University American College Skopje

Course: Object Programming

Pointers

Exercises

Prepared by: Biljana Stojčevska, PhD.
Edited for Spring semester 2013/14 by
Zlatko Ivanovski, M.Sc.

Assignment 1

- The following program illustrates operators * and **&.** The pointer **dPtr** points to the variable **x,** and then to the variable **y.**

Assignment 1

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

void main()
{
   double x, y;
   double *dPtr;    // a pointer to a double value

   x=10;
   y=25.3;

   dPtr = &x;  //dPtr obtains the value of the address of x

   cout << "The address of x is " << &x
           << "\nThe value of dPtr is " << dPtr;

   cout << "\n\nThe value of x is " << x
           << "\nThe value of *dPtr is " << *dPtr;

   cout << "\n\nDemonstrating that * and & are inverses to one another "
           << "\n&*dPtr = " << &*dPtr
           << "\n*&dPtr = " << *&dPtr <<endl;...
```

Assignment 1

```
...
 dPtr=&y; //dPtr obtains the value of the address of y

   cout << "\n\nThe address of y is " << &y
           << "\nThe value of dPtr is " << dPtr;

   cout << "\n\nThe value of y is " << y
           << "\nThe value of *dPtr is " << *dPtr;

   *dPtr=0.5; //this statement actually assigns a new value to y

   cout << "\n\nThe address of y is " << &y
           << "\nThe value of dPtr is " << dPtr;

   cout << "\n\nThe vallue of y is " << y
           << "\nThe value of *dPtr is " << *dPtr;


   cout << "\n\nDemonstrating that * and & are inverses to one another "
           << "\n&*dPtr = " << &*dPtr
           << "\n*&dPtr = " << *&dPtr <<endl;
}
```

Assignment 1

- Output:

```
The address of x is 0012FF40
The value of dPtr is 0012FF40

The value of x is 10
The value of *dPtr is 10

Demonstrating that * and & are inverses to one another
&*dPtr = 0012FF40
*&dPtr = 0012FF40


The address of y is 0012FF38
The value of dPtr is 0012FF38

The value of y is 25.3
The value of *dPtr is 25.3

The address of y is 0012FF38
The value of dPtr is 0012FF38

The vallue of y is 0.5
The value of *dPtr is 0.5

Demonstrating that * and & are inverses to one another
&*dPtr = 0012FF38
*&dPtr = 0012FF38
Press any key to continue
```

Assignment 2

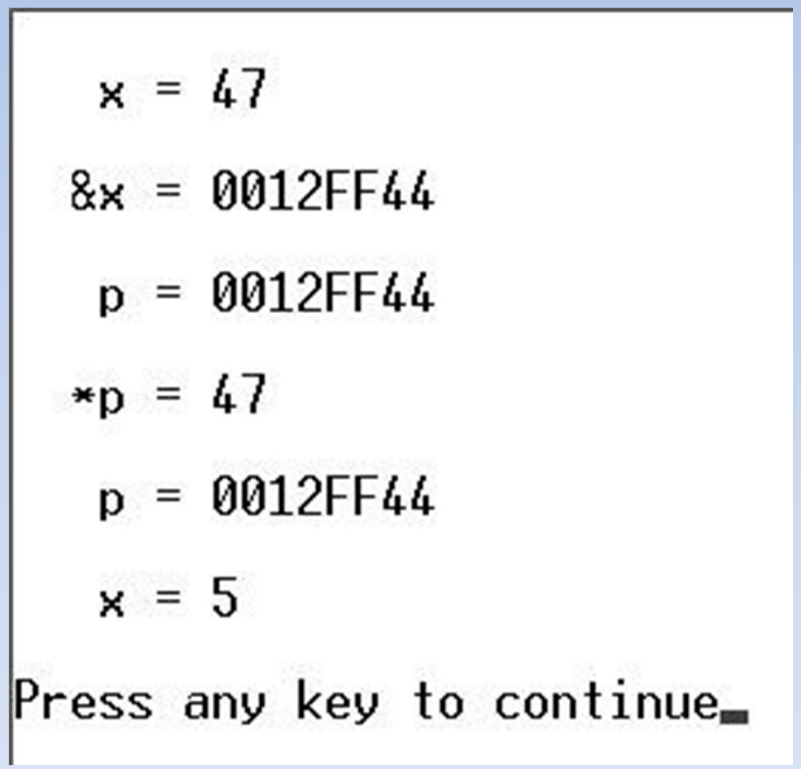
- The following program illustrates passing a pointer as a function argument.

Assignment 2

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

void f(int* p);

void main()
{
  int x = 47;
  cout << "\n   x = " << x << endl;
  cout << "\n  &x = " << &x << endl;
  f(&x);            // the address of x is given as an argument
  cout << "\n   x = " << x << endl<< endl;
}

void f(int* p)
{
  cout << "\n   p = " << p << endl;
  cout << "\n  *p = " << *p << endl;
  *p = 5;
  cout << "\n   p = " << p << endl;
}
```

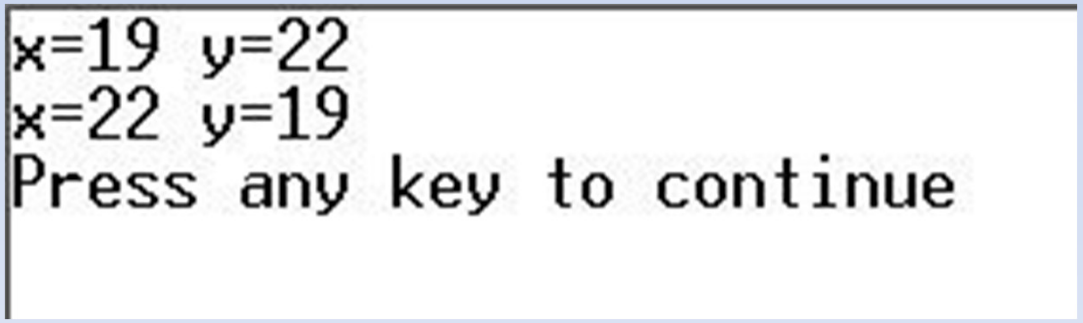
Assignment 2

- Output:



Assignment 3

- Create a program that will swap the values of its two parameters in a function. Use pointers to simulate passing by reference.



Assignment 4

- The pointers are tightly coupled to arrays.

- The next example shows that the name of the array **a** has the value of the address of the element **a[0],** and in fact is a pointer which points to the location of the first element of the array.

Assignment 4

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


int main() {
  int a[10];
  cout << "    a = " << a << endl;
  cout << "&a[0] = " << &a[0] << endl<< endl;
}
```

- Output:

```
    a = 0012FF20
&a[0] = 0012FF20

Press any key to continue
```

Assignment 5

- What is the output from the following program?

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

void main() {
   int a[10];
   int i;
   int *ip = a;

   a=ip;
}
```

Assignment 6

- Input an array of integers (at most 50) through the keyboard. Then, output all of the elements of the array (and their addresses) using a pointer that points to the beginning of the array and accesses the individual elements using:
  - Indices
  - Pointer arithmetics

Assignment 6

- Output:

```
How many elements will the array have (at most 50)? 5
a[0] = 54
a[1] = 8
a[2] = 3
a[3] = 80
a[4] = 86

Outputting the elements of the array using pointer indices:
ip[0] = 54      &ip[0] = 0012FE80
ip[1] = 8       &ip[1] = 0012FE84
ip[2] = 3       &ip[2] = 0012FE88
ip[3] = 80      &ip[3] = 0012FE8C
ip[4] = 86      &ip[4] = 0012FE90

Outputting the elements of the array using pointer arithmetics:
*(ip+0) = 54    (ip+0) = 0012FE80
*(ip+1) = 8     (ip+1) = 0012FE84
*(ip+2) = 3     (ip+2) = 0012FE88
*(ip+3) = 80    (ip+3) = 0012FE8C
*(ip+4) = 86    (ip+4) = 0012FE90
Press any key to continue
```

Assignment 7

- The next program demonstrates dynamic memory allocation using the operators **new** and **delete**.

Assignment 7

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

void main()
{

      int* ip = new int;

      *ip=10;

      cout << "ip = " << (long)ip << " *ip="<<*ip<<endl;

      delete ip;

      cout << "ip = " << (long)ip << " *ip="<<*ip<<endl;
      //The variable created with new is deleted
      //ip now points to an unallocated memory location
}
```

Casting to a `long` data type – useful for representing memory addresses in an integer format (casting to an `int` is also acceptable)

Assignment 7

- Output:

```
ip = 22091840 *ip=10
ip = 22091840 *ip=-572662307
Press any key to continue
```

Assignment 8

- Create a program that will let the user input an array through the keyboard. The size of the array should be dynamically allocated, after the user enters the size of the array
- After the array is input, output it on the screen, and then release the memory that the array holds (i.e. delete the array from memory)

Assignment 8

- Output:

```
Enter the number of elements in the array: 8
a[0] = 64
a[1] = 3
a[2] = 6
a[3] = 90
a[4] = 23
a[5] = 6
a[6] = 246
a[7] = 8

The beginning element of the array is 64
The array elements are:
a[0] = 64
a[1] = 3
a[2] = 6
a[3] = 90
a[4] = 23
a[5] = 6
a[6] = 246
a[7] = 8

The array is deleted. The beginning element of the array now is -572662307
Press any key to continue
```