

**Christmas Tree Pattern**

Write a program that takes an integer as input and prints a Christmas tree pattern with the given number of levels. For example, if the input is 5, the program should print a Christmas tree like this:



**Hollow Diamond Pattern**

Write a program that takes an odd integer as input and prints a hollow diamond pattern with the given number of rows. For example, if the input is 7, the program should print a pattern like this:



**Pyramid Pattern**

Write a program that takes an integer as input and prints a pyramid pattern with the given number of levels. For example, if the input is 4, the program should print a pyramid like this:



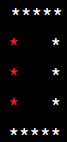
**Inverted Pyramid Pattern**

Write a program that takes an integer as input and prints an inverted pyramid pattern with the given number of levels. For example, if the input is 3, the program should print an inverted pyramid like this:



**Square Patter**n

Write a program that takes an integer as input and prints a square pattern with the given number of rows and columns. For example, if the input is 5, the program should print a square like this:



1. \*\*Find Maximum Element:\*\*

Given a 1D array, write a program to find the maximum element in the array.

2. \*\*Find Minimum Element:\*\*

Write a program to find the minimum element in a 1D array.

3. \*\*Sum of Elements:\*\*

Calculate the sum of all elements in a 1D array.

4. \*\*Find the Second Largest Element:\*\*

Write a program to find the second largest element in a 1D array.

5. \*\*Reverse the Array:\*\*

Given a 1D array, write a program to reverse the elements in the array. For example, if you have `[1, 2, 3, 4, 5]`, the program should transform it into `[5, 4, 3, 2, 1]`.

6. \*\*Remove Duplicates:\*\*

Write a program to remove duplicates from a 1D array and return a new array with unique elements.

7. \*\*Count Occurrences:\*\*

Given an element, write a program to count the number of times it occurs in a 1D array.

8. \*\*Sort the Array:\*\*

Implement a sorting algorithm (e.g., bubble sort, insertion sort) to sort the elements in a 1D array in ascending or descending order.

9. \*\*Find Subarray with Maximum Sum:\*\*

Write a program to find the subarray (contiguous elements) with the maximum sum in a 1D array. This is also known as the "maximum subarray problem."

10. \*\*Find the Missing Number:\*\*

Given an array containing n distinct numbers taken from 0, 1, 2, ..., n, write a program to find the missing number in the sequence.

1. \*\*Print Horizontal Rows:\*\*

Given a 2D matrix (a list of lists), write a program to print all the horizontal rows. For example, if you have the following 2D matrix:

```

1 2 3

4 5 6

7 8 9

```

The program should print:

```

1 2 3

4 5 6

7 8 9

```

2. \*\*Print Vertical Columns:\*\*

Write a program to print all the vertical columns of a given 2D matrix. Using the same matrix as above, the program should print:

```

1 4 7

2 5 8

3 6 9

```

3. \*\*Find Maximum Element:\*\*

Given a 2D matrix, write a program to find the maximum element in the matrix.

4. \*\*Sum of Elements:\*\*

Write a program to calculate the sum of all elements in a 2D matrix.

5. \*\*Transpose a Matrix:\*\*

Write a program to transpose a given 2D matrix. This means swapping rows and columns. For example, the transpose of the matrix given above would be:

```

1 4 7

2 5 8

3 6 9

```

6. \*\*Rotate a Matrix 90 Degrees:\*\*

Given a 2D matrix, write a program to rotate it 90 degrees clockwise. For example, if you have the following matrix:

```

1 2 3

4 5 6

7 8 9

```

The program should rotate it to:

```

7 4 1

8 5 2

9 6 3

```

7. \*\*Diagonal Elements:\*\*

Write a program to print the diagonal elements of a 2D matrix. For the matrix given above, the program should print: `1 5 9`.