```python
# collections exercise
# https://codeshare.io/g8N9zN
# SaiPrabath Chowdary S

'''Exercise 1: List Operations
Create a list called `numbers` containing the numbers `1`, `2`, `3`, `4`,
and `5`.
Append the number `6` to the list. Remove the number `3` from the list.
Insert the number `0` at the beginning of the list. Print the final list.
'''
numbers = [1, 2, 3, 4, 5]
numbers.append(6)
numbers.remove(3)
numbers.insert(0, 0)
print(numbers)



'''Exercise 2: Tuple Operations
Create a tuple called `coordinates` containing the elements `10.0`,
`20.0`, and `30.0`.
Access and print the second element of the tuple.
Try to change the third element of the tuple to `40.0`. What happens?"**
'''
coordinates = (10.0, 20.0, 30.0)
print(coordinates[1])
#coordinates[2] = 40.0  # gives a TypeError because tuples are immutable.



'''Exercise 3: Set Operations
Create a set called `fruits` containing 'apple', 'banana', 'cherry'.
Add 'orange' to the set. Remove 'banana' from the set.
Check if 'cherry' is in the set and print a message based on the result.
Create another set called `citrus` with elements 'orange', 'lemon',
'lime'.
Perform a union of `fruits` and `citrus` and print the result.
Perform an intersection of `fruits` and `citrus` and print the result.
'''
fruits = {"apple", "banana", "cherry"}
fruits.add("orange")
fruits.remove("banana")

if "cherry" in fruits:
    print("cherry is in the set")
else:
```

```
    print("cherry is not in the set")

citrus = {"orange", "lemon", "lime"}
union_set = fruits.union(citrus)
intersection_set = fruits.intersection(citrus)
print(union_set)
print(intersection_set)



'''Exercise 4: Dictionary Operations
Create a dictionary called `person` with keys 'name', 'age', and 'city',
and values 'John', 30, and 'New York', respectively.
Access and print the 'name' key from the dictionary.
Update the 'age' key to 31.
Add a new key-value pair 'email': 'john@example.com' to the dictionary.
Remove the 'city' key from the dictionary.
Print the final dictionary."
'''
person = {"name": "John", "age": 30, "city": "New York"}
print(person["name"])
person["age"] = 31
person["email"] = "john@example.com"
del person["city"]
print(person)



'''Exercise 5: Nested Dictionary
Create a dictionary called `school` where the keys are student names and
the values are dictionaries containing the subjects and their
corresponding grades.
Print the grade of 'Alice' in 'Math'.
Add a new student 'David' with grades 'Math': 80 and 'Science': 89.
Update 'Bob's 'Science' grade to 95. Print the final `school` dictionary."
'''
school = {
    "Alice": {"Math": 90, "Science": 85},
    "Bob": {"Math": 78, "Science": 92},
    "Charlie": {"Math": 95, "Science": 88}
}
print(school["Alice"]["Math"])
school["David"] = {"Math": 80, "Science": 89}
school["Bob"]["Science"] = 95
print(school)
```

```
'''Exercise 6: List Comprehension
Given a list of numbers `[1, 2, 3, 4, 5]`, use list comprehension to
create a new list where each number is squared.
Print the new list.
'''
numbers = [1, 2, 3, 4, 5]
squared_numbers = [x**2 for x in numbers]
print(squared_numbers)



'''Exercise 7: Set Comprehension
Create a set comprehension that generates a set of squared numbers from
the list `[1, 2, 3, 4, 5]`.
Print the resulting set.
'''
squared_set = {x**2 for x in [1, 2, 3, 4, 5]}
print(squared_set)



'''Exercise 8: Dictionary Comprehension
Create a dictionary comprehension that generates a dictionary where the
keys are the numbers from `1` to `5`, and the values are the cubes of the
keys.
Print the resulting dictionary.
'''
cubes = {x: x**3 for x in range(1, 6)}
print(cubes)



'''Exercise 9: Combining Collections
Create two lists: `keys = ['name', 'age', 'city']` and `values = ['Alice',
25, 'Paris']`.
Use the `zip()` function to combine the `keys` and `values` lists into a
dictionary.
Print the resulting dictionary.
'''
keys = ["name", "age", "city"]
values = ["Alice", 25, "Paris"]
combined_dict = dict(zip(keys, values))
print(combined_dict)
```

```
'''Exercise 10: Count Word Occurrences (Using a Dictionary)
Write a Python program that takes a string as input and counts the
occurrences of each word in the string using a dictionary.
Example input: `sentence = 'the quick brown fox jumps over the lazy dog
the fox'`.
Print the resulting dictionary with word counts.
'''
sentence = "the quick brown fox jumps over the lazy dog the fox"
words = sentence.split()
word_count = {word: words.count(word) for word in set(words)}
print(word_count)


'''Exercise 11: Unique Elements in Two Sets
Create two sets: `set1 = {1, 2, 3, 4, 5}` and `set2 = {4, 5, 6, 7, 8}`.
Find and print the unique elements in both sets combined.
Find and print the common elements between the two sets.
Find and print the elements that are only in `set1` but not in `set2`.
'''
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}
unique_elements = set1.union(set2)
common_elements = set1.intersection(set2)
only_in_set1 = set1.difference(set2)
print(unique_elements)
print(common_elements)
print(only_in_set1)


'''Exercise 12: Tuple Unpacking
Create a tuple with three elements: `('Alice', 25, 'Paris')`.
Unpack the tuple into three variables: `name`, `age`, and `city`.
Print the variables to verify the unpacking.
'''
info = ("Alice", 25, "Paris")
name, age, city = info
print(name, age, city)
```

```
'''Exercise 13: Frequency Counter with Dictionary
Write a Python program that counts the frequency of each letter in a given
string using a dictionary.
Example string: `text = 'hello world'`.
Print the resulting dictionary with letter frequencies.
'''
text = "hello world"
frequency = {letter: text.count(letter) for letter in set(text)}
print(frequency)

text = "hello world"
frequency = {}
for i in text:
    frequency[i]=frequency.get(i,0)+1
print(frequency)



'''Exercise 14: Sorting a List of Tuples
Given a list of tuples representing students and their grades: `students =
[('Alice', 90), ('Bob', 80), ('Charlie', 85)]`,
sort the list by grades in descending order and print the sorted list.
'''
students = [("Alice", 90), ("Bob", 80), ("Charlie", 85)]
sorted_students = sorted(students, key=lambda stu: stu[1], reverse=True)
print(sorted_students)
```