Use pointers as much as possible.

1. Read a string from user. The program should then calculate the frequency of occurrence of characters in the input string. It should store the results in a two dimentional array - first column is for the characters and 2nd column is for their frequency (you can use 2nd column of type char to store int, as we can safely assume that the frequency number would fit in one byte). It should then display the 2-column table of char-frequency.

Input : "I am a programmer"

```
Output : [
          ['i','1'],
          ['a','3'],
          ['m','3'],
          ['p','1'],
          ['r','3'],
          ['o','1'],
          ['g','1'],
          ['e','1']
         ]
```

2. Read a string from user. The program then should display the string with its charaters in alphabetic order.

Input : "I can do coding"

Output : "I acn do cdgino"

3. Read five strings from user. Find the longest string among the strings input and display it along with its length.

```
Input :
"a"
"ab"
"xyz"
"pqrs"
"x"

Output :
"pqrs"
length = 4
```

4. There is a treasure that can be opened by a specific combination of three integers. These three integers are to be selected from numbers stored in a 3-d array, secrets[PAGES][ROWS][COLS] (take PAGES=ROWS=COLS=3).
The owner of this treasure knows a number whose individual digits are indices of numbers to be picked up from 1st, 2nd and 3rd page of the secrets array. Eg. if the number is 259, it means pick the 2nd number from page 1, 5th from page 2 and 9th from page 3.
Write a program that first fills up secrets array using numbers input by user. Then it takes the number from owner of treasure and displays the secret combination of 3 numbers.

Input :

```
int arr[3][3][3]=
         { {{1,2,3},{4,5,6},{7,8,9}},
           {{10,11,12},{13,14,15},{16,17,18}},
           {{19,20,21},{22,23,24},{25,26,27}}
         };
```

number = 259

Output :

21427

Explanation :

```
page(1)   1 2 3
          4 5 6
          7 8 9
                  3x3


page(2)   10 11 12
          13 14 15
          16 17 18
                    3x3


page(3)   19 20 21
          22 23 24
          25 26 27
                    3x3
```

5. Simulate a chat UI in a 2-d array of characters. We can input few strings from console and consider every even string as reponse to the previous one. First store strings got from console in a 2-d array normally. Then from this array, store these input strings in a another two dimentional array such that 1,3,5... strings (which are assumed from chat user 1) should be left aligned and 2,4,6... (which from chat user 2) should be right aligned. This is your 'model' of data. Now we want to show it to user also - i.e. create a view. So copy these aligned stings to yet another array that is padded with spaces so that when you display this padded array on console using a pointer, it should show chat strings aligned.
Eg.
Input is -
Hello
hi
how are you
i'm fine, how about you?
me fine too

In the 'model' array (* signfies don't-care char as the string is null

terminated, _ signfies null character):
Hello_**********
**********hi_
how are you_**********
*****i'm fine, how about you?_
me fine too_**********

In the 'view' array- (* signfies space character and _ signfies null character - as we want to see the output just like in this array)
Hello**********_
**********hi_
how are you**********_
******i'm fine, how about you?_
me fine too**********_

Note both * and _ are for describing the question, not for your program. It should show space character as space.