1. Write a program to create tuples (name, age, address, college) for at least two members and concatenate the tuples and print the concatenated tuples.

Answer:

member1 = ("Alice", 20, "123 Main St, Springfield", "Springfield University")

member2 = ("Bob", 22, "456 Elm St, Springfield", "Springfield College")

# Concatenating the tuples

combined\_members = member1 + member2

# Printing the concatenated tuples

print("Concatenated Tuples:")

print(combined\_members)

1. Write a program to count the number of vowels in a string (No control flow allowed).

Answer:

def count\_vowels(s):

# Define the vowels

vowels = "aeiouAEIOU"

# Use the sum function with a generator expression to count vowels

return sum(map(s.count, vowels))

# Example string

input\_string = "Hello, how are you?"

# Count and print the number of vowels

vowel\_count = count\_vowels(input\_string)

print("Number of vowels:", vowel\_count)

**Explanation:**

* The count\_vowels function uses map to apply the count method for each vowel in the string.
* sum then adds up all the counts to give the total number of vowels.

**Output**

When you run this program with the input string "Hello, how are you?", it will output:

Number of vowels: 7

1. Write a program to check if a given key exists in a dictionary or not.

Answer:

my\_dict = {

"name": "Alice",

"age": 25,

"city": "New York"

}

# Function to check if a key exists in the dictionary

def check\_key\_exists(dictionary, key):

return key in dictionary

# Key to check

key\_to\_check = "age"

# Checking if the key exists

exists = check\_key\_exists(my\_dict, key\_to\_check)

# Output the result

if exists:

print(f"The key '{key\_to\_check}' exists in the dictionary.")

else:

print(f"The key '{key\_to\_check}' does not exist in the dictionary.")

**Explanation:**

* The check\_key\_exists function takes a dictionary and a key as arguments and returns True if the key exists, and False otherwise.
* The program then checks for the key "age" and prints the result.

**Output**

If you run this program, it will output:

The key 'age' exists in the dictionary.

4. Write a program to add a new key-value pair to an existing dictionary.

Answer:

my\_dict = {

"name": "Alice",

"age": 25,

"city": "New York"

}

# Function to add a new key-value pair

def add\_key\_value(dictionary, key, value):

dictionary[key] = value

# New key-value pair to add

new\_key = "email"

new\_value = "alice@example.com"

# Adding the new key-value pair

add\_key\_value(my\_dict, new\_key, new\_value)

# Output the updated dictionary

print("Updated Dictionary:")

print(my\_dict)

**Explanation:**

* The add\_key\_value function takes a dictionary, a key, and a value as arguments and adds the new key-value pair to the dictionary.
* After defining the new key and value, the function is called to update the dictionary.

**Output**

Updated Dictionary:

{'name': 'Alice', 'age': 25, 'city': 'New York', 'email': 'alice@example.com'}

5. Write a program to sum all the items in a given dictionary.

Answer:

my\_dict = {

"item1": 10,

"item2": 20,

"item3": 30,

"item4": 40

}

# Function to sum all the values in the dictionary

def sum\_dict\_values(dictionary):

return sum(dictionary.values())

# Calculating the sum of all items

total\_sum = sum\_dict\_values(my\_dict)

# Output the result

print("Sum of all items in the dictionary:", total\_sum)

**Explanation:**

* The sum\_dict\_values function uses the sum function on the values of the dictionary to calculate the total.
* It retrieves the values using dictionary.values().

**Output**

When you run this program, it will output:

Sum of all items in the dictionary: 100