Problems on Dynamic Programming
Problem 1 -- Minimum cost

Write a program To Calculate the minimum cost to reach the destination city from the source city

Given a N*N matrix where each cell of the matrix(i,j) indicated the cost of the direct flight from city i to city j.Find the minimum cost to reach the destination city N-1 from the source city 0.

Note that the entries(j<i) in the matrix are irrelevant.

Input and Output Format:

Refer sample input and output for formatting specifications.
All text in bold corresponds to input and the rest corresponds to output.

Sample Input :
Enter the value of n
3
Enter the costs to reach from one city to another city
0 20 30
20 0 15
30 15 0

Sample Output:
The minimum cost is 30

SNo	Name	Input	Output
1	TC 4	3 0 25 15 25 0 20 15 20 0	Enter the value of n Enter the costs to reach from one city to another city The minimum cost is 15
2	TC 3	4 0 35 55 70 35 0 15 20 55 15 0 10 70 20 10 0	Enter the value of n Enter the costs to reach from one city to another city The minimum cost is 55
3	TC 2	4 0 20 30 100 20 0 15 75 30 15 0 50 100 75 50 0	Enter the value of n Enter the costs to reach from one city to another city The minimum cost is 80

SNo	Name	Input	Output
4	TC 1	3 0 20 30 20 0 15 30 15 0	Enter the value of n Enter the costs to reach from one city to another city The minimum cost is 30
5	TC 5	3 0 20 35 20 0 10 35 10 0	Enter the value of n Enter the costs to reach from one city to another city The minimum cost is 30

Problem 2 - Tiles game

Write a program to find number of ways to fill a n x 4 matrix with 1 x 4 tiles

Given a n*4 matrix where n is a positive number, find number of ways to fill the matrix with 1*4 tiles

For example,there are 3 ways to place 1*4 tiles in a 5*4 matrix -

1.Place all 5 tiles horizontally.
2.Place 4 tiles vertically in the first four rows and 1 tile horizontally in the last row.
3.Place 1 tile horizontally in the first row and adjust 4 tiles vertically in the remaining rows.

similarly,there are 4 ways to place 1*4 tiles in a 6*4 matrix -

1.Place all 6 tiles horizontally.
2.Place 4 tiles vertically in the first four rows and remaining 2 tiles horizontally in the last two rows.
3.Place 2 tiles horizontally in the first two rows and adjust the remaining 4 tiles vertically in the remaining rows.
4.Place 2 tiles horizontally in the first & the last row and adjust the remaining 4 tiles vertically in the middle rows.

Input and Output Format:

Refer sample input and output for formatting specifications.
All text in bold corresponds to input and the rest corresponds to output.

Sample Input:

Enter the value of n
5

Sample Output:

Total number of ways are 3

Test Cases

SNo	Name	Input	Output
1	TC 3	3	Enter the value of n Total number of ways are 1
2	TC 4	4	Enter the value of n Total number of ways are 2
3	TC 5	9	Enter the value of n Total number of ways are 10
4	TC 1	6	Enter the value of n Total number of ways are 4
5	TC 2	5	Enter the value of n Total number of ways are 3

Problem 3 - Collect Coins

Write a program to Collect maximum value of coins in a matrix

Given a M x N matrix where each cell contains a coin of some denomination, collect maximum value of coins by traversing the grid.

The first traversal starts from the top-left corner of the matrix and end at the bottom-left corner and the second traversal starts from the top-right corner and end at the bottom-right corner. From any cell (i, j) in the matrix, we are allowed to move to cell (i+1, j+1) or (i+1, j-1) or (i+1, j) . If both traversals passes through a same cell, only one can collect coin of that cell.

Input and Output Format:

Refer sample input and output for formatting specifications.
All text in bold corresponds to input and the rest corresponds to output.

Sample Input :

Enter the value of m
4
Enter the value of n
5
Enter the value of coins
2 3 4 5 1

3 4 5 1 2

5 6 7 8 9

2 4 3 2 1

Sample output:

The maximum coins collected is 30

Test Cases

SNo	Name	Input	Output
1	TC 4	4 5 1 2 3 6 5 6 7 8 9 5 1 2 5 6 9 5 3 6 4 7	Enter the value of m Enter the value of n The maximum coins collected is 45
2	TC 5	4 4 1 5 9 3 2 6 5 4 1 2 3 69 3 2 6 5	Enter the value of m Enter the value of n The maximum coins collected is 94
3	TC 3	4 5 2 3 4 5 1 3 4 5 1 2 5 6 7 8 9 5 4 3 2 1	Enter the value of m Enter the value of n The maximum coins collected is 30
4	TC 2	5 4 0 2 4 1 4 8 3 7 2 3 6 2 9 7 8 3 1 5 9 4	Enter the value of m Enter the value of n The maximum coins collected is 47
5	TC 1	4 4 0 2 4 1 4 8 3 7 2 3 6 2 9 7 8 3	Enter the value of m Enter the value of n The maximum coins collected is 37

Problem 4 - Count of paths

Count all paths in a matrix from first cell to last cell

Given a rectangular grid, efficiently count all paths starting from the first cell to the last cell in the grid. We can either move down, or move towards right from a cell.

The idea is to start from the top-left corner of the matrix and recur for the next cell which can be either immediate right cell or immediate bottom cell.

Input and Output Format:

Refer sample input and output for formatting specifications.
All text in bold corresponds to input and the rest corresponds to output.

Sample Input :

Enter the value of m
3
Enter the value of n
3

Sample Output :

Total number of paths are: 6

Test Cases

SNo	Name	Input	Output
1	TC 1	3 3	Enter the value of m Enter the value of n Total number of paths are: 6
2	TC 5	6 5	Enter the value of m Enter the value of n Total number of paths are: 126
3	TC 4	4 4	Enter the value of m Enter the value of n Total number of paths are: 20

SNo	Name	Input	Output
4	TC 3	5 4	Enter the value of m Enter the value of n Total number of paths are: 35
5	TC 2	4 5	Enter the value of m Enter the value of n Total number of paths are: 35

Problem 5 - Longest path

Write a program to Find length of longest path in the matrix with consecutive characters

Given a M x N matrix of characters, find the length of longest path in the matrix starting from a given character. All characters in the longest path should be increasing and consecutive to each other in alphabetical order.

We are allowed to search the string in all eight possible directions i.e. North, West, South, East, North-East, North-West, South-East, South-West.

Input and Output Format:

Refer sample input and output for formatting specifications.
All text in bold corresponds to input and the rest corresponds to output.

Sample Input and Output :

Enter the row size of matrix

4

Enter the cloumn size of matrix

4

Enter elements of matrix

a b c d

e f g h

i j k l

m n o p

Enter the starting character
a
The length of longest path with consecutive characters starting from character a is 4

Test Cases

SNo	Input	Output
1	3 4 p q r s l k m j k i o p p	Enter the row size of matrix Enter the column size of matrix Enter the elements of matrix Enter the starting character The length of longest path with consecutive characters starting from character p is 4
2	4 4 a b c d b d e r p o i u m l k j a	Enter the row size of matrix Enter the column size of matrix Enter the elements of matrix Enter the starting character The length of longest path with consecutive characters starting from character a is 5
3	4 4 a b c d e f g h i j k l m n o p a	Enter the row size of matrix Enter the column size of matrix Enter the elements of matrix Enter the starting character The length of longest path with consecutive characters starting from character a is 4
4	3 3 i d c h e b g f a a	Enter the row size of matrix Enter the column size of matrix Enter the elements of matrix Enter the starting character The length of longest path with consecutive characters starting from character a is 9
5	3 3 h i j d f e a b c a	Enter the row size of matrix Enter the column size of matrix Enter the elements of matrix Enter the starting character The length of longest path with consecutive characters starting from character a is 3

Problem 6 -- String wildcard pattern

Given a string and a pattern containing wildcard characters, write an efficient algorithm to check if the input string matches with the wildcard pattern or not. wildcard character '?' can match to any character in the input string and the '*' wildcard character can match to zero or more characters in the input string.

The time complexity of recursive solution is exponential. We can use Dynamic Programming to bring down the time complexity to O(m*n) using O(m*n) extra space.

Apply Dynamic Programming solution to solve the given problem using memoization

Input Format:

Input consist of two strings.

Output Format:

The output should be "ture" if input string matches with the wildcard pattern print "false" otherwise.

Refer sample input and output for formatting specifications.
[All text in bold corresponds to input and the rest corresponds to output]

Sample Input and Output 1:

XYXZZXY

X*Y**

true

Sample Input and Output 2:

XYXZZXY

XX?
false

Sample Input and Output 3:

XYXZZXY

X*ZZ??
true

Test Cases

SNo	Name	Input	Output
1	T6	XYXZZXY X***X	false
2	T5	XKXDDXK *X*X?	true
3	T3	XYXZZXY X*ZZ??	true
4	T4	XYXZZXY *	true
5	T1	XYXZZXY X***Y	true

SNo	Name	Input	Output
6	T2	XYXZZXY XX?	false

Problem 7 - Pots of Gold

In Pots of gold game, there are two players A & B and there are 'n' pots of gold arranged in a line, each containing some gold coins. The players can see how many coins are there in each gold pot and each player gets alternating turns in which the player can pick a pot from one of the ends of the line. The winner is the player which has a higher number of coins at the end. Write a program to “maximize” the number of coins collected by A, assuming B also plays optimally and A starts the game.

Input and Output Format:

The first line of input consists of an integer 'n' which corresponds to the number of pots.

Next n integer inputs correspond to the number of gold coins in the pots.

The output should display the coins collected by player A and B respectively.

Refer sample input and output for formatting specifications.

[All text in bold corresponds to input and the rest corresponds to output]

Sample Input 1:

4

4 6 2 3

Sample Output 1:

9 6

Sample Input 2:

6

6 1 4 9 8 5

Sample Output 2:

18 15

Test Cases

SNo	Input	Output
1	4 2 5 3 6	11 5
2	4 4 6 2 3	9 6
3	8 4 5 2 8 4 1 9 3	19 17
4	6 6 1 4 9 8 5	18 15

```
#include<stdio.h>
int main()
{
        int n,a[100],i,A=0,B=0;
        scanf("%d",&n);
        for(i=0;i<n;i++)
        {
            scanf("%d",&a[i]);
        }
        for(i=0;i<n;i++)
        {
            if(i%2==0)
            A=A+a[i];
            else
            B=B+a[i];
        }
        if(A<B)
        printf("%d %d",B,A);
        else
        printf("%d %d",A,B);
        return 0;
}
```

Problem 8 - Maximum Profit of Shares

Given a list containing a future prediction of share prices, write a program to find maximum profit that can be earned by buying and selling shares at most twice with a constraint that a new transaction can only start after the previous transaction is complete. i.e. we can only hold at most one share at a time.

Input Format:

The first input consists of an integer which corresponds to the number of shares to buy and sell.

The next n inputs are the share prices.

Output Format:

The output should display maximum profit that can be earned by buying and selling shares.

Refer sample input and output for formatting specifications.
[All text in bold corresponds to input and the rest corresponds to output]

Sample Input and Output 1:

7

2 4 7 5 4 3 5

The maximum profit is 7

Sample Input and Output 2:

4

8 7 6 4

The maximum profit is 0

Test Cases

SNo	Name	Input	Output
1	tc3	5 10 6 8 3	The maximum profit is 2
2	tc2	4 8 7 6 4	The maximum profit is 0
3	tc1	7 2 4 7 5 4 3 5	The maximum profit is 7
4	tc5	18 2 5 4 7 5 2 1 6 7 12 2 3 4 5 6 7 8 9	The maximum profit is 18
5	tc4	8 2 5 4 7 5 2 1 6	The maximum profit is 10

Problem 9 -- Implement Diff Utility

Write a program to implement your own diff utility. i.e given two similar strings, efficiently list out all differences between them.
The diff utility is a data comparison tool that calculates and displays the differences between two text. It tries to determine the smallest set of deletions and insertions to create one text from the other. Diff is line-oriented rather than character-oriented.

Use the following notations to display the difference between the two strings.

- If a character is absent in the second string but present in the first string, it must have been deleted (indicated by the '–' marks).
- If it is absent in the first string but present in the second string, it must have been inserted (indicated by the '+' marks).

Input Format:

Inputs for this problem are two strings.

Output Format:

The output should display the differences between the two strings.

Refer sample input and output for formatting specifications.
[All text in bold corresponds to input and the rest corresponds to output]

Sample Input and Output 1:

XMJYAUZ

XMJAATZ

X M J -Y A -U +A +T Z

Sample Input and Output 2:

ABCDFGHJQZ

ABCDEFGIJKRXYZ

A B C D +E F G -H +I J -Q +K +R +X +Y Z

Test Cases

SNo	Name	Input	Output
1	TC3	MAXIMUM MINIMUM	M -A -X +I +N I M U M
2	TC4	WEATHER WHETHER	W +H E -A T H E R

SNo	Name	Input	Output
3	tc2	ABCDFGHJQZ ABCDEFGIJKRXYZ	A B C D +E F G -H +I J -Q +K +R +X +Y Z
4	tc1	XMJYAUZ XMJAATZ	X M J -Y A -U +A +T Z
5	tc5	WEATHER weather	-W -E -A -T -H -E -R +w +e +a +t + h +e +r

Problem 10 - Maximum Sum of a subsequence with no Adjacent Elements

Given an array of integers, write a program to find the maximum sum of a subsequence of the given array where subsequence contains no adjacent elements.

Hint: Consider the current element only if it is not adjacent to the previous element considered.

Input Format:

The first input consists of an integer which corresponds to the number of elements present in the single dimension array.
The next n inputs are the elements in the array.

Output Format:

The output should display the sum of a subsequence of the given array where subsequence contains no adjacent elements.

Refer sample input and output for formatting specifications.
[All text in bold corresponds to input and the rest corresponds to output]

Sample Input and Output 1:

9

1 2 9 4 5 0 4 11 6

Maximum sum is 26

SNo	Name	Input	Output
1	tc2	9 1 2 9 4 5 0 4 11 6	Maximum sum is 26
2	tc1	9 3 2 8 6 5 7 8 11 12	Maximum sum is 36
3	tc4	10 1 2 3 4 5 6 7 8 9 10	Maximum sum is 30
4	tc3	5 3 2 1 6 15	Maximum sum is 19