1. Write a program to print X star pattern series using loop.

| Input | Output |
| --- | --- |
| 5 | ```
*       *
  *   *
    * *
     *
    * *
  *   *
*       *
``` |

2. Given an array of numbers. Print frequency of each unique number. (Frequency is the count of particular element divided by the count of all elements)

| Input | Output |
| --- | --- |
| [1, 1, 2, 2, 3] | 1: 0.4<br>2: 0.4<br>3: 0.2 |
| [4, 4] | 4: 1 |
| [1, 2, 3] | 1: 0.3333333333333333<br>2: 0.3333333333333333<br>3: 0.3333333333333333 |

3. Given an array of strings and numbers. Print the number of integers and the number of strings in the array.

| Input | Output |
| --- | --- |
| [1, '10', 'hi', 2, 3] | "Numbers: 3, Strings: 2" |
| [1, 4, 'i am a string', '456'] | "Numbers: 2, Strings: 2" |

4.  Write a function that accepts a string(a sentence) as a parameter and finds the longest word within the string   If there are several words which are the longest ones, print the last word(words can be separated by space, comma or hyphen).

| Input | Output |
|---|---|
| "A revolution without dancing is a revolution not worth having." | "revolution" |
| "Which would be worse - to live as a monster, or to die as a good man?" | "monster" |

5.  Write a function to find longest substring in a given a string without repeating characters except space character.
    If there are several, return the last one. Consider that all letters are lowercase.

| Input | Output |
|---|---|
| "there are no two words in the english language more harmful than "good job"." | "rmful than "go" |
| "parting your soup is not a miracle, bruce." | "up is not a m" |

6.  Write a function to compute a new string from the given one by moving the first char to come after the next two chars, so "abc" yields "bca". Repeat this process for each subsequent group of 3 chars. Ignore any group of fewer than 3 chars at the end.

| Input | Output |
|---|---|
| "dfgjkloyp" | "fgdkljypo" |
| "aweyoolp" | "weaooylp" |

7. Write a function, which receives an array as an argument which elements arrays of numbers. Find biggest negative number of each array. Return product of that numbers.If there is not any negative number in an array, ignore that one. Check that items of the given array are arrays.

| Input | Output |
|---|---|
| [[2, -9, -3, 0], [1, 2], [-4 , -11, 0]] | 12 |
| [[3, 4], [11, 0], [5, 6, 7, 8]] | 'No negatives' |
| [1, 2, 3] | 'Invalid Argument' |

8. Write a JavaScript function to get all possible subsets of length 3 of the given array. Assume that all elements in the array are unique.

| Input | Output |
|---|---|
| [4] | [4] |
| [19, 6] | [19, 6] |
| [5, 9, 23, 0, -2, -1] | [[5, 9, 23], [5, 9, 0], [5, 9, -2], [5, 9, -1], [5, 23, 0], [5, 23, -2], [5, 23, -1], [5, 0, -2], [5, 0, -1], [5, -2, -1], [9, 23, 0], [9, 23, -2], [9, 23, -1], [9, 0, -2], [9, 0, -1], [9, -2, -1], [23, 0, -2], [23, 0, -1], [23, -2, -1], [0, -2, -1]] |