OBJECTIVES

1. Write functions accepting multiple pointers.
2. Learn Bubble-Sort algorithm

GENERAL INFORMATION

By now, you should have a feeling how important pointers and arrays are in programming, especially in technical/engineering fields. Here is a simple example problem to work on;
A function accepts three pointers to integers. The first one carries both negative and positive numbers. Your function copies positive and negative numbers into second and third arrays respectively. Example;

```c
int A[9]={10,-12,-5,24,-7,9,-5,10,12}, B[9],C[9];
MyFunc(A,9,B,C); /* separate negatives and positives */
/* B now has 10,24,9,10,12 and C now has -12,-5,-7,-5 */
```

EXPERIMENT

1. Write the function described above and test it by printing out the resulting array elements in function main.

2. Sorting arrays according to a criterion is a standard process in many applications. Bubble-Sort is a well-known sorting algorithm and it works by traversing the array several times (until it is sorted) and swapping the consecutive elements according to the sort criterion. Example; (sort smaller to bigger)

```c
int A[9]={10,-12,-5,24,-7,9,-5,10,12};
Starting from the first two elements 10 and -12 (indexed by 0 and 1), since -12 is smaller swap them, so the array now has {-12,10,-5,24,-7,9,-5,10,12}. Now, move to the second pair 10 and -5 (indexed by 1 and 2). Obviously they should be swapped again and the array should now have {-12,-5,10,24,-7,9,-5,10,12}. Continuing in this fashion at the end of the loop, the array should have a partly sorted list, {-12,-5,10,-7,9,-5,10,12,24}. The operation must be repeated until no swapping takes place. It is clear that two nested loops are needed, one for traversing the array and swapping as necessary, and one for the repeated operations until nothing is changed. Write the function void BSort(int A[], int N);
Test your function by printing out the final array in function main().
```

QUESTIONS:

1. You know how to find the biggest valued element in an array. Try to find biggest 3 elements in an array and place them into the second array with size 3. That is
```c
void Biggest3(int A[], int N, int B[]); /* B has the biggest 3 of A */
```
2. Assume that you have a very long string and you are assigned to count the number of occurrences of each character that exists in the string. For example, the string is "this is supposed to be one damn long character array" and the two output array should be
"this upoedbnamlgcry"{2,2,2,4,9,1,2,3,4,2,1,3,5,1,1,1,2,1}
Can you write the function that does that?