Assignment-5

1. Take as input str, a string. Write a recursive function which moves all ‘x’ from the string to its end. Print the value returned
   E.g. for “abexexdexed” return “abeedeedxxx”.

2. Take as input row and column of a rectangular board. The rat starts its journey from the top left corner and must reach the bottom-right corner to get the cheese. In one move the rat can move 1 step horizontally (right) or 1 step vertically (down).
   a. Write a recursive function which returns an ArrayList of moves for all valid paths across the board. Print the value returned.
      E.g. for a board of size 3,3; a few valid paths will be “HHVV”, “VVHH”, “VHHV” etc.

3. Take as input row and column of a rectangular board. The rat starts its journey from the top left corner and must reach the bottom-right corner to get the cheese. In one move the rat can move 1 step horizontally (right) or 1 step vertically (down) or 1 step diagonally (south-east).
   a. Write a recursive function which returns an ArrayList of moves for all valid paths across the board. Print the value returned. e.g. for a board of size 3,3; a few valid paths will be “HHVV”, “VVHH”, “VHHV”, “DD”, “VDH”, “HVD” etc.

4. You are given an N*M grid. Each cell (i,j) in the grid is either blocked, or empty. The rat can move from position (i,j), down or right on the grid. Initially rat is on the position (1,1). It wants to reach position (N,M). Find the rightmost path through which, rat can reach this position. A path is rightmost, if the rat is at position (i,j), it will always move to (i,j+1), if there exists a path from (i,j+1) to (N,M).

   Eg:
   ```
   5 4                  Output: 1 0 0 0
   OXOO                         1 1 0 0
   OOOX                         0 1 0 0
   OOXO                         0 1 1 1
   XOOO                         0 0 0 1
   XXOO
   ```

5. Take as input str, a string. Assume that value of a=1, b=2, c=3, d=4, ... z=26. Write a recursive function (return type void) to print all possible codes for the string. E.g: “1123” -> possible codes are aabc, kbc, alc, aaw, kw.

6. Take as input str, a string. str represents keys pressed on a nokia phone keypad. We are concerned with all possible words that can be written with the pressed keys.
   E.g: “12” can produce following words “ad”, “ae”, “af”, “bd”, “be”, “bf”, “cd”, “ce”, “cf”
   a. Write a recursive function which returns the count of words for a given keypad string. Print the value returned.
   b. Write a recursive function which returns an ArrayList of words for a given keypad string. Print the value returned.
   c. Write a recursive function which prints all possible words for a given keypad string (void is the return type for function).

7. Take as input str, a string. We are concerned with all possible permutations of characters in str.
   E.g. “abc” can produce following words “abc”, “acb”, “bac”, “bca”, “cab”, “cba”
   a. Write a recursive function which returns the count of different permutations for a given string. Print the value returned.
   b. Write a recursive function which returns an ArrayList of permutations for a given string. Print the value returned.
   c. Write a recursive function which prints all possible permutations for a given string (void is the return type for function).

8. Take an array as input and write recursive functions for the following:
   a. Count the number of ways in which you can divide the array in two parts such that the sum of elements of both the parts will be equal. Each number in the array must belong to one of the two parts.
   b. Print the two parts in the form of arraylist. All possible answers should be printed. Eg: for {-1,3,7,4,2,1} answers will be
      [-1, 3, 4, 2] : [7, 1]
      [-1, 7, 2] : [3, 4, 1]
      [3, 4, 1] : [-1, 7, 2]
      [7, 1] : [-1, 3, 4, 2]

9. Take an array as input & a target value. Then, write a recursive function which returns the subsets of the array which sum to target. Return type of the function should be ArrayList<ArrayList<Integer>>. Print the value returned.

10. Take as input N, a number. Write a recursive function which prints counting from 1 to N in lexicographical order. In lexicographical (dictionary) order, for 11 output will be {1, 10, 11, 2, 3, 4, 5, 6, 7, 8, 9}

11. Take as input ‘n’, the size of a chess board. We are asked to place ‘n’ number of Knights in it, so that no knight can kill others.
    a. Write a recursive function which returns an ArrayList of all valid configurations.
    b. Write a recursive function which prints all valid configurations (return type for the function will be void).

12. You are given an n*n sudoku grid (n is a multiple of 3). Solve the sudoku using backtracking and print the solution.

E.g: 9

```
5 3 0 0 7 0 0 0 0
6 0 0 1 9 5 0 0 0
0 9 8 0 0 0 0 6 0
8 0 0 0 6 0 0 0 3
4 0 0 8 0 3 0 0 1
7 0 0 0 2 0 0 0 6
0 6 0 0 0 0 2 8 0
0 0 0 4 1 9 0 0 5
0 0 0 0 8 0 0 7 9
```

Output:

```
5 3 4 6 7 8 9 1 2
6 7 2 1 9 5 3 4 8
1 9 8 3 4 2 5 6 7
8 5 9 7 6 1 4 2 3
4 2 6 8 5 3 7 9 1
7 1 3 9 2 4 8 5 6
9 6 1 5 3 7 2 8 4
2 8 7 4 1 9 6 3 5
3 4 5 2 8 6 1 7 9
```

13. Leetcode problems:
    a. https://leetcode.com/problems/n-th-tribonacci-number/
    b. https://leetcode.com/problems/combination-sum/
    c. https://leetcode.com/problems/combination-sum-ii/
    d. https://leetcode.com/problems/combinations/
    e. https://leetcode.com/problems/path-with-maximum-gold/
    f. https://leetcode.com/problems/letter-combinations-of-a-phone-number/
    g. https://leetcode.com/problems/generate-parentheses/
    h. https://leetcode.com/problems/beautiful-arrangement/