Pointers

Remember that :

```
printf( "Address of iptr itself: %p\n", &iptr );
printf( "iptr points at        : %p\n",  iptr );
printf( "Address of ival       : %p\n", &ival );
```

Assignment 7

Make a program with two variables of the type int x=10 and y=20.
We would like to let the program printout the value in the variable, as well as the address.
When we make use of a pointer we need to create it with the right type.

```
int  x  = 10;
int  y  = 20;
int *ptr_x;
int *ptr_y;
```

now we let the pointer point at the addresses of the variables.

```
ptr_x = &x;
ptr_y = &y;
```

Now print the values.

```
printf( ”x = %d y=%d\n”, *ptr_x,*ptr_y );
```

Now print the addresses.

```
printf( ”x = %d y=%d\n”, &x, &y );
```

You can use pointers to transfer data to and from a function.
You can use pointers to work through arrays.
Is it a multi dimensional array, you are working with a table.

Assignment 8

To see how pointers are working an example could be a Bobblesort.

```
#include<stdio.h>
void swap(int *a,int *b);


main()
{
    int i, j, n;
    int ar[]  = { 7, 3, 9, 2, 11};


    printf("Array before sort:\n\n");
    for(i=0;i<5;i++)
       printf( "ar[%d]=%d\n",i, ar[i] );


    n  =  5; /*numberof items in sort array*/
    for ( i=0; i<n-1; i++ )
    {
       for( j=0; j<n-1; j++)
       {
           if (   ar[j] > ar[j+1] )
              swap( &ar[j], &ar[j+1] );
       }
    }

    printf( "Array after sort:\n\n" );
    for( i=0; i<5; i++ )
        printf( "ar[%d]=%d\n", i, ar[i] );
    system( "pause" );
    return 0;
}

void swap( int *a, int *b)
{
    int temp;
    temp = *a;
    *a   = *b;
    *b   =  temp;
}
```

What will the printout of this program be ?

Assignment 9

Take the program/code from Assignment 8 and change ANY index reference to an array position so you use pointers instead.

Assignment 10

Make a new program, declare an one dimensional array by the type of int, assign the array with the values by using a pointer(s).

```
      1    2    3    4    5    6    7    8    9   10
     11   12   13   14   15   16   17   18   19   20
```

up to

```
     91   92   93   94   95   96   97   98   99   100
```

Now call a procedure to make a formatted printout the values from the array. Again use pointer(s).

Assignment 11

Now we want to represent the array in **Assignment 10** in 10 1-dimensional arrays. I.e. we need 10 different pointers, one for each row.

Construct the code where this 10 different pointers are stored in an array of pointers.

Make some check printouts.

Assignment 12

Take the array of pointers from Assignment 11 and make a new pointer to point to any of the 10 pointers in the array.

Make this pointer circular. That is, if you add 1 to the 10th array reference, the result must be the 1st array.

Make a check printout, both of this pointer and the pointer it points to.