Assignment-4

1. Take ‘n’ as input and print the following pattern for n = 5 using recursion:

```
1
1       1
1       2       1
1       3       3       1
1       4       6       4       1
1       5       10      10      5       1
```

2. Take ‘n’ as input and write a recursive function to return the sum of 1 to n.
   Eg: for n=5 return 15 (1+2+3+4+5)

3. Given an array and a target value, write a recursive function that will return the last index of the occurrence of that target value. If not present in the array return -1.
   Eg: {3 2 1 8 6 1 3} target = 1
   Output: 5

4. Take an array as input and a target value. Write a recursive function which returns an array having indices stored in it at which the target value is stored in the original array.
   Eg: {0, 4, 2, 2, 1, 2, 3, 4, 2} target = 2
   Output: {2, 3, 5, 8}

5. Write a recursive function to reverse an array.

6. Write a recursive function to find the inverse of an array. (for inverse refer assignment-2)

7. Write a recursive function for Bubble Sort. (Using no loops)
8. Write a recursive function for Selection Sort. (Using no loops)

9. Take as input str, a string. Write a recursive function that checks if the string was generated using the following rules and returns a Boolean value: a. the string begins with an 'a' b. each 'a' is followed by nothing or an 'a' or "bb" c. each "bb" is followed by nothing or an 'a' Print the value returned.
   Eg: abba
   Output: true

10. Take as input str, a string. A “twin” is defined as two instances of a char separated by a char. E.g. "AxA" the A's make a “twin”. “twins” can overlap, so "AxAxA" contains 3 “twins” - 2 for A and 1 for x. Write a function which recursively counts the number of “twins” in a string. Print the value returned.
    Eg: AXAXA
    Output: 3

11. Take as input str, a string. We are concerned with all the possible ascii-subsequences of str. E.g. “ab” has following ascii-subsequences “”, “b”, “98”, “a”, “ab”, “a98”, “97”, “97b”, “9798”
    a. Write a recursive function which returns the count of ascii-subsequences for a given string. Print the value returned.
    b. Write a recursive function which prints all possible ascii-subsequences for a given string (void is the return type for function).

    Eg: ab
    Output: b 98 a ab a98 97 97b 9798
    9

12. Take as input str, a string. Write a recursive function which returns a new string in which all duplicate consecutive characters are separated by a ‘ - ’. E.g. for “hello” return “hel-lo”.

13. Take as input str, a string. Write a recursive function which returns a new string in which all duplicate consecutive characters are removed. E.g. for “heeeello” return “helo”.

14. Take as input str, a string. The string is a mathematical expression. Eg: “[a + {b + (c+d) + e} + f]”. Write a recursive function which checks if the brackets are balanced or not.

15. Take as input N, a number. Take N more inputs and store that in an array.
    a. Write a recursive function which counts the number of ways the array could be split in two groups, so that the sum of items in both groups is equal. Each number in the array must belong to one of the two groups. Print the value returned.
    b. Write a recursive function which keeps track of ways an array could be split in two groups, so that the sum of items in both groups is equal. Each number in the array must belong to one of the two groups. Return type of function should be void. Print the two groups, each time you find a successful split.
    Eg: 6
    1 2 3 3 4 5
    Output: 6
    1 2 3 3 and 4 5
    1 3 5 and 2 3 4
    1 3 5 and 2 3 4
    2 3 4 and 1 3 5
    2 3 4 and 1 3 5
    4 5 and 1 2 3 3