Exercise -- Pointers & Arrays

Run. Study what's happening.

```c
#include <stdio.h>

int my_array[] = {1,23,17,4,-5,100};
int *ptr;

int main(void)
{
    int i;
    ptr = &my_array[0]; /* point our pointer to the first
              element of the array */
    printf("\n");
    for (i = 0; i < 6; i++)
      {
      printf("A) my_array[%d] = %d \n",i,my_array[i]);
      printf("B) ptr + %d = %d \n\n",i, *(ptr + i));
      }
    return 0;
}
```

Exercise -- Pointers & Arrays

Then Change line B to read:

`printf("ptr + %d = %d\n",i, *ptr++);` and run it again.

Then change it to:

`printf("ptr + %d = %d\n",i, *(++ptr));` and try once more.

Each time try and predict the outcome and carefully look at the actual outcome.

http://www.programiz.com/c-programming/c-pointers

Write a main() program that prints an n by n square of characters. Divide the square into quarters, and fill each quarter with a different character. Here is the output when n==15:

For extra fun, make n and the characters for each quadrant command line parameters. (But the answer does not do this.)

What does the following code write to the monitor?

```
#include <stdio.h>
int main( void )
{
  int  a = 44 ;
  int  b = 66 ;
  int *pa, *pb ;

  pa = &a ;
  pb = &b ;
  printf("*pa=%d *pb=%d\n", *pa, *pb );

  *pa = *pb ;
  printf("a=%d b=%d\n", a, b );

  system("pause");
  return 0;
}
```

What does the following code write to the monitor?

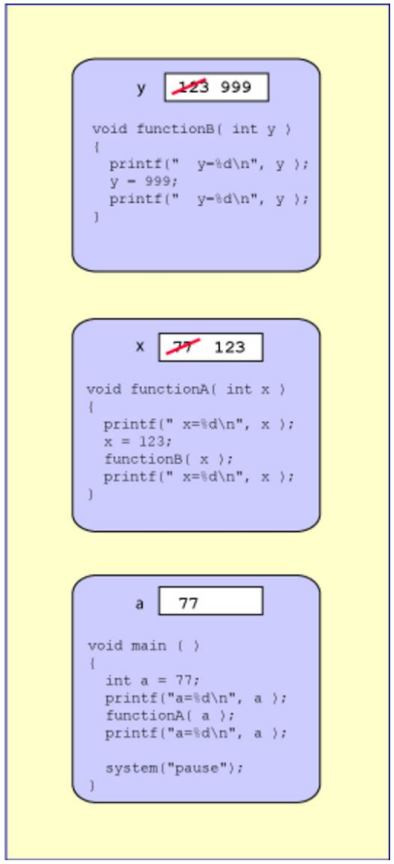
```
#include <stdio.h>

void functionB( int y )
{
  printf("    y=%d\n", y );
  y = 999;
  printf("    y=%d\n", y );
}

void functionA( int x )
{
  printf("  x=%d\n", x );
  x = 123;
  functionB( x );
  printf("  x=%d\n", x );
}

void main ( void )
{
  int a = 77;
  printf("a=%d\n", a );
  functionA( a );
  printf("a=%d\n", a );

  system("pause");
}
```



In this puzzle `main()` calls `functionA()` which calls `functionB()`. Of course, call by value is used for all these calls and changes made by a function to its parameter do not affect any variable in the caller.

In the following, `main()` calls `swap()` to reverse the values in the variables `a` and `b`. But what really happens?

```
#include <stdio.h>

void swap( int x, int y )
{
  int temp;

  printf("  x=%d  y=%d\n", x, y ) ;
  temp = x;
  x = y;
  y = temp;
  printf("  x=%d  y=%d\n", x, y ) ;

}

void main ( void )
{
  int a = 44, b = 77 ;

  printf("a=%d  b=%d\n", a, b ) ;
  swap( a, b ) ;
  printf("a=%d  b=%d\n", a, b ) ;

  system("pause") ;
}
```

This puzzle shows a classic C programming error. Be sure you understand what goes wrong.

Then, fix the swap function.