LOGIC WHILE DAILY ASSESSMENT

Day-1: 25/07/25

1. Sleep of Energy system

def shutdown\_steps(energy\_level):

steps = 0

while energy\_level > 0:

if energy\_level % 2 == 0:

energy\_level //= 2

else:

energy\_level -= 1

steps += 1

return steps

def main():

try:

energy\_level = int(input("Enter the robot's energy level (0 to 10^6): "))

if 0 <= energy\_level <= 10\*\*6:

result = shutdown\_steps(energy\_level)

print("Total steps to shut down the robot:", result)

else:

print("Please enter a value between 0 and 10^6.")

except ValueError:

print("Invalid input. Please enter an integer.")

if \_name\_ == "\_main\_":

    main()

1. Decoding Signals

def decode\_signal(signal\_strength):

steps = 0

while signal\_strength > 0:

remainder = signal\_strength % 3

if remainder == 0:

signal\_strength //= 3

elif remainder == 1:

signal\_strength -= 1

else: # remainder == 2

signal\_strength -= 2

steps += 1

return steps

def main():

try:

signal\_strength = int(input("Enter the alien signal strength (0 to 10^6): "))

if 0 <= signal\_strength <= 10\*\*6:

result = decode\_signal(signal\_strength)

print("Total operations to decode signal:", result)

else:

print("Please enter a value between 0 and 10^6.")

except ValueError:

print("Invalid input. Please enter an integer.")

if \_name\_ == "\_main\_":

    main()

Day-2: 26/07/25

Day-3: 27/07/25

Day-4: 28/07/25

1. Unique License Plate

import java.util.Scanner;

public class UniqueLicensePlate {

public static boolean isUnique(String plate) {

boolean[] seen = new boolean[36];

for (int i = 0; i < plate.length(); i++) {

char ch = plate.charAt(i);

int index;

if (ch >= 'A' && ch <= 'Z') {

index = ch - 'A'; // A-Z → 0–25

} else if (ch >= '0' && ch <= '9') {

index = ch - '0' + 26; // 0–9 → 26–35

} else {

return false;

}

if (seen[index]) {

return false;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the license plate: ");

String plate = scanner.nextLine().toUpperCase(); // Convert to uppercase

boolean result = isUnique(plate);

System.out.println(result);

    }

}

1. Product Except self

import java.util.Scanner;

public class ProductExceptSelf {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter space-separated productivities: ");

String[] input = scanner.nextLine().split(" ");

int n = input.length;

int[] nums = new int[n];

for (int i = 0; i < n; i++) {

nums[i] = Integer.parseInt(input[i]);

}

int[] output = new int[n];

int prefix = 1;

for (int i = 0; i < n; i++) {

output[i] = prefix;

prefix \*= nums[i];

}

int suffix = 1;

for (int i = n - 1; i >= 0; i--) {

output[i] \*= suffix;

suffix \*= nums[i];

}

for (int val : output) {

System.out.print(val + " ");

       }

    }

}

Day-5: 29/07/25

Day-6: 30/07/25

1. Swapping First & Last Characters

Input : welcome to java programming

Output : eelcomw ot aavj grogramminp

import java.util.Scanner;

public class SwapFirstLastCharacters {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String input = sc.nextLine();

char[] chars = input.toCharArray();

int n = chars.length;

int i = 0;

while (i < n) {

while (i < n && chars[i] == ' ') {

i++;

}

int start = i;

while (i < n && chars[i] != ' ') {

i++;

}

int end = i - 1;

if (start < end) {

char temp = chars[start];

chars[start] = chars[end];

chars[end] = temp;

}

}

System.out.println(new String(chars));

    }

}

1. Suspicious Transactions pairs

Input : 6

4000 6000 2000 8000 1000 9000

10000

Output : 4000 6000

2000 8000

1000 9000

import java.util.\*;

public class SuspiciousTransactionPairs {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] transactions = new int[n];

for (int i = 0; i < n; i++) {

transactions[i] = sc.nextInt();

}

int targetSum = sc.nextInt();

Set<Integer> seen = new HashSet<>();

Set<String> printedPairs = new HashSet<>();

boolean found = false;

for (int i = 0; i < n; i++) {

int complement = targetSum - transactions[i];

if (seen.contains(complement)) {

int a = Math.min(transactions[i], complement);

int b = Math.max(transactions[i], complement);

String pair = a + " " + b;

}

}

}

}

Day-7: 31/07/25

1. Second Extremes

Input : 6

12 45 7 89 23 67

Output : 12 67

import java.util.\*;

public class SecondExtremes {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int n = scanner.nextInt();

int[] arr = new int[n];

for (int i = 0; i < n; i++) {

arr[i] = scanner.nextInt();

}

Arrays.sort(arr);

int secondSmallest = arr[0];

for (int i = 1; i < n; i++) {

if (arr[i] != arr[0]) {

secondSmallest = arr[i];

break;

}

}

int secondLargest = arr[n - 1];

for (int i = n - 2; i >= 0; i--) {

if (arr[i] != arr[n - 1]) {

secondLargest = arr[i];

break;

}

}

System.out.println(secondSmallest + " " + secondLargest);

    }

}

1. Case Sensitive

Input: customerBillingAddress

Output: customer Billing Address

import java.util.Scanner;

public class CamelCaseSplitter {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

String input = scanner.nextLine();

StringBuilder output = new StringBuilder();

for (int i = 0; i < input.length(); i++) {

char ch = input.charAt(i);

if (Character.isUpperCase(ch) && i != 0) {

output.append(" ");

}

output.append(ch);

}

System.out.println(output.toString());

    }

}

Day-8: 01/08/25

Day-9: 02/08/25

1. Kth Largest Element

Input: 3 2 1 5 6 4

2

Output: 5

import java.util.\*;

public class KthLargestElement {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the elements of the array (space-separated):");

String[] input = sc.nextLine().split(" ");

int[] arr = new int[input.length];

for (int i = 0; i < input.length; i++) {

arr[i] = Integer.parseInt(input[i]);

}

System.out.print("Enter the value of k: ");

int k = sc.nextInt();

Arrays.sort(arr);

for (int i = 0; i < arr.length / 2; i++) {

int temp = arr[i];

arr[i] = arr[arr.length - 1 - i];

arr[arr.length - 1 - i] = temp;

}

if (k >= 1 && k <= arr.length) {

System.out.println("Kth largest element is: " + arr[k - 1]);

1. Roman To Integer

Input : LVIII

Output : 50

public class Main {

public static int romanToInt(String s) {

Map<Character, Integer> romanMap = new HashMap<>();

romanMap.put('I', 1);

romanMap.put('V', 5);

romanMap.put('X', 10);

romanMap.put('L', 50);

romanMap.put('C', 100);

romanMap.put('D', 500);

romanMap.put('M', 1000);

int total = 0;

int prev = 0;

for (int i = s.length() - 1; i >= 0; i--) {

int curr = romanMap.get(s.charAt(i));

if (curr < prev) {

total -= curr;

} else {

total += curr;

}

prev = curr;

}

return total;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String roman = sc.next().trim(); // Read full input from stdin

System.out.println(romanToInt(roman));

sc.close();

    }

}

Day-10: 03/08/25

1.Bracket Validator

Input : (){}[]

Output : true

import java.util.\*;

public class BracketValidator {

public static boolean isValid(String s) {

Stack<Character> stack = new Stack<>();

for (char c : s.toCharArray()) {

if (c == '(' || c == '{' || c == '[') {

stack.push(c);

} else {

if (stack.isEmpty()) return false;

char top = stack.pop();

if ((c == ')' && top != '(') ||

(c == '}' && top != '{') ||

(c == ']' && top != '[')) {

return false;

}

}

}

return stack.isEmpty();

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the bracket string: ");

String input = sc.nextLine();

if (isValid(input)) {

System.out.println("true");

} else {

System.out.println("false");

       }

    }

}

2. Sorting colors codes

Input: 2 0 2 1 1 0

Output: 0 0 1 1 2 2

import java.util.\*;

public class SortColors {

public static void sortColors(int[] nums) {

int low = 0, mid = 0, high = nums.length - 1;

while (mid <= high) {

if (nums[mid] == 0) {

int temp = nums[low];

nums[low] = nums[mid];

nums[mid] = temp;

low++;

mid++;

} else if (nums[mid] == 1) {

mid++;

} else { // nums[mid] == 2

// swap nums[mid] and nums[high]

int temp = nums[mid];

nums[mid] = nums[high];

nums[high] = temp;

high--;

}

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String[] input = sc.nextLine().split(" ");

int[] nums = new int[input.length];

for (int i = 0; i < input.length; i++) {

nums[i] = Integer.parseInt(input[i]);

}

sortColors(nums);

for (int num : nums) {

System.out.print(num + " ");

       }

    }

}

Day-11: 04/08/25

1. Alien Decoder

Input : abc

2

Output : cde

Import java.util.\*;

public class AlienDecoder {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s = sc.nextLine();

int k = sc.nextInt();

String result = decodeMessage(s, k);

System.out.println(result);

sc.close();

}

public static String decodeMessage(String s, int k) {

StringBuilder decoded = new StringBuilder();

for (char c : s.toCharArray()) {

char shifted = (char) ((c - 'a' + k) % 26 + 'a');

decoded.append(shifted);

}

return decoded.toString();

     }

}

1. Equilibrium Index Finder

Input : 7

-7 1 5 2 -4 3 0

Output : 3

import java.util.Scanner;

public class EquilibriumIndexFinder {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int[] arr = new int[n];

for (int i = 0; i < n; i++) {

arr[i] = sc.nextInt();

}

System.out.println(findEquilibriumIndex(arr));

sc.close();

}

public static int findEquilibriumIndex(int[] arr) {

int totalSum = 0;

for (int num : arr) {

totalSum += num;

}

int leftSum = 0;

for (int i = 0; i < arr.length; i++) {

int rightSum = totalSum - leftSum - arr[i];

if (leftSum == rightSum) {

return i;

}

leftSum += arr[i];

}

return -1;

     }

}