Date: 07/02/2026

Duration: 1 hour 45

CONTINUOUS ASSESSMENT 1

II YEAR – SEMESTER 4

CSE23AE204 - PCP III

(B.Tech. E01,E02,E03,E05,E06)

SET B

Question 1: Problem Statement

You are part of a **capacity analysis team** in an enterprise organization.

A legacy analytics module generates a **numeric utilization snapshot** that visually expands and contracts based on a given scale value.

Each row of the snapshot reflects **row-level indexing** and must be stored as text for automated verification.

Since the analytics engine validates output programmatically, the snapshot must be returned as a **list/array of strings**, where:

- Each string represents **one row**
- Numbers in a row correspond to the **row index**
- The structure is vertically symmetric
- Alignment must be preserved exactly

Your task is to generate this snapshot for a given integer N.

Input Format

- A single integer N representing the snapshot scale.

Output Format

- Return an **array/list of strings**
- Each string represents one row of the snapshot

Example 1

Input

3

Output

```
[
 "33333",
 " 222 ",
 "  1  ",
 " 222 ",
 "33333"
]
```

Question 2: Problem Statement

A large manufacturing plant tracks the workload handled by each robotic unit on an assembly line.

You are given an array loads, where:

- loads[i] = number of tasks completed per minute by robot unit **i**

The factory wants to identify a **Load Balance Point (LBP)**:

A position i where:

Sum of loads strictly to the left of i == Sum of loads strictly to the right of i

This helps determine where to place supervisors and load balancers to maintain efficiency.

Return the **first Load Balance Point index** if one exists.

If no such point exists, return **-1**.

Input Format

loads → integer array

Output Format

integer → pivot index or -1

Example 1

Input:

loads = [1,7,3,6,5,6]

Output:

3

Explanation:

Left sum = 1 + 7 + 3 = 11

Right sum = 5 + 6 = 11

Date: 07/02/2026

Duration: 1 hour 45

CONTINUOUS ASSESSMENT 1

II YEAR – SEMESTER 4

CSE23AE204 - PCP III

(B.Tech. E01,E02,E03,E05,E06)

SET B

Question 3: Problem Statement

Problem Statement

You are developing a **discount calculation module** for an e-commerce platform. The platform processes **one order at a time**, provided as a **space-separated string**. Each order belongs to a specific customer category and follows different discount rules.

The system must:

- Identify the order type from input
- Apply discounts based on customer category
- Calculate the final payable amount
- Use **inheritance and polymorphism** to support multiple order types
- Use **method overloading** to apply different discount strategies
-

Discount Rules

Regular Order

- A percentage-based discount is applied on the original price.
- The discounted amount is subtracted from the original price.
- The remaining value is the final payable amount.

Prime Order

- A percentage-based discount is applied on the original price.
- The discounted amount is subtracted from the original price.
- An additional fixed prime discount is further subtracted.
- The remaining value is the final payable amount.

Input Format

A **single space-separated string**.

Regular Order

REGULAR <CustomerName> <Price> <DiscountPercentage>

Prime Order

PRIME <CustomerName> <Price> <DiscountPercentage> <PrimeDiscount>

Output Format

Return a **double value** representing the **final payable amount**.

Example 1

Example 1

Input

{ "order": "REGULAR Ravi 2000 10" }

Output

1800.0

Explanation

A percentage discount is applied on the original price and subtracted to get the final payable amount.