📋 Lists

🔰 Beginner

1. **Create a List**
   - Create a list containing the numbers from 1 to 10.
2. **Access Elements**
   - Access the third element in a given list.
3. **List Length**
   - Write a program to find the length of a list.
4. **Append Element**
   - Add an element to the end of a list.
5. **Insert Element**
   - Insert an element at a specific position in a list.
6. **Remove Element**
   - Remove the first occurrence of a specified element from a list.
7. **List Slicing**
   - Slice a list to obtain the first five elements.
8. **List Concatenation**
   - Concatenate two lists.
9. **List Replication**
   - Replicate a list three times.
10. **Check Membership**
    - Check if an element exists in a list.

🟢 Intermediate

11. **List Comprehension**
    - Generate a list of squares for numbers from 1 to 20 using list comprehension.
12. **Find Maximum**
    - Find the maximum element in a list without using the `max()` function.
13. **List Reversal**
    - Reverse a list without using the `reverse()` method.
14. **Remove Duplicates**
    - Remove duplicate elements from a list.
15. **Sort List**
    - Sort a list of integers in ascending and descending order.
16. **Sum of Elements**
    - Calculate the sum of all elements in a list.
17. **Product of Elements**
    - Calculate the product of all elements in a list.
18. **Find Index**
    - Find the index of a specified element in a list.
19. **Count Occurrences**
    - Count how many times a specific element appears in a list.
20. **Merge Lists**
    - Merge two lists into a single list without duplicates.

🚀 Advanced

21. **Nested Lists**
    - Flatten a nested list (a list of lists) into a single list.
22. **List Filtering**
    - Filter out all odd numbers from a list using lambda and filter.
23. **List Mapping**
    - Apply a function to all elements in a list using `map()`.
24. **List Sorting with Key**
    - Sort a list of tuples based on the second element.
25. **List Intersection**
    - Find the intersection of two lists.
26. **List Difference**
    - Find the difference between two lists.
27. **List Union**
    - Find the union of two lists.
28. **List of Dictionaries**
    - Sort a list of dictionaries based on a specific key.
29. **Remove Specific Elements**
    - Remove all elements that satisfy a condition (e.g., remove all negative numbers).
30. **List to String**
    - Convert a list of characters into a single string.

🔷 Tuples

🔰 Beginner

31. **Create a Tuple**
    - Create a tuple containing the names of five fruits.
32. **Access Elements**
    - Access the second element in a tuple.
33. **Tuple Length**
    - Find the length of a tuple.
34. **Tuple Concatenation**
    - Concatenate two tuples.
35. **Tuple Replication**
    - Replicate a tuple two times.
36. **Check Membership**
    - Check if an element exists in a tuple.
37. **Tuple Slicing**
    - Slice a tuple to obtain the last three elements.
38. **Convert List to Tuple**
    - Convert a given list into a tuple.
39. **Tuple to List**
    - Convert a tuple into a list.
40. **Delete Tuple**
    - Delete an entire tuple.

🟢 Intermediate

41. **Find Maximum**
    - Find the maximum element in a tuple.
42. **Find Minimum**

    - Find the minimum element in a tuple.
43. **Count Occurrences**
    - Count how many times a specific element appears in a tuple.
44. **Index of Element**
    - Find the index of a specified element in a tuple.
45. **Tuple Packing and Unpacking**
    - Demonstrate tuple packing and unpacking with multiple variables.
46. **Immutable Nature**
    - Attempt to modify a tuple and handle the exception.
47. **Nested Tuples**
    - Access elements in a nested tuple structure.
48. **Tuple Iteration**
    - Iterate through a tuple and print each element.
49. **Slice and Concatenate**
    - Slice a tuple and concatenate it with another tuple.
50. **Tuple Comprehension**
    - Explain why tuple comprehension is not possible and provide an alternative using generator expressions.

🚀 Advanced

51. **Sort Tuple**
    - Sort a tuple without converting it to a list.
52. **Tuple with Mixed Data Types**

- Create a tuple with different data types and access each type.

53. **Tuple Multiplication**
    - Multiply a tuple by a number and explain the result.
54. **Comparing Tuples**
    - Compare two tuples lexicographically.
55. **Tuple Methods**
    - Demonstrate the use of `count()` and `index()` methods with tuples.
56. **Tuple as Dictionary Keys**
    - Use tuples as keys in a dictionary and explain why it's possible.
57. **Tuple from User Input**
    - Create a tuple from user input containing multiple values.
58. **Zip Function with Tuples**
    - Use the `zip()` function to combine two lists into a tuple of pairs.
59. **Unzipping Tuples**
    - Unzip a list of tuples into separate lists.
60. **Convert Nested Lists to Tuples**
    - Convert a nested list structure into nested tuples.

🟠 **Sets**

🔰 **Beginner**

61. **Create a Set**
    - Create a set containing unique vowels.
62. **Set Length**
    - Find the number of elements in a set.
63. **Add Element**
    - Add an element to a set.
64. **Remove Element**
    - Remove a specific element from a set.
65. **Set Union**
    - Perform a union of two sets.
66. **Set Intersection**
    - Perform an intersection of two sets.
67. **Set Difference**
    - Find the difference between two sets.
68. **Check Subset**
    - Check if one set is a subset of another.
69. **Check Superset**
    - Check if one set is a superset of another.
70. **Clear Set**
    - Remove all elements from a set.

🟢 Intermediate

71. **Set Comprehension**
    - Create a set of squares for numbers from 1 to 10 using set comprehension.
72. **Find Unique Elements**
    - Find unique elements in a list using a set.

73. **Set Symmetric Difference**
    - Find the symmetric difference between two sets.
74. **Frozen Set**
    - Explain and create a frozen set.
75. **Set Operations with Strings**
    - Find common characters between two strings using sets.
76. **Disjoint Sets**
    - Check if two sets are disjoint.
77. **Power Set**
    - Generate the power set of a given set.
78. **Set Update**
    - Update a set with elements from another iterable.
79. **Remove Duplicates from List**
    - Remove duplicates from a list while preserving order using sets.
80. **Set to List**
    - Convert a set into a list.

🚀 Advanced

81. **Subset Generation**
    - Generate all possible subsets of a given set.
82. **Find Missing Elements**
    - Find elements that are in one set but not in another.
83. **Set Operations in Practical Scenarios**
    - Use set operations to solve a real-world problem (e.g., common friends in social networks).

84. **Immutable Sets**
    - Explain the difference between sets and frozensets and provide examples.
85. **Set Intersection Update**
    - Update a set to keep only elements found in another set.
86. **Find Unique Words in a Text**
    - Find all unique words in a given paragraph.
87. **Set Combinations**
    - Create all possible combinations of elements in a set of size two.
88. **Cartesian Product Using Sets**
    - Generate the Cartesian product of two sets.
89. **Set Membership Testing**
    - Efficiently test for membership of multiple elements in a set.
90. **Implementing Set Operations Manually**
    - Implement union, intersection, and difference without using built-in methods.

🟡 Dictionaries

🔰 Beginner

91. **Create a Dictionary**
    - Create a dictionary with keys as student names and values as their ages.

92. **Access Value**
    - Access the value associated with a specific key in a dictionary.
93. **Add Key-Value Pair**
    - Add a new key-value pair to a dictionary.
94. **Remove Key-Value Pair**
    - Remove a key-value pair from a dictionary.
95. **Dictionary Length**
    - Find the number of key-value pairs in a dictionary.
96. **Check Key Existence**
    - Check if a key exists in a dictionary.
97. **Iterate Through Dictionary**
    - Iterate through a dictionary and print keys and values.
98. **Dictionary Keys and Values**
    - Retrieve all keys and all values from a dictionary.
99. **Update Dictionary**
    - Update the value of an existing key in a dictionary.
100. **Merge Dictionaries** - Merge two dictionaries into one.

🟢 Intermediate

101. **Dictionary Comprehension**
    - Create a dictionary with numbers as keys and their squares as values using comprehension.
102. **Nested Dictionaries**
    - Create and access elements in a nested dictionary.

103. **Count Characters**
    - Count the frequency of each character in a string using a dictionary.
104. **Invert Dictionary**
    - Invert a dictionary so that keys become values and values become keys.
105. **Find Key with Maximum Value**
    - Find the key with the highest value in a dictionary.
106. **Merge Dictionaries with Overlapping Keys**
    - Merge two dictionaries and handle overlapping keys by summing their values.
107. **Dictionary from Two Lists**
    - Create a dictionary from two lists, one for keys and one for values.
108. **Filter Dictionary**
    - Create a new dictionary by filtering out key-value pairs based on a condition.
109. **Dictionary Sorting**
    - Sort a dictionary by keys and by values.
110. **Default Dictionary**
    - Use `defaultdict` to handle missing keys gracefully.

🚀 Advanced

111. **Deep Merge Dictionaries**
    - Merge two deeply nested dictionaries.
112. **Group By Key**

- Group a list of dictionaries by a specific key.

113. **Serialize Dictionary**
    - Serialize a dictionary to a JSON string and deserialize it back.
114. **Dictionary with Tuples as Keys**
    - Create a dictionary with tuple keys and access the values.
115. **Access Nested Dictionary Safely**
    - Access elements in a nested dictionary without raising `KeyError`.
116. **Dictionary Views**
    - Explain and demonstrate dictionary view objects (`keys()`, `values()`, `items()`).
117. **Implementing a Dictionary from Scratch**
    - Implement a simple dictionary data structure using lists.
118. **Top N Frequent Elements**
    - Find the top N most frequent elements in a list using a dictionary.
119. **Dictionary Pretty Print**
    - Pretty print a dictionary with indentation.
120. **Dictionary Intersection**
    - Find common keys in two dictionaries and merge their values.

🔄 Mixed Data Structures

🟢 Intermediate

121. **List of Tuples**
    - Create a list containing tuples of (name, age) and perform operations.
122. **Set of Tuples**
    - Create a set containing tuples and explain immutability.
123. **Dictionary of Lists**
    - Create a dictionary where each key maps to a list of values.
124. **Dictionary of Sets**
    - Create a dictionary where each key maps to a set of values.
125. **List of Dictionaries**
    - Create a list of dictionaries representing different employees and their details.
126. **Tuple of Dictionaries**
    - Create a tuple containing multiple dictionaries and access their elements.
127. **Convert List of Tuples to Dictionary**
    - Convert a list of key-value tuples into a dictionary.
128. **Nested Data Structures**
    - Create a complex data structure combining lists, tuples, sets, and dictionaries and perform operations.

129. **Merge Complex Structures**
    - Merge two complex data structures containing lists, tuples, sets, and dictionaries.
130. **Search in Nested Structures**
    - Implement a search function that looks for a value within nested lists, tuples, sets, and dictionaries.

🚀 Advanced

131. **Data Transformation**
    - Transform a list of dictionaries into a dictionary of lists.
132. **Deep Copy of Nested Structures**
    - Create a deep copy of a nested data structure without using the `copy` module.
133. **Flatten Nested Structures**
    - Flatten a deeply nested combination of lists, tuples, sets, and dictionaries into a single list.
134. **Dictionary Aggregation**
    - Aggregate data from multiple dictionaries into a single summary dictionary.
135. **Multi-level Sorting**
    - Sort a list of dictionaries based on multiple keys.
136. **Graph Representation**
    - Represent a graph using dictionaries and perform basic graph operations.
137. **Cache Implementation**

- Implement a simple caching mechanism using dictionaries and lists.

138. **Complex Query on Data Structures**
    - Perform complex queries on a mixed data structure, such as finding all unique values across lists within dictionaries.
139. **Serialization of Mixed Structures**
    - Serialize and deserialize a mixed data structure containing lists, tuples, sets, and dictionaries.
140. **Implement a Phonebook**
    - Implement a phonebook using nested dictionaries and lists, allowing multiple numbers per contact.

🎯 Additional Challenge Questions

141. **Find the Most Common Element in a List**
    - Use a dictionary to count occurrences and determine the most common element.
142. **Transpose a Matrix**
    - Represent a matrix as a list of lists and transpose it.
143. **Anagram Checker**
    - Check if two strings are anagrams using dictionaries.
144. **Word Frequency Counter**
    - Count the frequency of each word in a text file using dictionaries.
145. **Unique Elements Across Lists**

- Find unique elements that appear in all provided lists using sets.

146. **Matrix Multiplication**
    - Multiply two matrices represented as lists of lists.
147. **Remove Keys from Dictionary**
    - Remove keys from a dictionary based on a list of keys to remove.
148. **Find Missing Keys**
    - Identify keys that are present in one dictionary but missing in another.
149. **Sort List of Dictionaries by Multiple Keys**
    - Sort a list of dictionaries first by one key and then by another.
150. **Implementing a Multi-value Dictionary**
    - Create a dictionary where each key can have multiple values (e.g., using lists or sets).
    -