M T W T F S S
Page No.
Date.
YOUVA

☆ Today's Jobs

- ~~KK~~ - Recursion Level 1
- KK - Recursion Array Questions.
- Leetcode - 3 ~~0~~ + problems + LinkedIn Test.
- Java Bro Code: 2 Hours (GUI Course)
  - Coding Ninjas
- ~~GFG~~ - Initialize + Streak Maintenance.

-> Recursion Level 1 >>

- Sum of Digits] N = 1342 -> 1+3+4+2 = 10

```
public static int sumOfDigits(int n){
    if(n/10==0){   // Base condition
        return n;
    }
    int remainder = n%10;
    return remainder + sumOfDigit(n/10);
}
```

Same can be applied to product of digits

Another Possible base condition
[n/10 == n] ~ return n;

-> Reverse a digits using recursion

Requirements: Length of digit

```
public static void main (String[] args){
    int n = 163257
    int digits = (int) Math.log10(n) + 1
    Sout (reverseDigit(n, digits));
}
public static int reverseDigit(
                    int n, int digits){
    if (n%10 == n) {
        return n; }
    return ( n%10 * (int)Math.pow(10, digits-1)) +
                reverseDigits (n/10, digits-1);
}
```

This doesn't work in Java

-> Based on this function, you can check if the number is palindrome or not.

-> Count a particular digit occurrence in an Integer

- Have to pass the counter in argument to maintain it in between recursive calls.

- You can also take a variable outside or have function default value for counter so not have to pass

-> LeetCode 1342] Count number of steps to reduce a number to zero

-> Check if the array is sorted in non decending order.

```
public static boolean checkSorteArray
                (int[] nums, int index){
  if (index == nums.length-1) return true;
  if (nums[index] <= nums[index+1])
        return checkSortedArray(nums,
                                index+1);
  else return false;
}
```

-> Linear Search with Recursion.

```
public static int recursiveLinearSearch
                (int[] nums, int target, int i){
  if (i == nums.length) return -1;
  if (nums[i] = target) return i;
  else return recursiveLinearSearch
                (nums, target, i+1);
}
```

-> Find all occurences of target value in an unsorted array.

-> Have a static ArrayList<Integer> list
-> Add index if found == target by list.add(index);

-> leetcode 1: Two Sum

-> Leetcode 11: Container with Most Water

-> Self Brute force Approach ~

Formula for area → (j - i) * temp

index of elem second

index of first elem

value of arr[i] or arr[j] which ever is less

-> Instead of brut forcing O(n²) use O(n) complexity with two pointers start and end
→ ←

-> Started Maintenance of streak on Coding Ninjas. Solved 7 Beginner problems and 5 MCQs

-> 1Kk Object Oriented Programming: OOPs - 30 min ✓

-> 2 LinkedIn Posts.