**ASSIGMNET 1**

class RecursionAssignment {

    static void power(int x, int n, int ans) {

        if (n == 0) {

            System.out.println("The answer is-:" + ans);

            return;

        }

        power(x, n - 1, ans \* x);

    }

    static int power(int x, int n) {

        if (n == 0)

            return 1;

        return x \* power(x, n - 1);

    }

    static void countZero(int n, int ans) {

        if (n == 0) {

            System.out.println("The no. of zeros are -:" + ans);

            return;

        }

        if (n % 10 == 0)

            ans++;

        countZero(n / 10, ans);

    }

    static int countZero(int n) {

        if (n == 0)

            return 0;

        if (n % 10 == 0)

            return countZero(n / 10) + 1;

        else

            return countZero(n / 10);

    }

    static void sumOfN(int n, int res) {

        if (n == 0) {

            System.out.println(res);

            return;

        }

        sumOfN(n - 1, res + n);

    }

    static int sumOfN(int n) {

        if (n == 0)

            return 0;

        return n + sumOfN(n - 1);

    }

    static void seriesSum(int n, int res) {

        if (n == 1) {

            System.out.println(res);

            return;

        }

        seriesSum(n - 1, (int) (n / Math.pow(n, n) \* res));

    }

    static int seriesSum(int n) {

        if (n == 1)

            return 0;

        int N = (int) (n / Math.pow(n, n));

        return N \* seriesSum(n - 1);

    }

    static void isPrime(int n, int c, int i) {

        if (i == n) {

            if (c > 2)

                System.out.println(false);

            else

                System.out.println(true);

            return;

        }

        if (n % i == 0)

            isPrime(n, c + 1, i + 1);

        else

            isPrime(n, c, i + 1);

    }

    static int isArmStrong(int n, int p) {

        if (n == 0)

            return 0;

        return (int) Math.pow(n % 10, p) + isArmStrong(n / 10, p);

    }

    static String isPallindrome(String str) {

        if (str.length() == 0)

            return "";

        String sub = str.substring(1);

        return str.charAt(0) + isPallindrome(sub);

    }

    static int searchFromLast(int arr[], int index, int x) {

        if (index == arr.length - 1)

            return -1;

        if (arr[arr.length - 1 - index] == x)

            return arr.length - 1 - index;

        return searchFromLast(arr, index + 1, x);

    }

    static void countOccurences(int arr[], int index, int count, int x) {

        if (index == arr.length - 1) {

            System.out.println(count);

            return;

        }

        if (arr[index] == x)

            countOccurences(arr, index + 1, count + 1, x);

        else

            countOccurences(arr, index + 1, count, x);

    }

    static int countOccurences(int arr[], int index, int x) {

        if (index == arr.length - 1) {

            return 0;

        }

        if (arr[index] == x)

            return countOccurences(arr, index + 1, x) + 1;

        else

            return countOccurences(arr, index + 1, x);

    }

    static void searchAndReplace(int arr[], int x, int r, int index) {

        if (index == arr.length - 1)

            return;

        if (arr[index] == x)

            arr[index] = r;

        searchAndReplace(arr, x, r, index + 1);

    }

    static String replaceWithhash(String str) {

        if (str.length() == 1)

            return str;

        if (str.charAt(0) == str.charAt(1)) {

*// String sub = str.replace(str.charAt(0), '#');*

            return str.charAt(0) + replaceWithhash(str.substring(1).replace(str.charAt(1), '#'));

        }

        return str.charAt(0) + replaceWithhash(str.substring(1));

    }

    static String addAStar(String str) {

        if (str.length() == 2)

            return str;

        if (str.charAt(0) == str.charAt(1)) {

            return str.charAt(0) + (char) '\*' + addAStar(str.substring(1));

        }

        return str.charAt(0) + addAStar(str.substring(1));

    }

    public static void main(String[] args) {

*/\**

*\* power(2, 5, 1);*

*\* System.out.println(power(2, 5));*

*\*/*

*/\**

*\* countZero(1000000, 0);*

*\* System.out.println(countZero(1000000));*

*\*/*

*/\**

*\* sumOfN(3, 0);*

*\* System.out.println(sumOfN(3));*

*\*/*

*/\**

*\* seriesSum(3, 0);*

*\* System.out.println(seriesSum(3));*

*\*/*

*// isPrime(5, 0, 1);*

*/\**

*\* if (isArmStrong(12, 2) == 12)*

*\* System.out.println("Armstrong Number");*

*\* else*

*\* System.out.println("Not Armstrong Number");*

*\*/*

*/\**

*\* if (isPallindrome("POP").compareTo("POP") == 0)*

*\* System.out.println("Pallindrome ");*

*\* else*

*\* System.out.println("Not Pallindrome");*

*\*/*

*/\**

*\* int arr[] = { 2, 5, 6, 7, 4, 3, 9 };*

*\* System.out.println(searchFromLast(arr, 0, 5));*

*\*/*

*/\**

*\* int arr[] = { 3, 5, 3, 7, 4, 3, 9 };*

*\* countOccurences(arr, 0, 0, 7);*

*\* System.out.println(countOccurences(arr, 0, 3));*

*\*/*

*/\**

*\* int arr[] = { 2, 5, 6, 7, 4, 3, 9 };*

*\* searchAndReplace(arr, 6, 10, 0);*

*\* for (int i : arr) {*

*\* System.out.print(i + " ");*

*\* }*

*\*/*

*// System.out.println(replaceWithhash("aabbcc"));*

*// System.out.println(addAStar("aabbcc"));*

    }

}