1. You have a list containing mixed data types represented as strings. Write a Python program using a filter() and lambda function to keep only the alphabetic items (letters only) and remove any numeric strings from the list."
   items = ['sql', '123', 'python']
2. You are managing an e-commerce inventory database. Given a list of product dictionaries, write a Python program using a filter() and lambda function to extract and display only the products that are currently in stock (instock: True)
   products = [
   {'id': 1, 'name': 'laptop', 'category': 'electronics', 'price': 1200, 'instock': True},
   {'id': 2, 'name': 'smartphone', 'category': 'electronics', 'price': 800, 'instock': False}
   ]

3. Create a simple calculator program using Python functions.

   Requirements:

   Create four separate functions: add(), subtract(), multiply(), and divide()
   Each function should take two numbers as parameters and return the result
   Create a main calculator() function that:

   - Displays a menu with options (Add, Subtract, Multiply, Divide, Exit)
   - Takes user input to select an operation
   - Gets two numbers from the user
   - Calls the appropriate function based on user choice
   - Displays the result
   - Continues until the user chooses to exit

4. Create a function, remove_at_idx, with the following features:

   - Takes a list as its first argument.
   - Takes a positive index as its second argument.
   - Removes the element at the given index and decreases the index of all subsequent elements by one.
   - Returns a new list.
   -

5. Write a Python program that simulates a voting eligibility check using Exception Handling. Create a function named vote that takes an integer representing a person's age as an argument. If the age is less than 18, the function should raise a ValueError with the message: 'not eligible: Must be 18 or older'. If the age is 18 or older, it should print 'Can vote'. Call the function inside a try block with a test age. Use except block to catch the ValueError and print the error message to the console so the program doesn't crash.
6. Develop a Python script to create a robust backup of a text file. The program must transfer data from a source file (a.txt) to a destination file (b.txt) using best practices for memory efficiency and resource safety.

- Use a single with statement to manage both the input (read) and output (write) file streams. This ensures that both files are properly closed by the system, regardless of whether the operation succeeds or fails.
- To prevent system strain when handling large datasets, do not load the entire file into RAM. Instead, implement a loop that processes the file line-by-line.
- For every line extracted from the source, immediately commit it to the destination file to maintain data integrity.

7. Write a program that counts how many lines exist in a specific file. Requirements:

    - Open story.txt in read mode.
    - Use a counter variable to keep track of the lines.
    - Print the total count at the end.
    - Real-life use: Log analysis where you need to know how many events occurred in a day.

8. Read a file numbers.txt containing one number per line, calculate the square of each number, and write the results to squared.txt. Requirements:

    - Remember to convert the string data from the file into an int() before calculating.
    - Convert the result back to a str() before writing, as files only accept strings.
    - Real-life use: Processing sensor data or performing batch calculations on a list of prices.

9. Write a program that reads input.txt and writes a version of it to output.txt where every single letter is capitalized. Requirements:

    - Use the .upper() string method on each line.
    - Ensure the formatting (line breaks) remains the same.
    - Real-life use: Standardizing data inputs (like converting all names to uppercase for a database).

10. Write any 4 difference between compiler and interpreter.
11. Write any 4 difference between high level language and low level language

12. Create a Python-based utility to normalize user-generated text. To ensure the messaging application can process data without errors, you must neutralize any characters that are neither letters nor numbers by converting them into a standardized placeholder.

    - The program should prompt the user to provide a raw text string.
    - Iterate through the input and identify "special characters" (any character that is not a letter or a digit).
    - Transform every identified special character into a # symbol.
    - Display the final, "cleaned" version of the string to the console.

13. Examine a collection of data points stored in a list called values. The goal is to identify the single element that appears more often than any other, store that specific value in a variable named most_frequent, and output it to the console.

    - Systematically track the number of occurrences for every unique item in the list.
    - Compare the counts to determine which item holds the "maximum" frequency.
    - Capture the winner in the designated variable and print it.

14. Develop a reusable function named count that scans a collection of data to determine how many times a specific "target" element appears.
15. Is python compiled or interpreted programming language.
16. Explain implicit and explicit type conversion
17. Explain issubset and issuperset method with example
18. Explain pop and popitem method with example
19. Explain append and insert method with example
20. Write a Python program to sum all the items in a list.
21. Write a Python program to get the largest number from a list.
22. Write a Python program to find the list of words that are longer than n from a given list of words.
23. Write a Python program to print a specified list after removing the 0th, 4th and 5th elements.
    Sample List : ['Red', 'Green', 'White', 'Black', 'Pink', 'Yellow']
    Expected Output : ['Green', 'White', 'Black']
24. Write a Python program to generate and print a list of the first and last 5 elements where the values are square numbers between 1 and 30 (both included).
25. Write a Python program to check if each number is prime in a given list of numbers. Return True if all numbers are prime otherwise False.
    Sample Data:
    ([0, 3, 4, 7, 9]) -> False
    ([3, 5, 7, 13]) -> True
    ([1, 5, 3]) -> False
26. Write a Python program to convert a list of characters into a string.
27. Explain about LEGB rules
28. Explain local variables and global variables
29. Explain variable length argument and keyword variable length argument
30. Write a Python program to get the frequency of elements in a list.
31. Write a Python program to select the odd items from a list.
32. Write a Python program to convert a list to a list of dictionaries.
    Sample lists: ["Black", "Red", "Maroon", "Yellow"], ["#000000", "#FF0000", "#800000", "#FFFF00"]
    Expected Output: [{'color_name': 'Black', 'color_code': '#000000'}, {'color_name': 'Red', 'color_code': '#FF0000'}, {'color_name': 'Maroon', 'color_code': '#800000'}, {'color_name': 'Yellow', 'color_code': '#FFFF00'}]

33. Write a Python program to calculate the length of a string.
34. Write a Python function to sum all the numbers in a list.
    Sample List : (8, 2, 3, 0, 7)
    Expected Output : 20
35. Write a Python program to reverse a string.

    Sample String : "1234abcd"
    Expected Output : "dcba4321"

36. Write a Python function to calculate the factorial of a number (a non-negative integer). The function accepts the number as an argument.
37. Write a Python function that accepts a string and counts the number of upper and lower case letters.
    Sample String : 'The quick Brow Fox'
    Expected Output :
    No. of Upper case characters : 3
    No. of Lower case Characters : 12
38. Write a Python program to print the even numbers from a given list.

    Sample List : [1, 2, 3, 4, 5, 6, 7, 8, 9]
    Expected Result : [2, 4, 6, 8]

39. Write a Python script to sort (ascending and descending) a dictionary by value.
40. Write a Python script to concatenate the following dictionaries to create a new one.
    Sample Dictionary :
    dic1={1:10, 2:20}
    dic2={3:30, 4:40}
    dic3={5:50,6:60}
    Expected Result : {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
41. Write a Python script to check whether a given key already exists in a dictionary.
42. Write a Python script to print a dictionary where the keys are numbers between 1 and 15 (both included) and the values are the square of the keys.
43. Sample Dictionary
    {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121, 12: 144, 13: 169, 14: 196, 15: 225}
44. Write a Python program to multiply all the items in a dictionary.
45. Write a Python program to print all distinct values in a dictionary.

Sample Data : [{"V":"S001"}, {"V": "S002"}, {"VI": "S001"}, {"VI": "S005"}, {"VII":"S005"}, {"V":"S009"},{"VIII":"S007"}]

Expected Output : Unique Values: {'S005', 'S002', 'S007', 'S001', 'S009'}

46. Write a function named is_palindrome with one argument: dna: A string representing the DNA sequence Implement the is_palindrome () function so that it returns a Boolean value saying whether or not the given DNA sequence is a palindrome. Test your function by printing the result on the provided inputs. Your function will also be checked by being executed on a set of hidden inputs.

47. Create a function, count, with the following features:
    - Takes in a list as its first argument.
    - The second argument is a potential element of the first.
    - Counts how many times the second argument occurs in the first and returns the value.

48. Explain the term indentation and why it is used to organize information. Finding Shared Tags Create a tool that compares the labels two different stores use for their products. Use set math to find exactly which labels appear in both stores so you can suggest similar items to customers.