1. Greedy Algorithm

Question: Minimum Number of Coins

You are given an amount of money and a list of coin denominations. Your task is to find the minimum number of coins needed to make that amount.

Input:

- The first line contains an integer **n** (1 ≤ n ≤ 100), the number of coin denominations.
- The second line contains **n** space-separated integers representing the coin denominations (1 ≤ denomination ≤ 1000).
- The third line contains an integer **amount** (1 ≤ amount ≤ 10000).

Output:

- Print the minimum number of coins needed to make the amount. If it is not possible, print **-1**.

Example:

Input:

3

1 3 4

6

Output:

2

2. Dynamic Programming

Question: Longest Increasing Subsequence

Given an array of integers, find the length of the longest increasing subsequence.

Input:

- The first line contains an integer **n** (1 ≤ n ≤ 1000), the number of elements in the array.
- The second line contains **n** space-separated integers (1 ≤ arr[i] ≤ 10000).

Output:

- Print the length of the longest increasing subsequence.

Example:

Input:

8

10 22 9 33 21 50 41 60

Output:

5

3. Graph Algorithm

Question: Number of Islands

Given a 2D grid of '1's (land) and '0's (water), count the number of islands. An island is surrounded by water and is formed by connecting adjacent lands horizontally or vertically.

Input:

- The first line contains two integers **m** and **n** (1 ≤ m, n ≤ 100), the number of rows and columns in the grid.
- The next **m** lines each contain a string of length **n** consisting of '0's and '1's.

Output:

- Print the number of islands.

Example:

Input:

4 5

11000

11000

00100

00011

Output:

3

4. Divide and Conquer

Question: Merge Sort

Implement the merge sort algorithm to sort an array of integers.

Input:

- The first line contains an integer **n** (1 ≤ n ≤ 1000), the number of elements in the array.
- The second line contains **n** space-separated integers.

Output:

- Print the sorted array.

Example:

Input:

5

38 27 43 3 9

Output:

3 9 27 38 43

5. Backtracking

Question: N-Queens Problem

The N-Queens puzzle is the problem of placing N chess queens on an N×N chessboard so that no two queens threaten each other. Write a program to solve the N-Queens problem.

Input:

- The first line contains an integer **n** ($1 \leq n \leq 10$), the number of queens.

Output:

- Print all distinct solutions to the N-Queens puzzle. Each solution should be represented by a list of strings, where each string represents a row of the chessboard.

Example:

Input:

4

Output:

.Q..

...Q

Q...

..Q.

..Q.

Q...

...Q

.Q..

6. String Manipulation

Question: Anagram Check

Given two strings, determine if they are anagrams of each other. An anagram is a word or phrase formed by rearranging the letters of a different word or phrase.

Input:

- The first line contains the first string **s1** ($1 \leq |s1| \leq 100$).
- The second line contains the second string **s2** ($1 \leq |s2| \leq 100$).

Output:

- Print "YES" if the strings are anagrams, otherwise print "NO".

Example:

Input:

listen

silent

Output:

YES

7. Searching Algorithm (continued)

Question: Binary Search

Implement a binary search algorithm to find the index of a target value in a sorted array. If the target is not found, return -1.

Input:

- The first line contains an integer **n** ($1 \leq n \leq 1000$), the number of elements in the array.
- The second line contains **n** space-separated integers (sorted in ascending order).
- The third line contains the target integer **target**.

Output:

- Print the index of the target value if found, otherwise print **-1**.

Example:

Input:

5

1 2 3 4 5

3

Output:

2

8. Hashing

Question: Two Sum

Given an array of integers, return the indices of the two numbers such that they add up to a specific target.

Input:

- The first line contains an integer **n** (1 ≤ n ≤ 10^4), the number of elements in the array.
- The second line contains **n** space-separated integers.
- The third line contains the target integer **target**.

Output:

- Print the indices of the two numbers (0-based index). If no such indices exist, print **-1**.

Example:

Input:

4

2 7 11 15

9

Output:

0 1

9. Bit Manipulation

Question: Single Number

Given a non-empty array of integers, every element appears twice except for one. Find that single one.

Input:

- The first line contains an integer **n** (1 ≤ n ≤ 1000), the number of elements in the array.
- The second line contains **n** space-separated integers.

Output:

- Print the single number.

Input:

5

4 1 2 1 2

Output:

4

10. Recursion

Question: Factorial Calculation

Write a recursive function to calculate the factorial of a given non-negative integer.

Input:

- The first line contains a single integer **n** ($0 \leq n \leq 20$).

Output:

- Print the factorial of **n**.

Example:

Input:

5

Output:

120

11. Sliding Window

Question: Maximum Sum Subarray of Size K

Given an array of integers and a number **k**, find the maximum sum of a subarray of size **k**.

Input:

- The first line contains two integers **n** ($1 \leq n \leq 1000$) and **k** ($1 \leq k \leq n$).
- The second line contains **n** space-separated integers.

Output:

- Print the maximum sum of a subarray of size **k**.

Example:

Input:

5 3

2 1 5 1 3

Output:

9

12. Tree Traversal

Question: Binary Tree Inorder Traversal

Given a binary tree, return the inorder traversal of its nodes' values.

Input:

- The input will be given as a series of integers representing the level order traversal of the binary tree, where **-1** represents a null node.

Output:

- Print the inorder traversal of the binary tree as space-separated integers.

Example:

Input:

1 2 3 -1 -1 4 5 -1 -1 -1

Output:

2 1 4 3 5

13. Dynamic Programming (Advanced)

Question: Coin Change Problem

You are given an integer array **coins** representing coins of different denominations and an integer **amount** representing a total amount of money. You want to compute the number of combinations that make up that amount.

Input:

- The first line contains an integer **n** ($1 \leq n \leq 100$), the number of coin denominations.
- The second line contains **n** space-separated integers representing the coin denominations ($1 \leq$ denomination ≤ 5000).
- The third line contains an integer **amount** ($1 \leq$ amount ≤ 5000).

Output:

- Print the number of combinations that make up the amount.

Example:

Input:

3

1 2 5

5

Output:

4