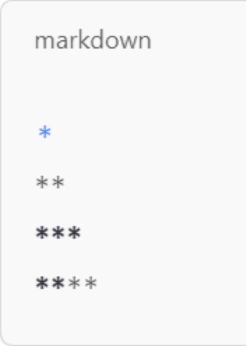
Lists and Tuples

1. Write a program to create a list of numbers, remove the last element, and then add a new element to the list.
2. Given a nested list [[1, 2, 3], [4, 5, 6], [7, 8, 9]], retrieve the element 8.
3. Demonstrate the difference between a shallow and deep copy of a list.
4. Create a tuple of integers. Print the last two elements and the first element in reverse order.

Loops and Patterns

1. Use a while loop to print numbers from 1 to 10, but skip the number 5.
2. Write a program to print the following star pattern:



```
*
**
***
****
```

3. Using a for loop, print all even numbers between 1 and 20.
4. Calculate the sum of all numbers from 1 to 100 using a loop.

Dictionaries

1. Create a dictionary with student details like name, age, and grade. Add a new key-value pair for the student's city.
2. Given a nested dictionary, access the value of a nested key (for example, in {'student': {'name': 'Alice', 'marks': {'math': 80}}}, print the value of 'math').
3. Using a dictionary of fruit prices, update the price of an existing fruit and add a new fruit to the dictionary.

Functions

1. Define a function that accepts two numbers and returns their product.
2. Create a function that takes an arbitrary number of arguments and returns their sum.
3. Use a lambda function to calculate the square of a number and test it with a few examples.

Lambdas, Map, Filter, and Reduce

1. Use a lambda function with map() to double each number in a list [1, 2, 3, 4, 5].
2. Write a program using filter() to filter out only positive numbers from a list.
3. Using reduce(), find the product of all numbers in a list [1, 2, 3, 4].
4. Create a lambda function to add two numbers. Use it with reduce() to sum all numbers in a list [10, 20, 30, 40].