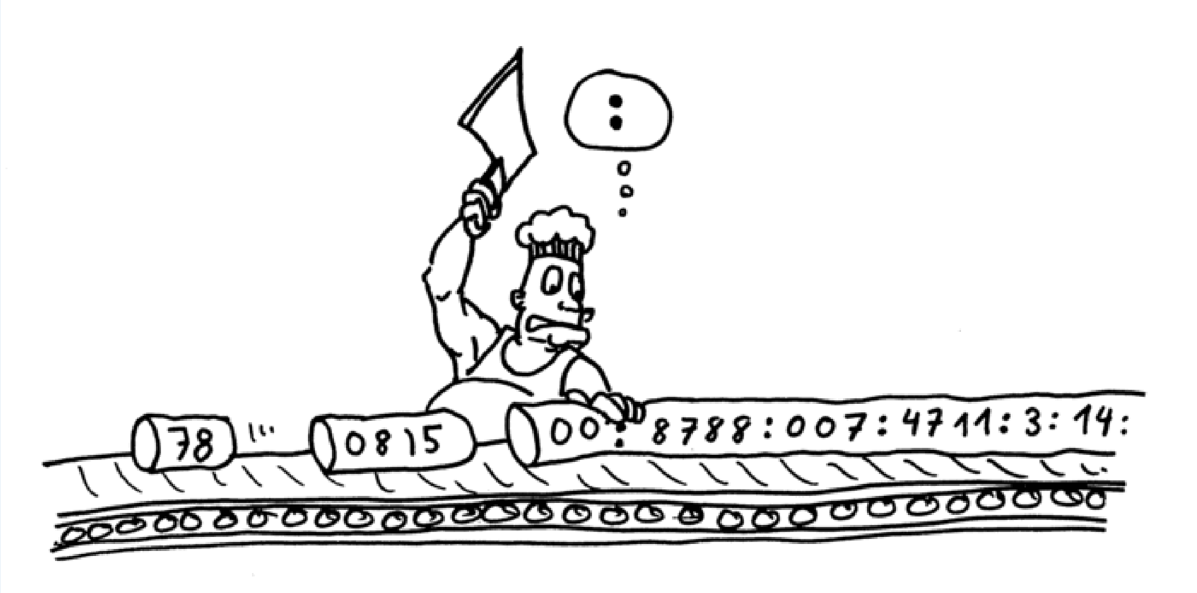
Function Call – By Value

- The local parameters are copies of the original arguments passed in.
- Changes made in the function to these variables do not affect originals.

Function Call – By Reference

- The local parameters are references to the storage locations of the original arguments passed in.
- Changes to these variables in the function will affect the originals.
- No copy is made, so overhead of copying (time, storage) is saved.
- To do so, you "invite" the function to work on the caller's argument by passing the address of the argument not its value.
- Specified by ampersand &, after the type in the formal parameters list (except arrays).
- When passing array variables, the default is by reference.

Exercises!



Swap Function Example

Consider a swapping function that takes 2 numbers and exchange their values.

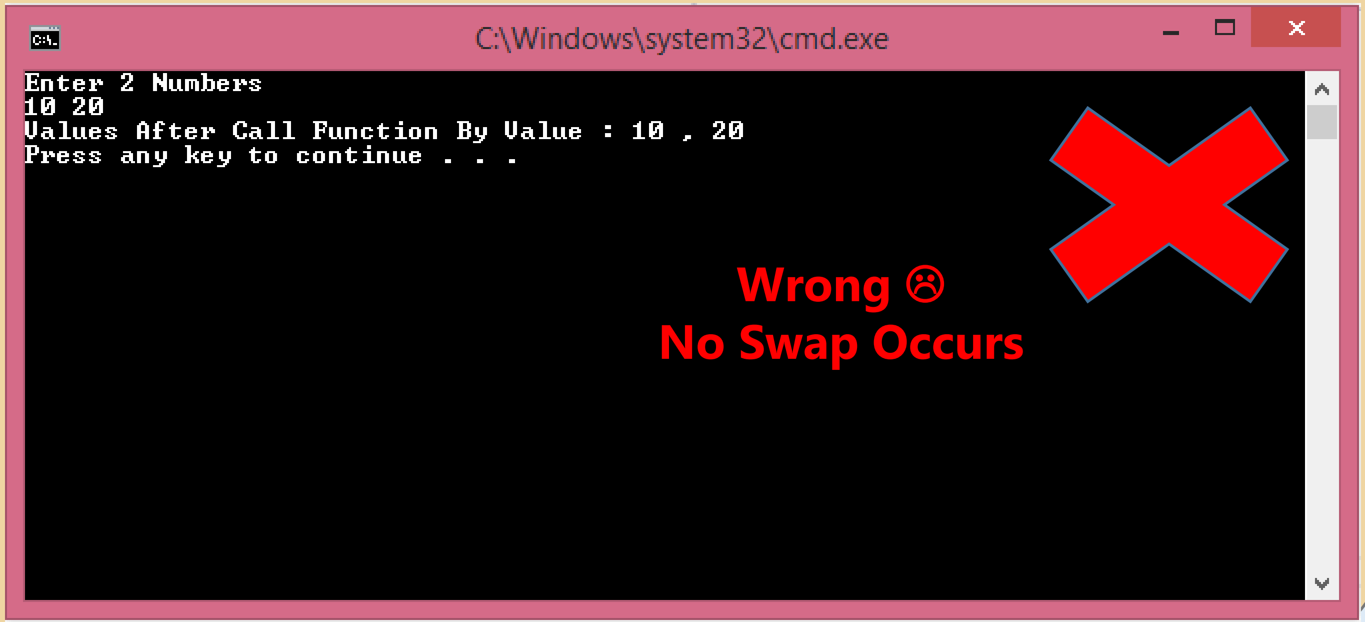
N.B.: The result of swap should be displayed from main function.

<u>Sample of Execution:</u>

```
Enter 2 Numbers: 10  20
Results After Calling Swap Function: 20 10
```

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

void SwapFunction(int Num1 , int Num2);//SwapFunction Declaration (By Value)

int main()
{

int i,j;
cout << "Enter 2 Numbers " <<endl;
cin >> i >> j;

SwapFunction(i , j);  // SwapFunction Call

// Display Result After SwapFunction Call
cout << "Values After Call Function By Value : " << i << "," << j;

}
```

```
void SwapFunction(int Num1 , int Num2);//SwapFunction Definition
{
    int Temp = Num1;
    Num1 = Num2;
    Num2 = Temp;

}
```

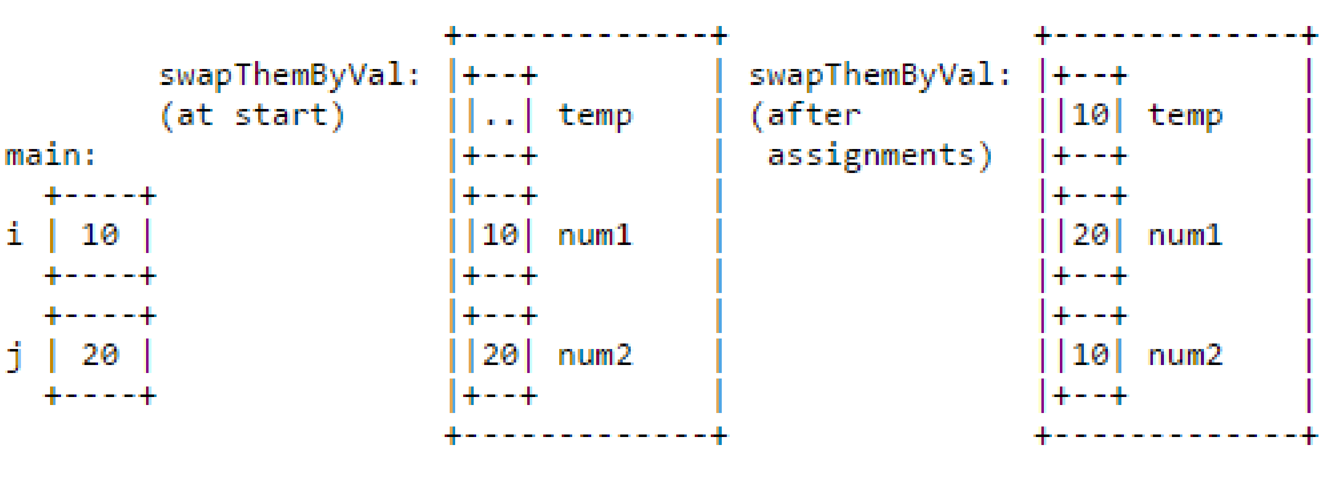


```
Enter 2 Numbers
10 20
Values After Call Function By Value : 10 , 20
Press any key to continue . . .
```

Wrong ☹
No Swap Occurs

Why ☹ ?!!

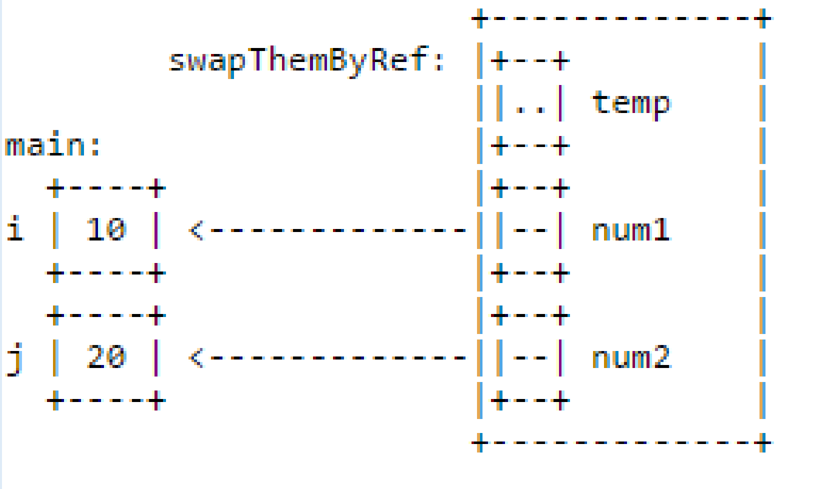
The contents of memory of i and j don't change. The contents of memory in the function changes, but when the function terminates, the memory is released and the changes are lost.



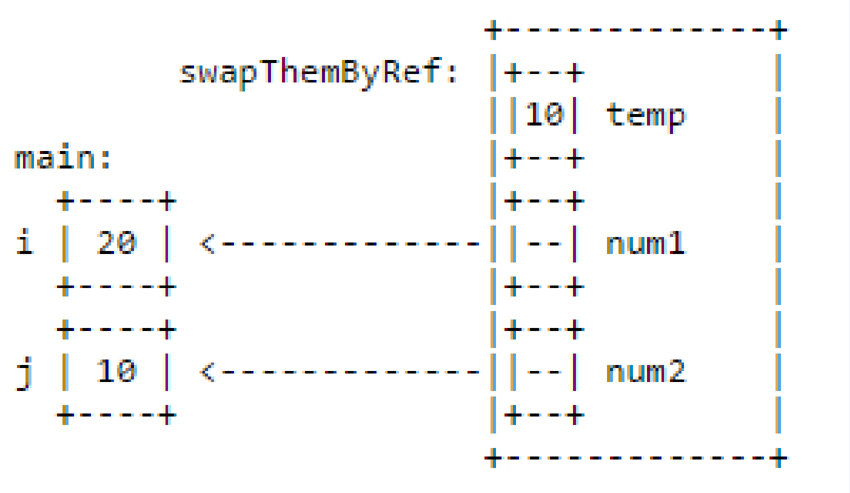
```
                       +--------------+                +--------------+
        swapThemByVal: |+--+          | swapThemByVal: |+--+          |
        (at start)     ||..| temp     | (after         ||10| temp     |
main:                  |+--+          |  assignments)  |+--+          |
  +----+               |+--+          |                |+--+          |
i | 10 |               ||10| num1     |                ||20| num1     |
  +----+               |+--+          |                |+--+          |
  +----+               |+--+          |                |+--+          |
j | 20 |               ||20| num2     |                ||10| num2     |
  +----+               |+--+          |                |+--+          |
                       +--------------+                +--------------+
```

Solution ☺

Pass parameters to swap function by reference.

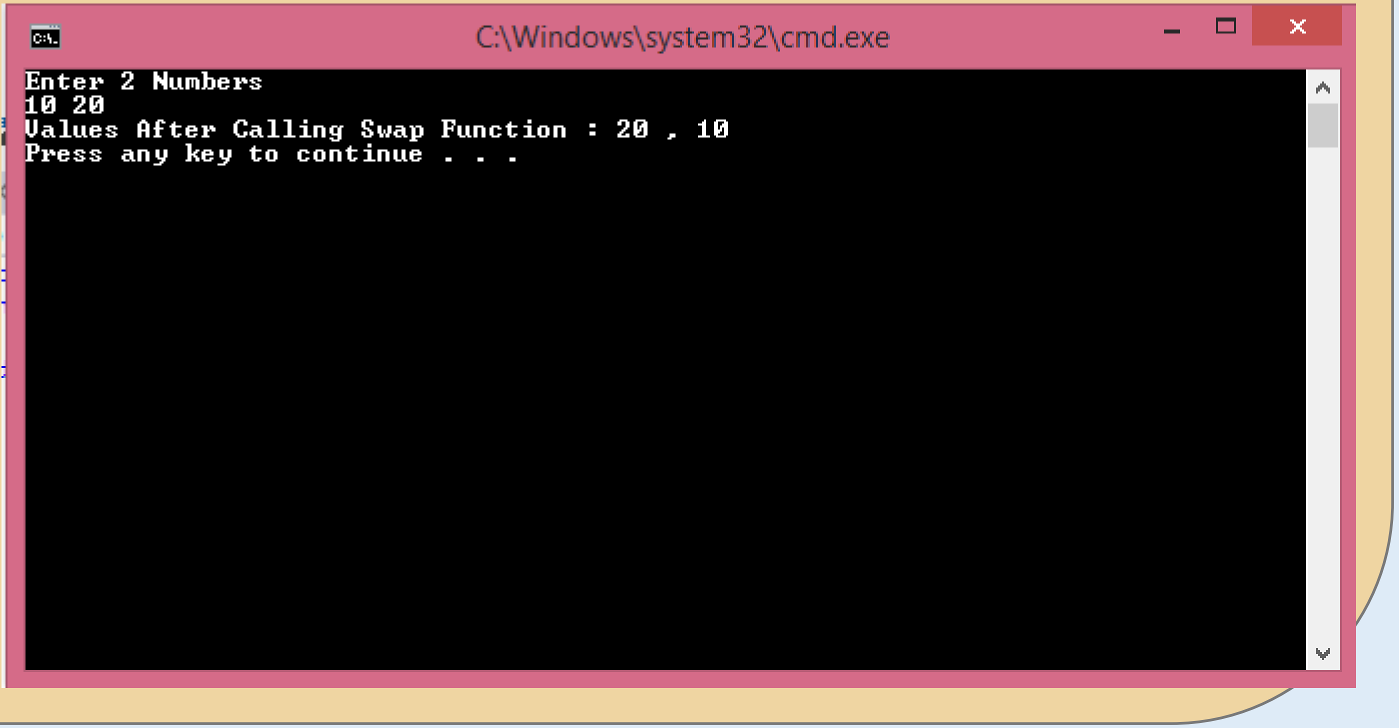


```
                      +-------------+
        swapThemByRef: |+--+        |
                       ||..| temp   |
main:                  |+--+        |
   +----+              |+--+        |
i  | 10 | <------------||--| num1   |
   +----+              |+--+        |
   +----+              |+--+        |
j  | 20 | <------------||--| num2   |
   +----+              |+--+        |
                       +-------------+
```



```
                      +-------------+
        swapThemByRef: |+--+        |
                       ||10| temp   |
main:                  |+--+        |
   +----+              |+--+        |
i  | 20 | <------------||--| num1   |
   +----+              |+--+        |
   +----+              |+--+        |
j  | 10 | <------------||--| num2   |
   +----+              |+--+        |
                       +-------------+
#include <iostream>
using namespace std;

void SwapFunction(int& Num1 , int& Num2);//SwapFunction Declaration (By
Value)

int main()
{

int i,j;
cout << "Enter 2 Numbers " <<endl;
cin >> i >> j;

SwapFunction(i , j);  // SwapFunction Call

// Display Result After SwapFunction Call
cout << "Values After Calling Swap Function : " << i << "," << j;

}
```

```
void SwapFunction(int& Num1 , int & Num2);//SwapFunction Definition
{
    int Temp = Num1;
    Num1 = Num2;
    Num2 = Temp;

}
```



```
Enter 2 Numbers
10 20
Values After Calling Swap Function : 20 , 10
Press any key to continue . . .
```

Ready… Steady… Code!

Ordering Three Numbers

Write a program that reads three data values into num1, num2, and num3 and rearranges them so that they are in increasing order, with the smallest value in num1.

- Hints:

1. Perform this with a function order().
2. Display the 3 numbers after rearrange from the main.
3. Use the previous SwapFunction().

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

void order (int& n1, int& n2, int& n3) {
    if((n1>n2)&&(n1>n3)) // n1 is the largest among them
        swap(n1, n3); // so switch its value with n3
    if((n2>n1)&&(n2>n3)) // n2 is the largest among them
        swap(n2, n3); // so switch its value with n3
    if(n2<n1) // n2 is smaller than the new n1 (old n3)
        swap(n1, n2); // so swap them
}

int main() {
    int num1, num2, num3;
    cin>>num1>>num2>>num3;
    order(num1, num2, num3);
    cout<<num1<<"\t"<<num2<<"\t"<<num3<<endl;
} // end main
```

Functions and Arrays

Write a program to fill an array of five numbers from the user then double each element and display the new list.

- <u>Hints:</u>

1. Perform this using 3 functions input() , Duplicate() , Display().

- <u>Sample of Execution:</u>

```
Enter 5 Numbers: 10 20 1 2 6
Results : 20 40 2 4 12
```

```
int main() {
    int arr[NUM] = {};

    input(arr);
    Duplicate(arr);
    display(arr);
} // end main
```

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

#define NUM 5

void input(int a[]) {
    for (int i=0; i<NUM; i++)
        cin>>a[i];
} // end input
void display(int a[]) {
    for (int i=0; i<NUM; i++)
        cout<<a[i]<<"\t";
    cout<<endl;
} // end display
void Duplicate(int a[]) {
    for (int i=0; i<NUM; i++)
        a[i] *=2;
} // end Duplicate
```

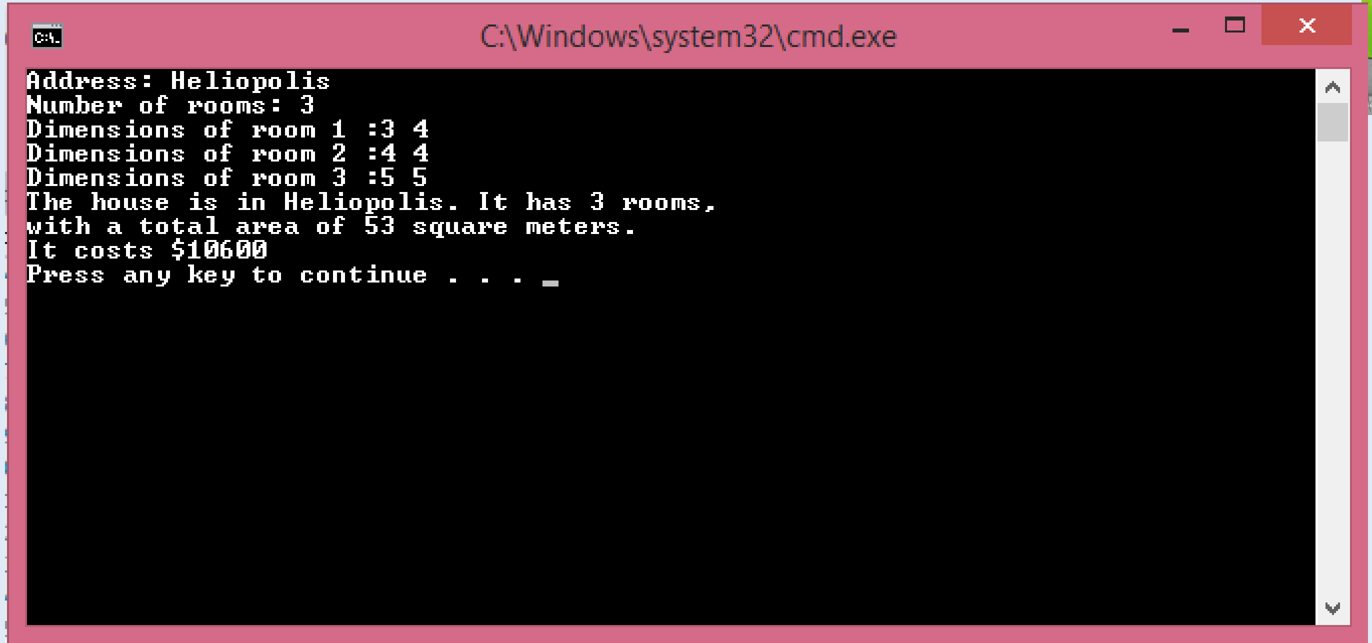
Notice NOT using the & in the input function but still the array changes

Functions and Structures

Write a program that:

- Represent a house as a structure. Each house is specified by its address, number of rooms, area, and price. Each room is specified by its dimensions.
- Collect data for one house and compute/display its data and price.
- Assume the price for the square meter is $200.

Sample of Execution:



```
Address: Heliopolis
Number of rooms: 3
Dimensions of room 1 :3 4
Dimensions of room 2 :4 4
Dimensions of room 3 :5 5
The house is in Heliopolis. It has 3 rooms,
with a total area of 53 square meters.
It costs $10600
Press any key to continue . . .
```

```cpp
#include <iostream>
#include <string>
using namespace std;
#define MAX_ROOMS 10
#define METER_PRICE 200

struct room {
    double length, width;
};
struct house {
    string address;
    int num_rooms;
    room rooms[MAX_ROOMS];
    double area, price;
};
```

```
void input(house & h) {
    cout<<"Address: ";
    cin>> h.address;
    cout<<"Number of rooms: ";
    cin>>h.num_rooms;
    for(int i=0; i<h.num_rooms; i++) {
        cout<<"Dimensions of room "<<i+1<<" :";
        cin>>h.rooms[i].length>>h.rooms[i].width;
    } // end for
} // end input
void display(house h) {
    cout<<"The house is in "<<h.address;
    cout<<". It has "; cout<<h.num_rooms<<" rooms, “ << endl;
    cout<<"with a total area of ";
    cout<<h.area<<" square meters. “<< endl;
    cout<<"It costs $“ <<h.price<<endl;
} // end display
```

```
void calc(house & h) {
   h.area = 0;
   for(int i=0; i<h.num_rooms; i++)
      h.area += (h.rooms[i].length*h.rooms[i].width);
   h.price = h.area*METER_PRICE;
} // end calc

int main() {
   house myHouse = {};
   input(myHouse);
   calc(myHouse);
   display(myHouse);
} // end main
```

Functions and Array of Structures

- Modify the previous program so that you are able to record/display the data of a neighborhood of several houses.
- Add to each house an id indicating its location in the street , address, number of rooms, area, and price.

Thank you!