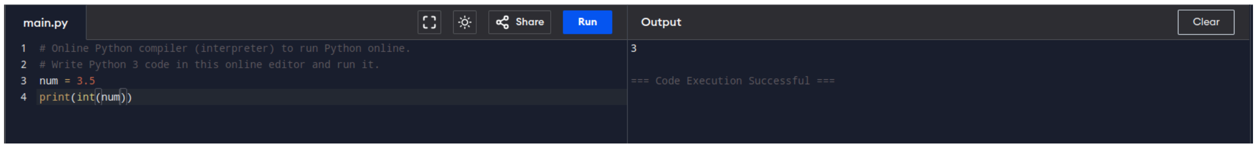
DATA TYPES

DATA TYPES

Objective: Convert between different data types.

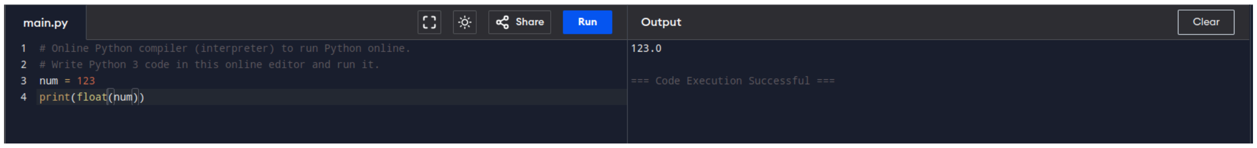
1. **Task:** Convert the following values to the specified types and print the results
    1. Convert `3.75` to an integer and print the value



```
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
num = 3.5
print(int(num))
```

```
3

=== Code Execution Successful ===
```

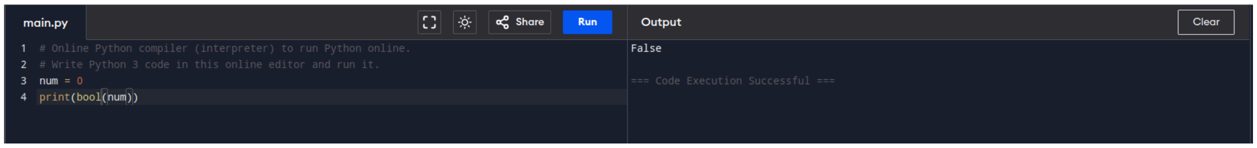
    2. Convert `"123"` to a float and print the value



```
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
num = 123
print(float(num))
```

```
123.0

=== Code Execution Successful ===
```

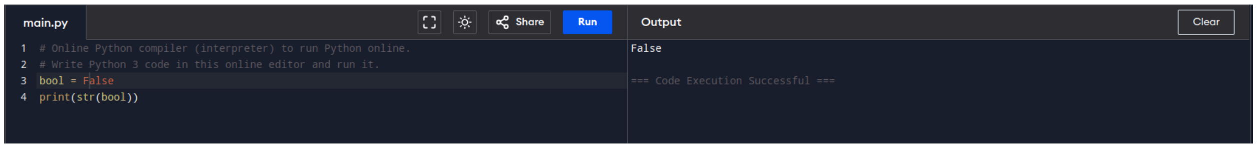
    3. Convert `0` to a boolean and print the value



```
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
num = 0
print(bool(num))
```

```
False

=== Code Execution Successful ===
```

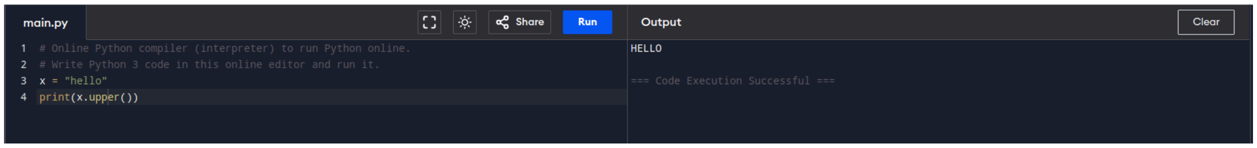
    4. Convert `False` to a string and print the value



```
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
bool = False
print(str(bool))
```

```
False

=== Code Execution Successful ===
```

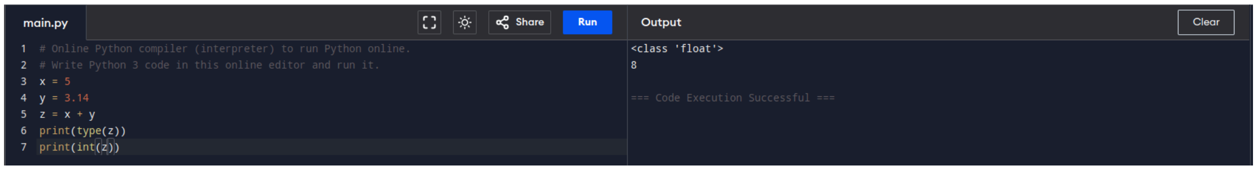
2. Convert all characters in the string to uppercase. x = "hello"



```
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
x = "hello"
print(x.upper())
```

```
HELLO

=== Code Execution Successful ===
```

3. Given `x = 5` and `y = 3.14`, calculate `z = x + y` and determine the data type of `z`. And convert it to integer.

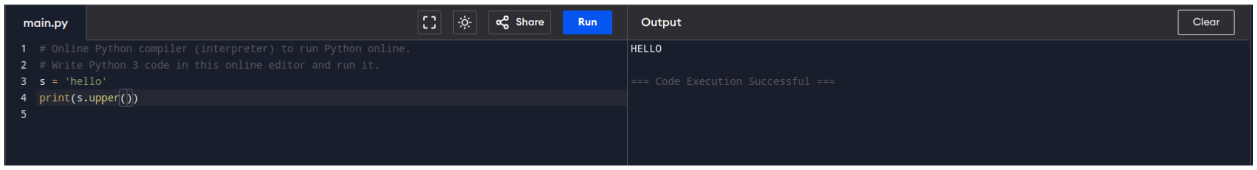


```
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
x = 5
y = 3.14
z = x + y
print(type(z))
print(int(z))
```

```
<class 'float'>
8

=== Code Execution Successful ===
```

4. Given the string `s = 'hello'`, perform the following operations:

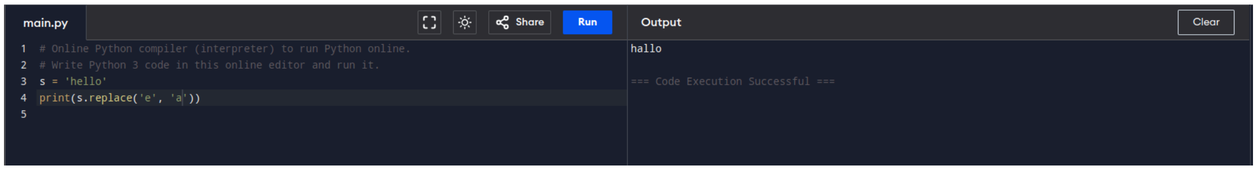
- Convert the string to uppercase.



```python
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
s = 'hello'
print(s.upper())
```

```
HELLO

=== Code Execution Successful ===
```

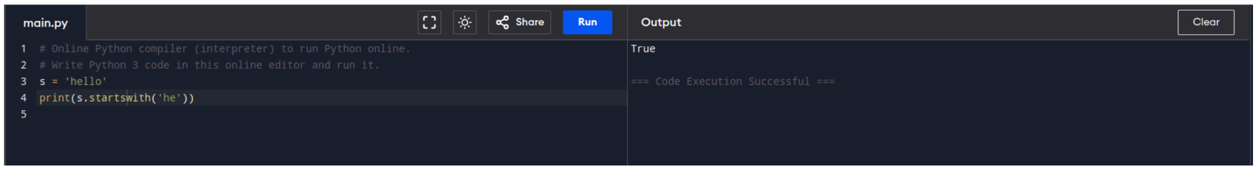
- Replace 'e' with 'a'.



```python
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
s = 'hello'
print(s.replace('e', 'a'))
```

```
hallo

=== Code Execution Successful ===
```

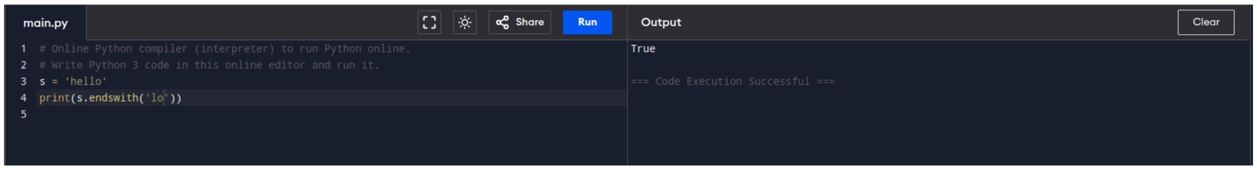
- Check if the string starts with 'he'.



```python
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
s = 'hello'
print(s.startswith('he'))
```

```
True

=== Code Execution Successful ===
```

- Check if the string ends with 'lo'.



```python
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
s = 'hello'
print(s.endswith('lo'))
```

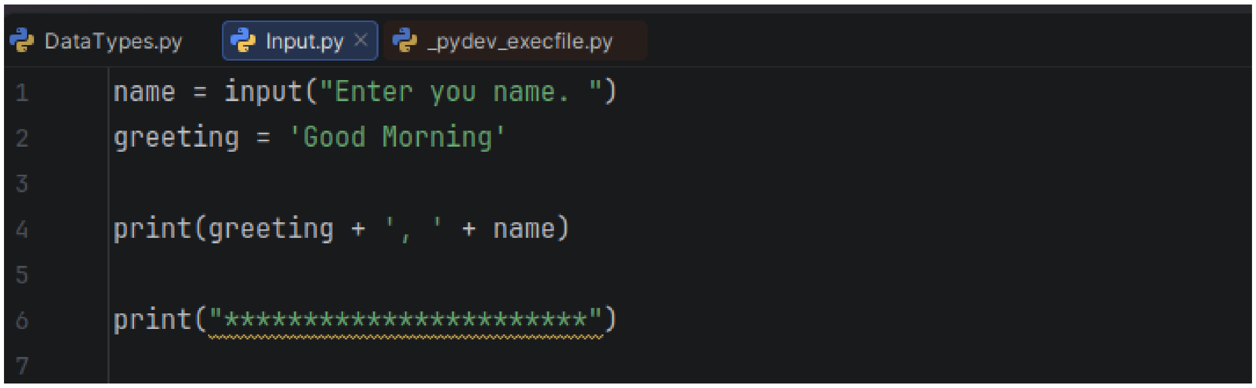
```
True

=== Code Execution Successful ===
```

INPUT AND PRINT

INPUT AND PRINT

1. **Objective:** Ask the user for their name and greet them.

   Task: Write a program that asks the user for their name and then prints a greeting message using their name.

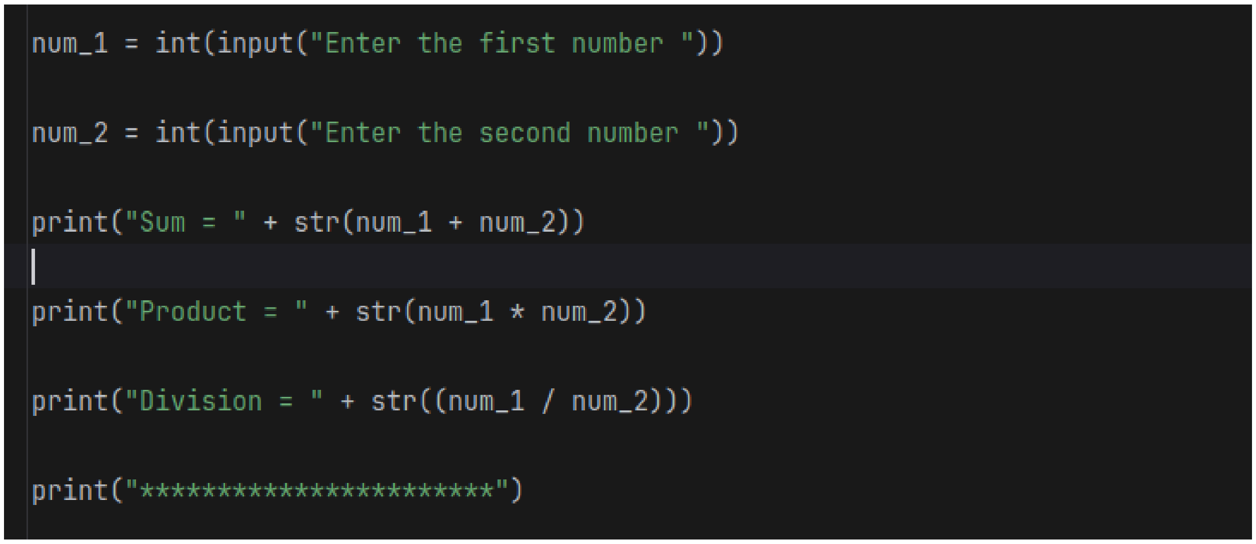


DataTypes.py | Input.py | _pydev_execfile.py

```python
name = input("Enter you name. ")
greeting = 'Good Morning'

print(greeting + ', ' + name)

print("***********************")
```

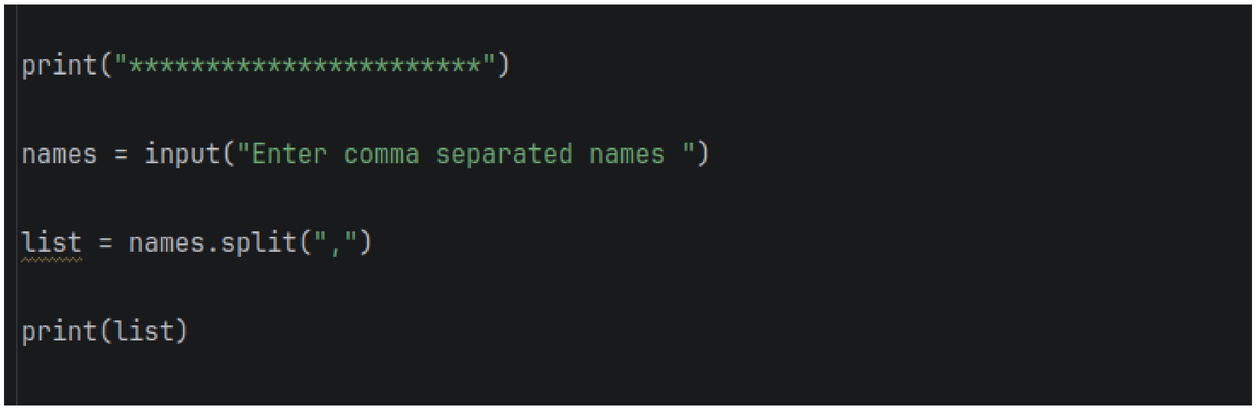
2. **Objective:** Perform basic arithmetic operations based on user input.

   Task: Ask the user to enter two numbers from the user and print their sum, multiplication, and division.



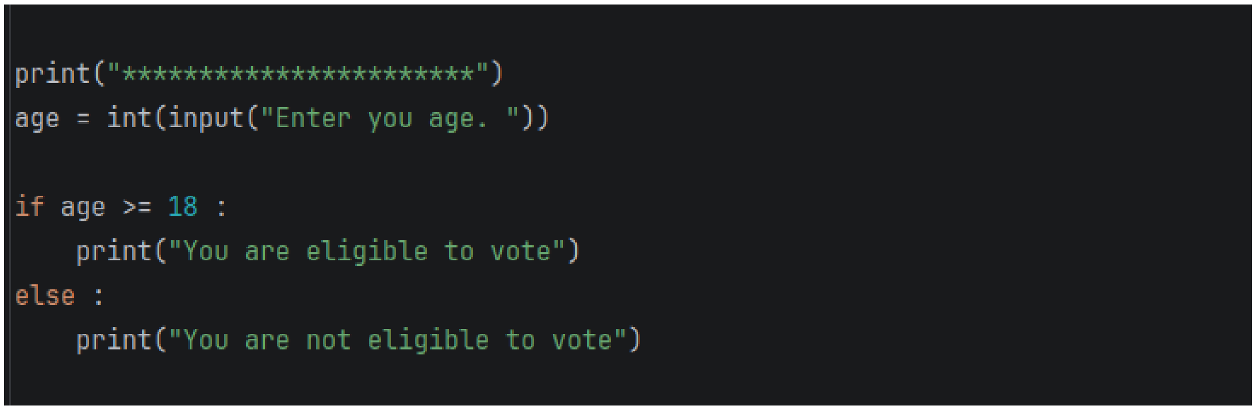
```python
num_1 = int(input("Enter the first number "))

num_2 = int(input("Enter the second number "))

print("Sum = " + str(num_1 + num_2))

print("Product = " + str(num_1 * num_2))

print("Division = " + str((num_1 / num_2)))

print("***********************")
```

3. **Task:** Ask the user to enter input names separated by commas, split the string from comma and copy to a list and print.



```
print("***********************")

names = input("Enter comma separated names ")

list = names.split(",")

print(list)
```

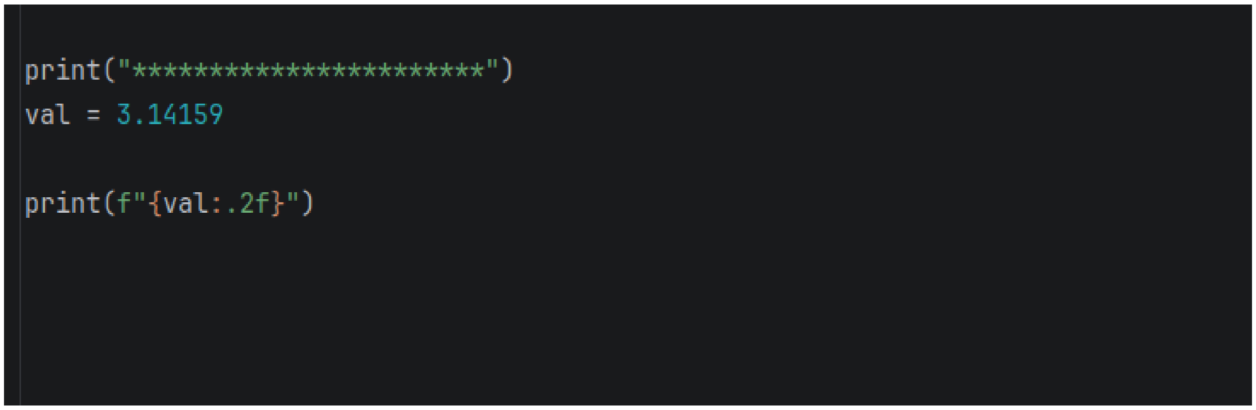
4. **Task:** Ask the user to enter their age and check if they are eligible to vote based on their age.



```
print("***********************")
age = int(input("Enter you age. "))

if age >= 18 :
    print("You are eligible to vote")
else :
    print("You are not eligible to vote")
```

5. For value = 3.14159, Using f-string print output for only up to 2 decimal places.

   Output: 3.14



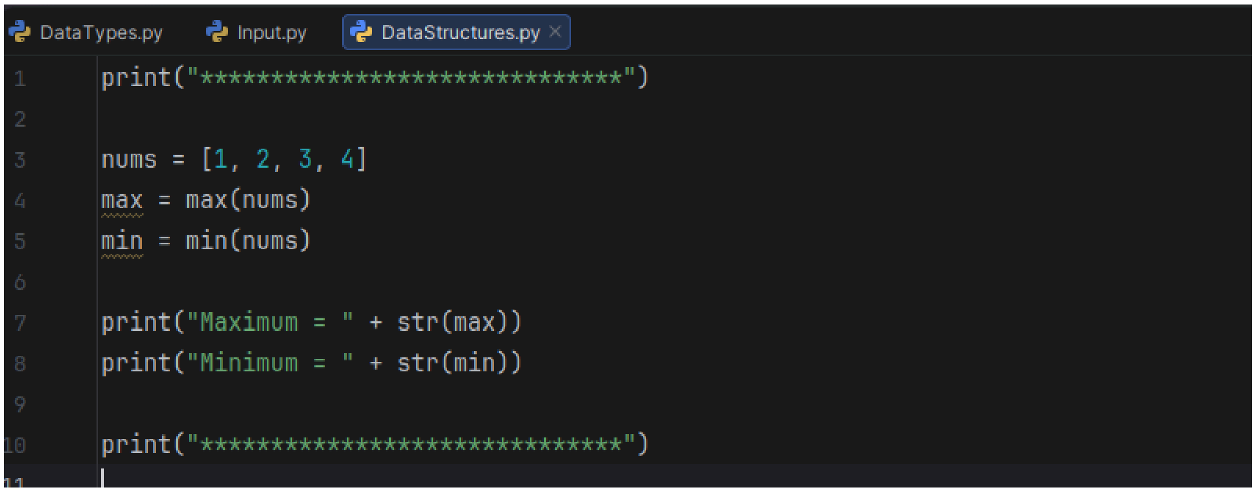
```
print("***********************")
val = 3.14159

print(f"{val:.2f}")
```

BUILT IN DATA STRUCTURES

BUILT IN DATA STRUCTURES

1. Given a list of numbers, find and print the maximum and minimum values.

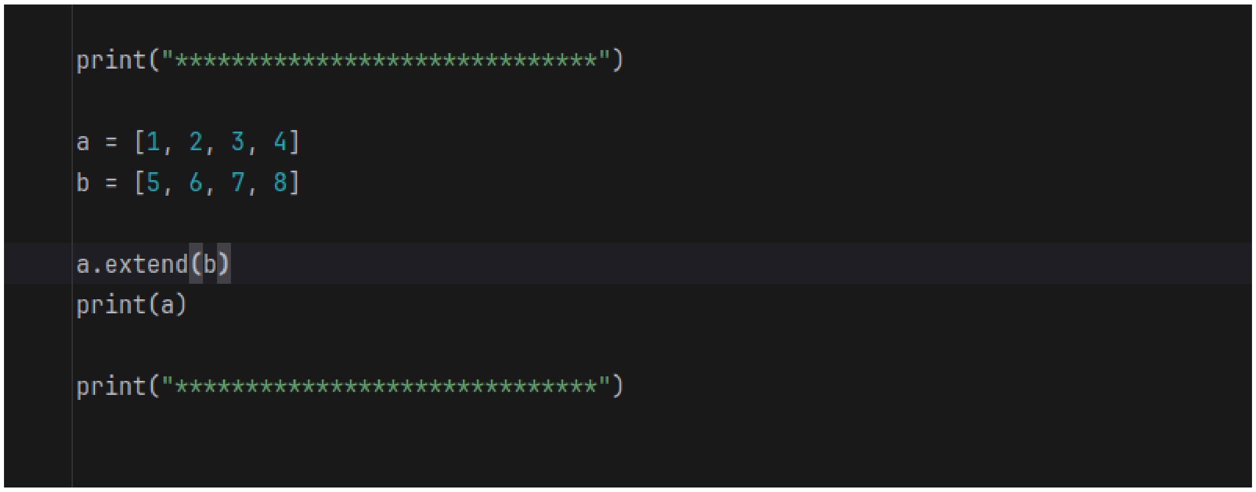
    nums = [1, 2, 3, 4, 5]


DataTypes.py   Input.py   DataStructures.py

```python
print("******************************")

nums = [1, 2, 3, 4]
max = max(nums)
min = min(nums)

print("Maximum = " + str(max))
print("Minimum = " + str(min))

print("******************************")
```

2. Given two lists below, merge the values from both lists to one and print

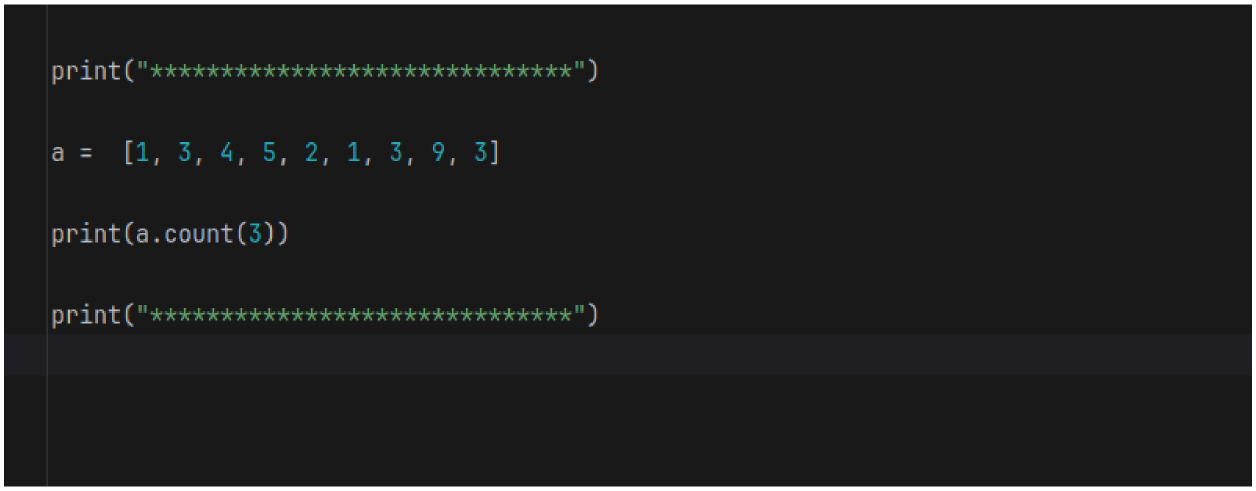
    a = [1,2,3,4]   b = [5,6,7,8]



```python
print("******************************")

a = [1, 2, 3, 4]
b = [5, 6, 7, 8]

a.extend(b)
print(a)

print("******************************")
```

3. From a list, print the number of times the value 3 appears in the list:

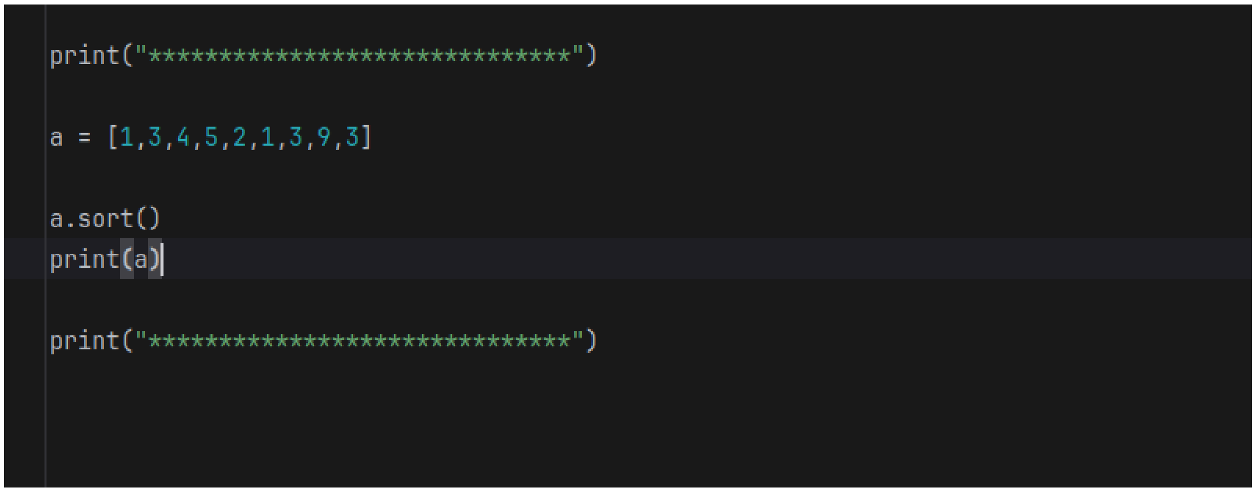
    a = [1,3,4,5,2,1,3,9,3]



```python
print("******************************")

a =  [1, 3, 4, 5, 2, 1, 3, 9, 3]

print(a.count(3))

print("******************************")
```

4. From below list, Sort the list and print

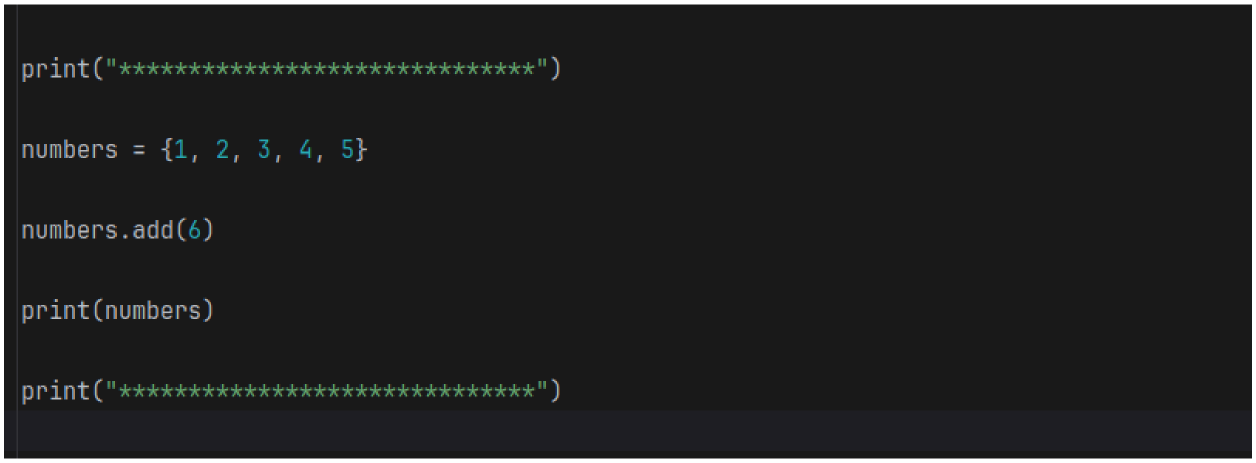
    a = [1,3,4,5,2,1,3,9,3]



```python
print("******************************")

a = [1,3,4,5,2,1,3,9,3]

a.sort()
print(a)

print("******************************")
```

5. Given a set, add the element 6 to it and print the updated set.

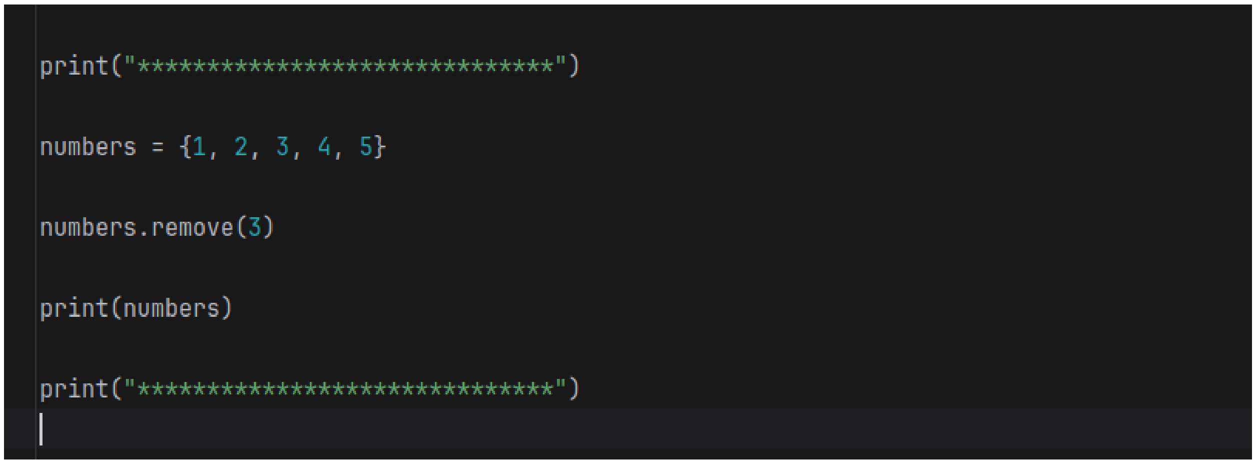
    numbers = {1, 2, 3, 4, 5}



```
print("******************************")

numbers = {1, 2, 3, 4, 5}

numbers.add(6)

print(numbers)

print("******************************")
```

6. Given a set, remove the element 3 from it and print the updated set.

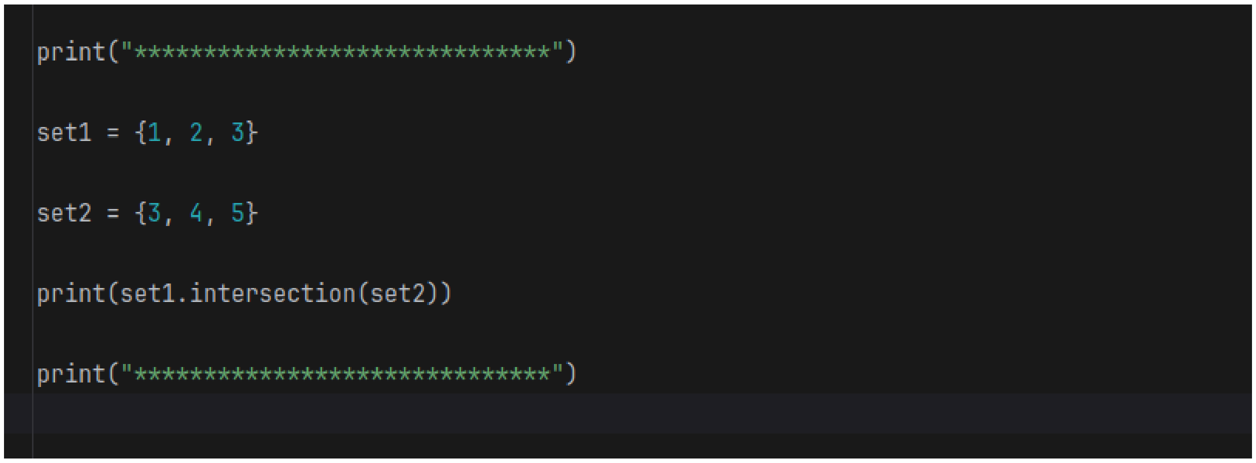
numbers = {1, 2, 3, 4, 5}



```
print("******************************")

numbers = {1, 2, 3, 4, 5}

numbers.remove(3)

print(numbers)

print("******************************")
```

7. Given two sets, find and print their intersection.

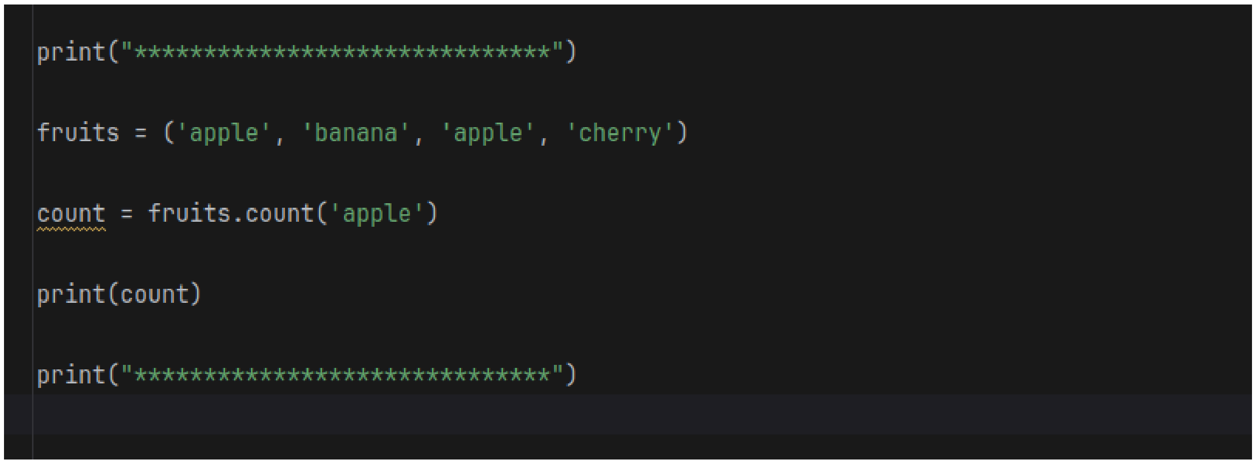
set1 = {1, 2, 3}   set2 = {3, 4, 5}



```python
print("******************************")

set1 = {1, 2, 3}

set2 = {3, 4, 5}

print(set1.intersection(set2))

print("******************************")
```

8. Given a tuple, count and print the number of occurrences of the element 'apple'.

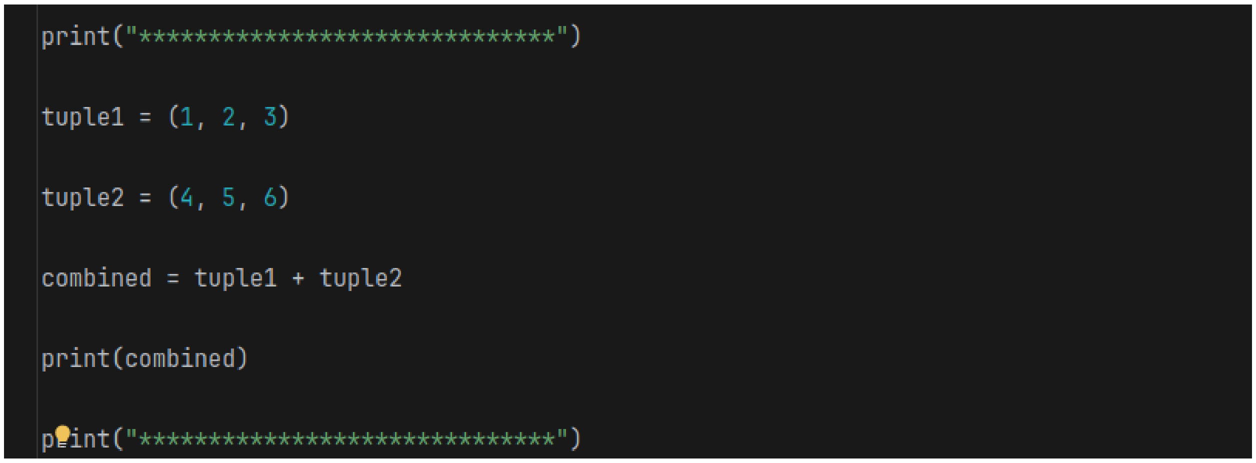
    fruits = ('apple', 'banana', 'apple', 'cherry')



```python
print("******************************")

fruits = ('apple', 'banana', 'apple', 'cherry')

count = fruits.count('apple')

print(count)

print("******************************")
```

9. Given two tuples, concatenate them and print the result.

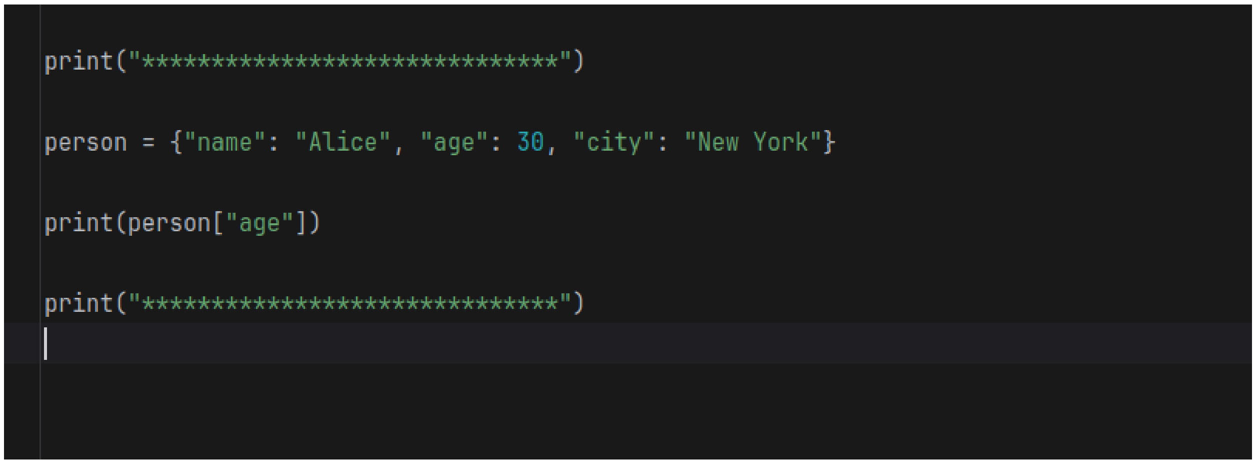
    tuple1 = (1, 2, 3)   tuple2 = (4, 5, 6)



```python
print("******************************")

tuple1 = (1, 2, 3)

tuple2 = (4, 5, 6)

combined = tuple1 + tuple2

print(combined)

print("******************************")
```

10. Access and print the value associated with the key `"age"` from the dictionary.

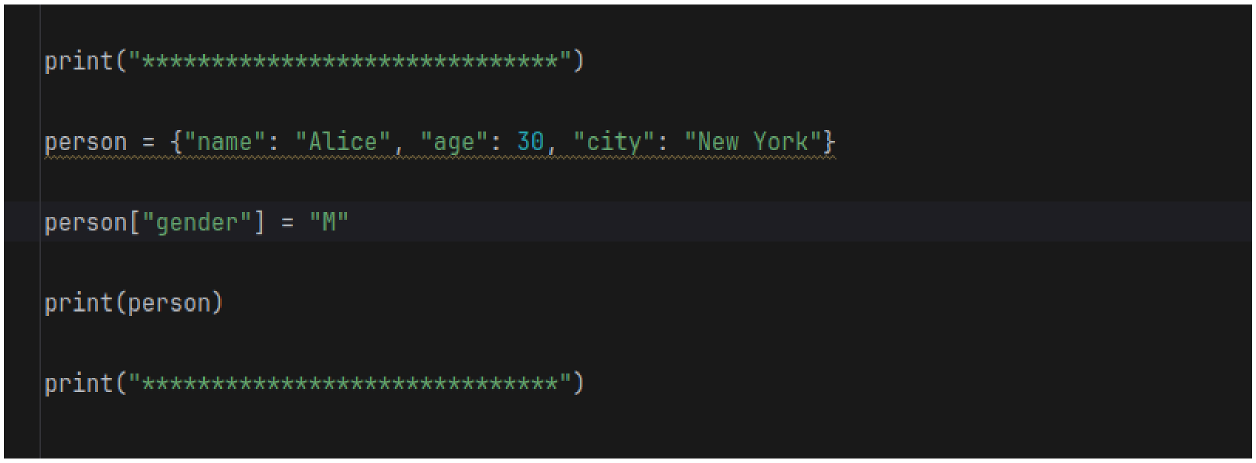
    person = {"name": "Alice", "age": 30, "city": "New York"}



```python
print("******************************")

person = {"name": "Alice", "age": 30, "city": "New York"}

print(person["age"])

print("******************************")
```

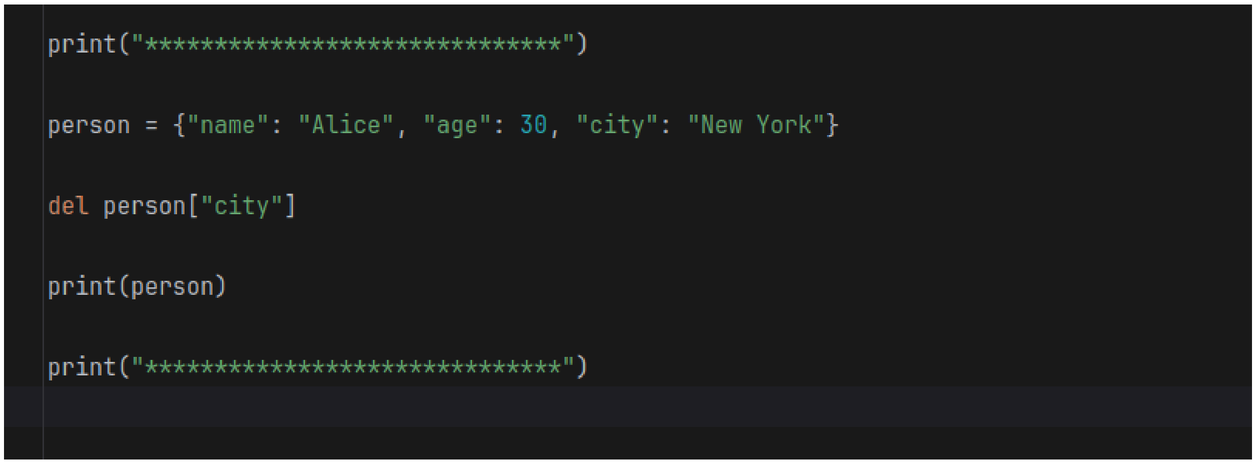
11. Add new key, gender to dictionary and assign “M” to it and print

    person = {"name": "Alice", "age": 30, "city": "New York"}



```python
print("******************************")

person = {"name": "Alice", "age": 30, "city": "New York"}

person["gender"] = "M"

print(person)

print("******************************")
```

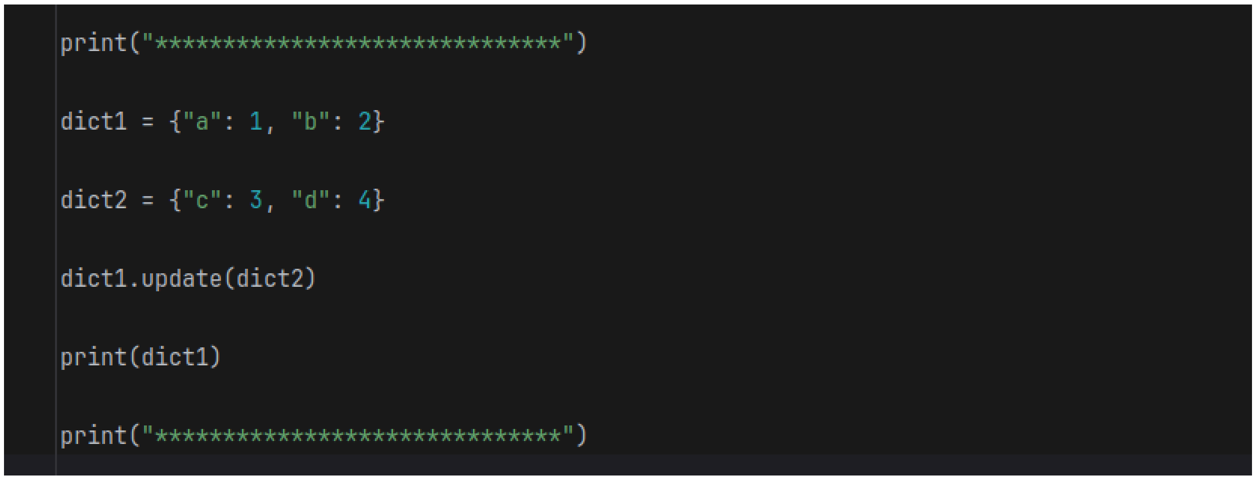
12. Remove the key `"city"` from the above Dict and print



```python
print("******************************")

person = {"name": "Alice", "age": 30, "city": "New York"}

del person["city"]

print(person)

print("******************************")
```

13. Given two dictionaries, merge them into one

dict1 = {"a": 1, "b": 2}   dict2 = {"c": 3, "d": 4}



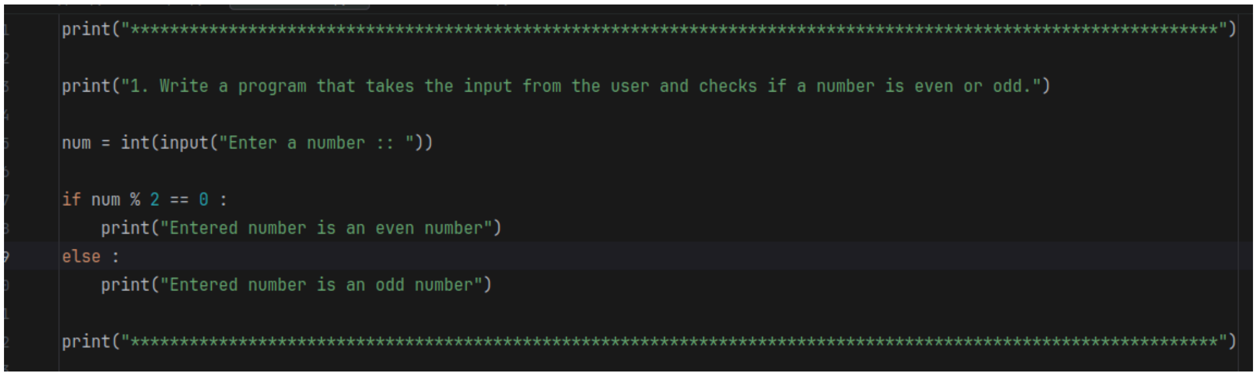
```python
print("******************************")

dict1 = {"a": 1, "b": 2}

dict2 = {"c": 3, "d": 4}

dict1.update(dict2)

print(dict1)

print("******************************")
```

CONTROL FLOW

CONTROL FLOW

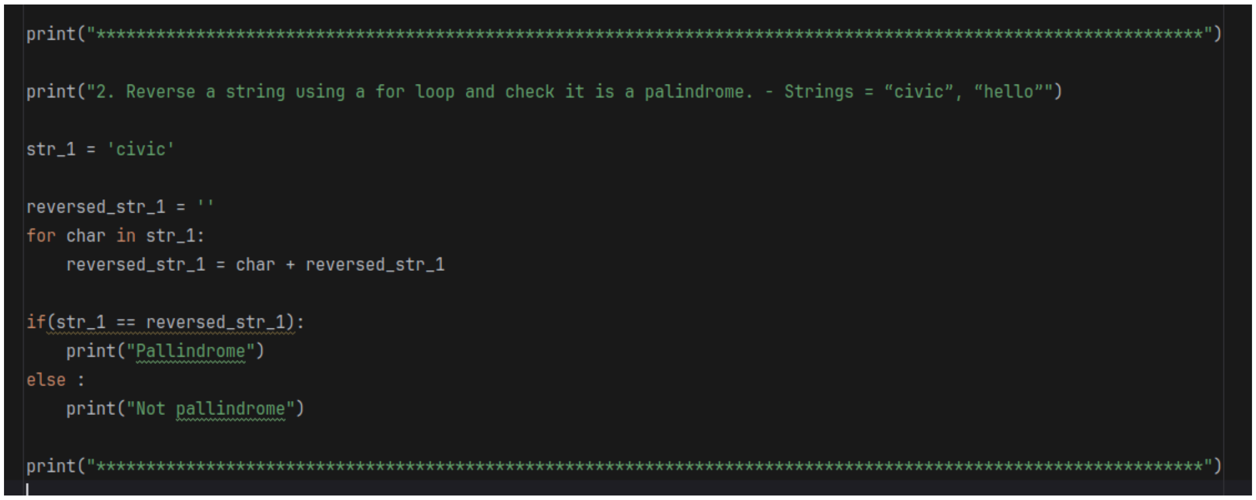
For loop

1. Write a program that takes the input from the user and checks if a number is even or odd.



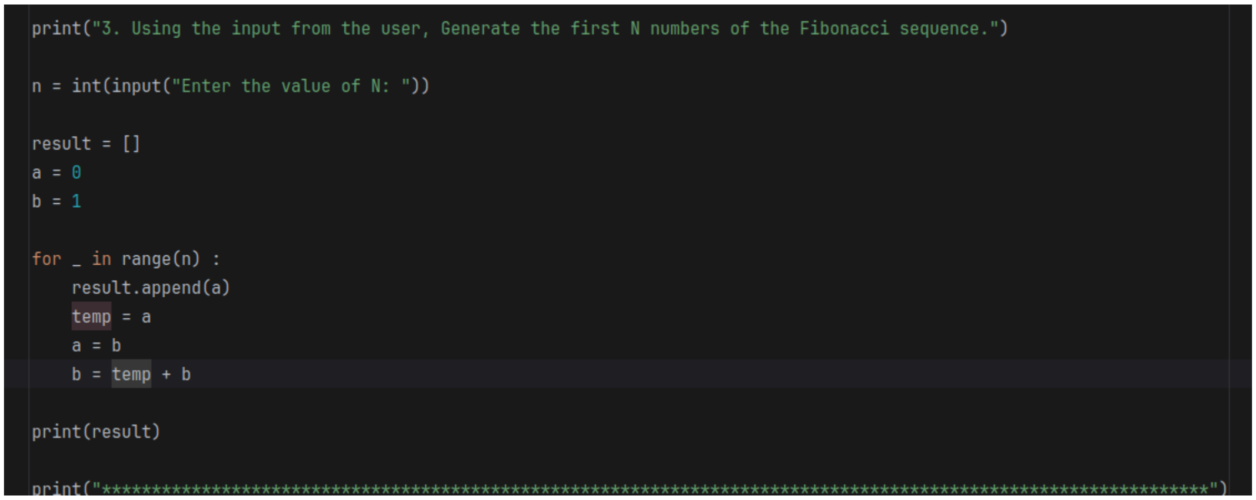
```python
print("***************************************************************************************************************")

print("1. Write a program that takes the input from the user and checks if a number is even or odd.")

num = int(input("Enter a number :: "))

if num % 2 == 0 :
    print("Entered number is an even number")
else :
    print("Entered number is an odd number")

print("***************************************************************************************************************")
```

2. Reverse a string using a for loop and check it is a palindrome. - Strings = “civic”, “hello”



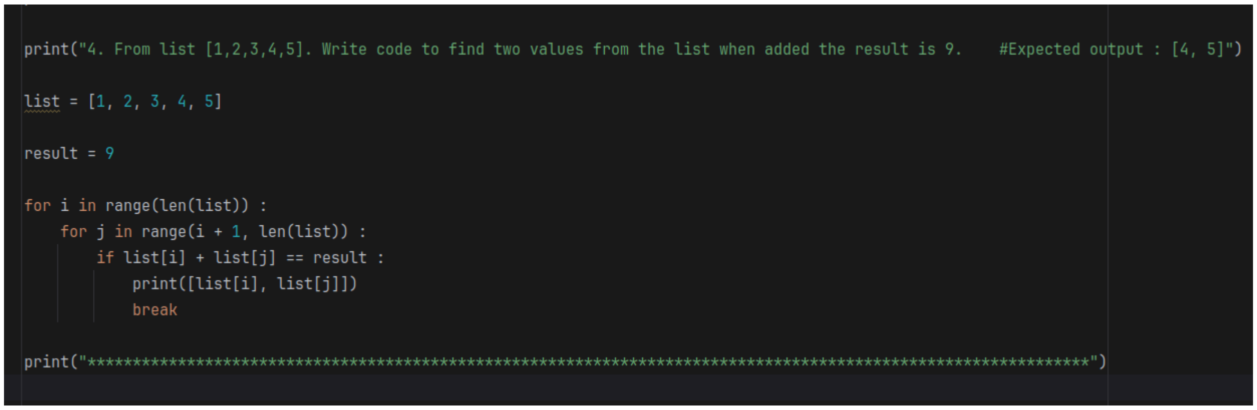
```python
print("***************************************************************************************************************")

print("2. Reverse a string using a for loop and check it is a palindrome. - Strings = “civic”, “hello”")

str_1 = 'civic'

reversed_str_1 = ''
for char in str_1:
    reversed_str_1 = char + reversed_str_1

if(str_1 == reversed_str_1):
    print("Pallindrome")
else :
    print("Not pallindrome")

print("***************************************************************************************************************")
```

3. Using the input from the user, Generate the first N numbers of the Fibonacci sequence.



```python
print("3. Using the input from the user, Generate the first N numbers of the Fibonacci sequence.")

n = int(input("Enter the value of N: "))

result = []
a = 0
b = 1

for _ in range(n) :
    result.append(a)
    temp = a
    a = b
    b = temp + b

print(result)

print("***************************************************************************************************************")
```

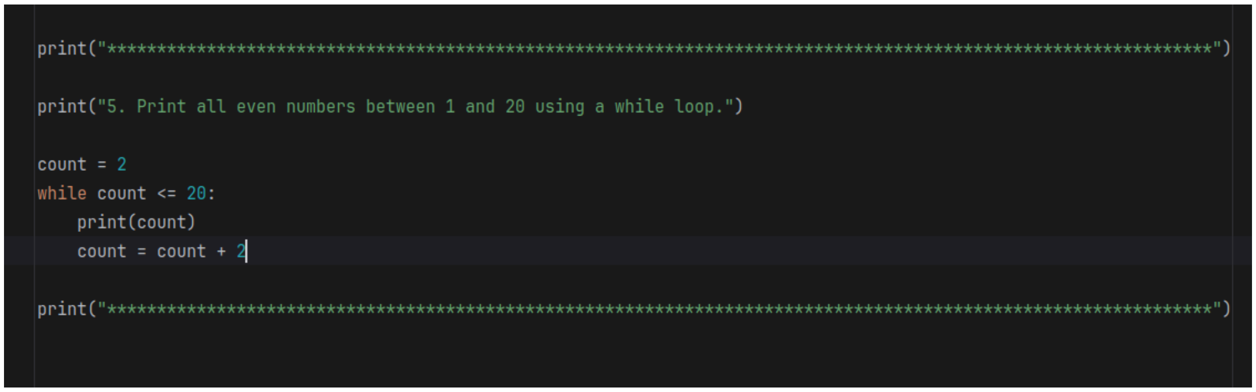
4. From list [1,2,3,4,5]. Write code to find two values from the list when added the result is 9.
   #Expected output : [4, 5]



```python
print("4. From list [1,2,3,4,5]. Write code to find two values from the list when added the result is 9.    #Expected output : [4, 5]")

list = [1, 2, 3, 4, 5]

result = 9

for i in range(len(list)) :
    for j in range(i + 1, len(list)) :
        if list[i] + list[j] == result :
            print([list[i], list[j]])
            break

print("*****************************************************************************************************************")
```

While loop

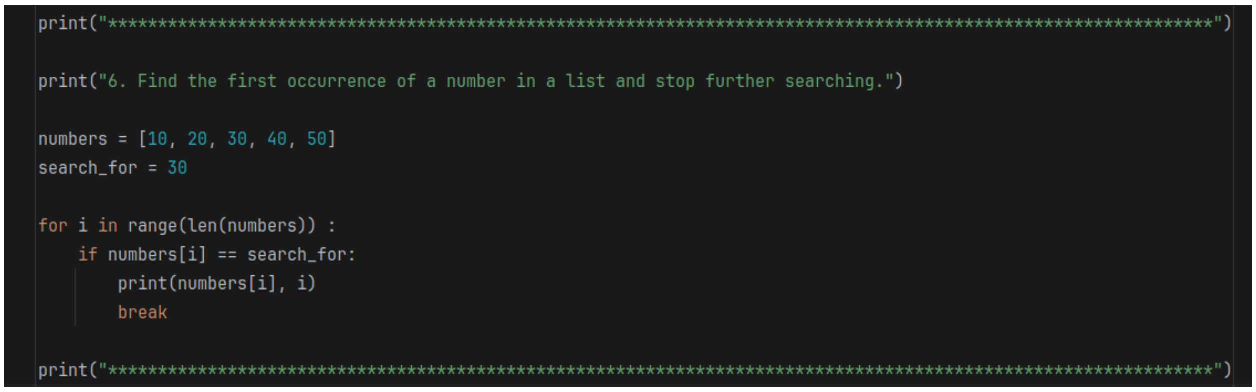
5. Print all even numbers between 1 and 20 using a `while` loop.



```python
print("*****************************************************************************************************************")

print("5. Print all even numbers between 1 and 20 using a while loop.")

count = 2
while count <= 20:
    print(count)
    count = count + 2

print("*****************************************************************************************************************")
```

Break

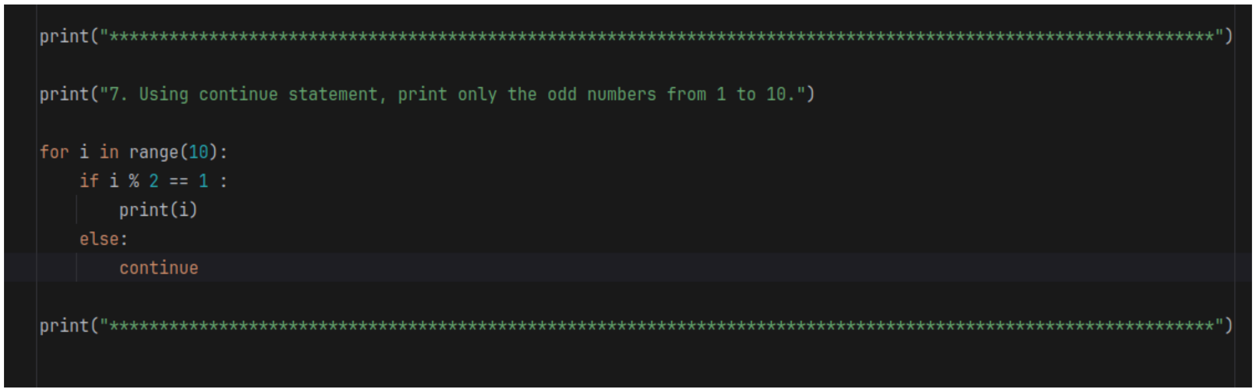
6. Find the first occurrence of a number in a list and stop further searching.
   numbers = [10, 20, 30, 40, 50]
   search_for = 30



```python
print("*****************************************************************************************************************")

print("6. Find the first occurrence of a number in a list and stop further searching.")

numbers = [10, 20, 30, 40, 50]
search_for = 30

for i in range(len(numbers)) :
    if numbers[i] == search_for:
        print(numbers[i], i)
        break

print("*****************************************************************************************************************")
```

Continue

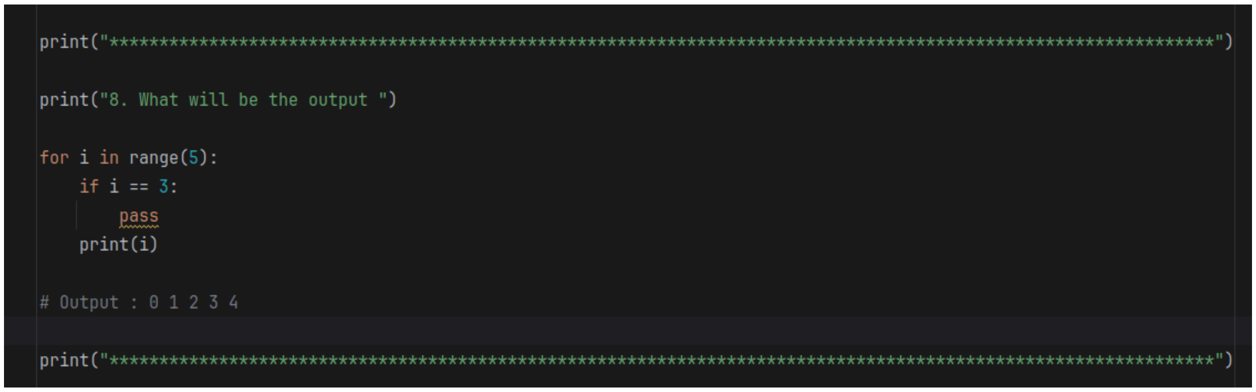
7. Using continue statement, print only the odd numbers from 1 to 10.



```python
print("****************************************************************************************************************")

print("7. Using continue statement, print only the odd numbers from 1 to 10.")

for i in range(10):
    if i % 2 == 1 :
        print(i)
    else:
        continue

print("****************************************************************************************************************")
```

Pass

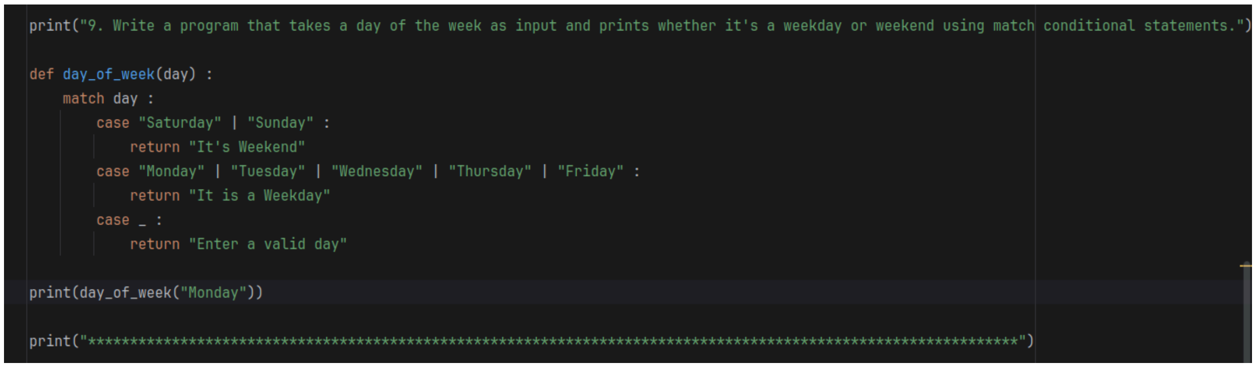
8. What will be the output
   for i in range(5):
       if i == 3:
           pass
       print(i)



```python
print("****************************************************************************************************************")

print("8. What will be the output ")

for i in range(5):
    if i == 3:
        pass
    print(i)

# Output : 0 1 2 3 4

print("****************************************************************************************************************")
```

Match

9. Write a program that takes a day of the week as input and prints whether it's a weekday or weekend using match conditional statements.

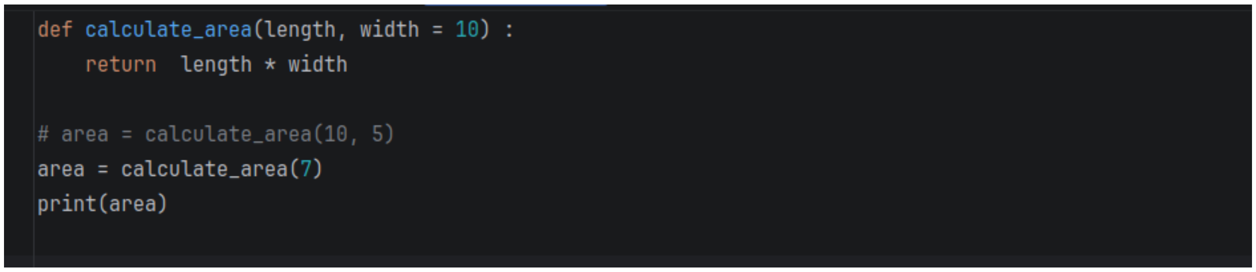


```python
print("9. Write a program that takes a day of the week as input and prints whether it's a weekday or weekend using match conditional statements.")

def day_of_week(day) :
    match day :
        case "Saturday" | "Sunday" :
            return "It's Weekend"
        case "Monday" | "Tuesday" | "Wednesday" | "Thursday" | "Friday" :
            return "It is a Weekday"
        case _ :
            return "Enter a valid day"

print(day_of_week("Monday"))

print("****************************************************************************************************************")
```

FUNCTIONS

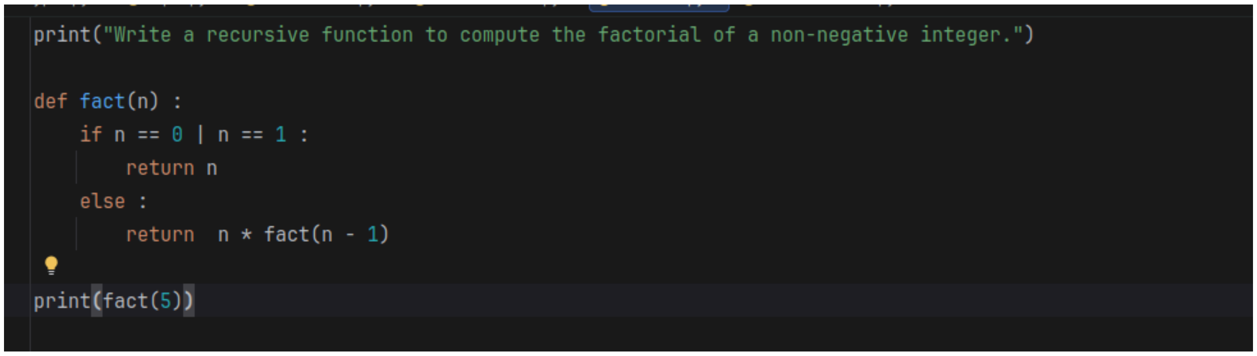
FUNCTIONS

1. Define a function `calculate_area` that calculates the area of a rectangle and return the result. If no width is provided, it defaults to 10.



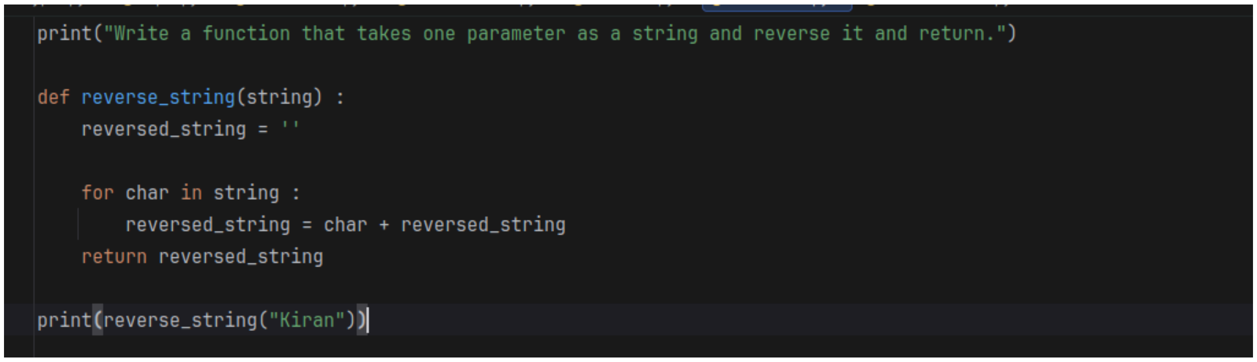
```
def calculate_area(length, width = 10) :
    return  length * width

# area = calculate_area(10, 5)
area = calculate_area(7)
print(area)
```

2. Write a recursive function to compute the factorial of a non-negative integer.



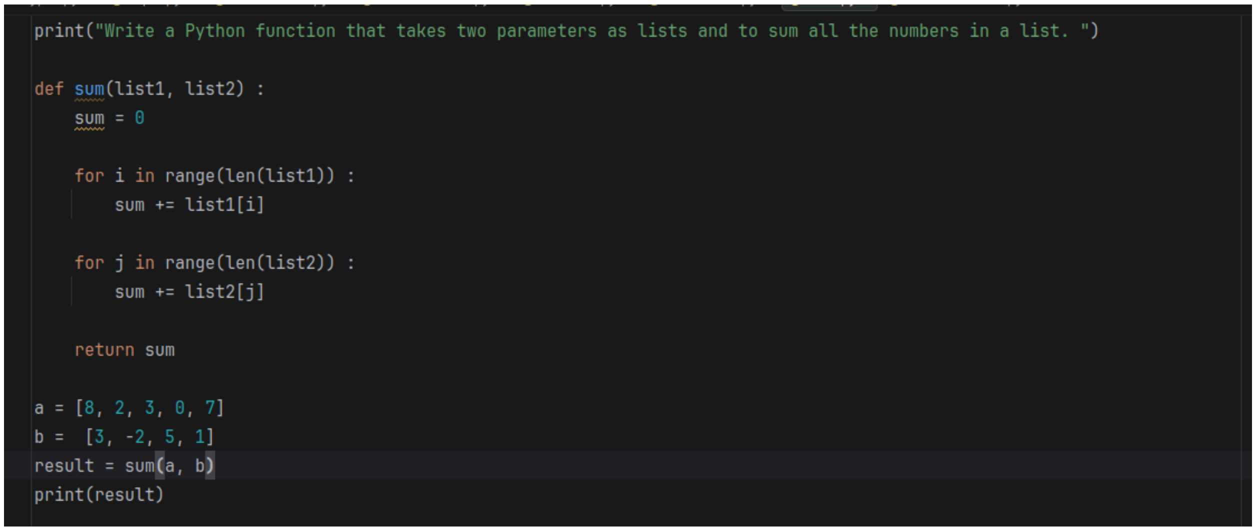
```
print("Write a recursive function to compute the factorial of a non-negative integer.")

def fact(n) :
    if n == 0 | n == 1 :
        return n
    else :
        return  n * fact(n - 1)

print(fact(5))
```

3. Write a function that takes one parameter as a string and reverse it and return.

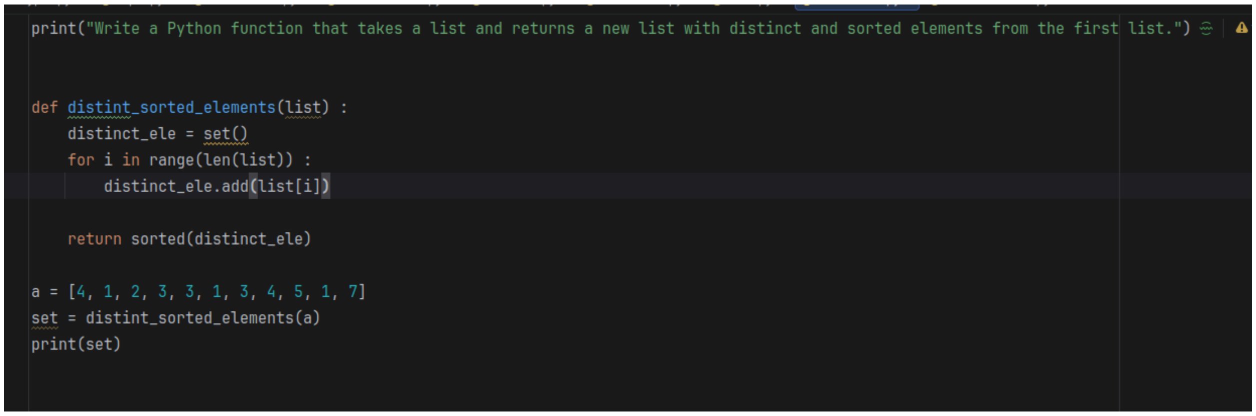


```
print("Write a function that takes one parameter as a string and reverse it and return.")

def reverse_string(string) :
    reversed_string = ''

    for char in string :
        reversed_string = char + reversed_string
    return reversed_string

print(reverse_string("Kiran"))
```

4. Write a Python function that takes two parameters as lists and to sum all the numbers in a list.
   a = [8, 2, 3, 0, 7], b = [3, -2, 5, 1] and return a value.


```python
print("Write a Python function that takes two parameters as lists and to sum all the numbers in a list. ")

def sum(list1, list2) :
    sum = 0

    for i in range(len(list1)) :
        sum += list1[i]

    for j in range(len(list2)) :
        sum += list2[j]

    return sum

a = [8, 2, 3, 0, 7]
b =  [3, -2, 5, 1]
result = sum(a, b)
print(result)
```

5. Write a Python function that takes a list and returns a new list with distinct and sorted elements from the first list. a = [4,1,2,3,3,1,3,4,5,1,7]
Output = [1,2,3,4,5,7]

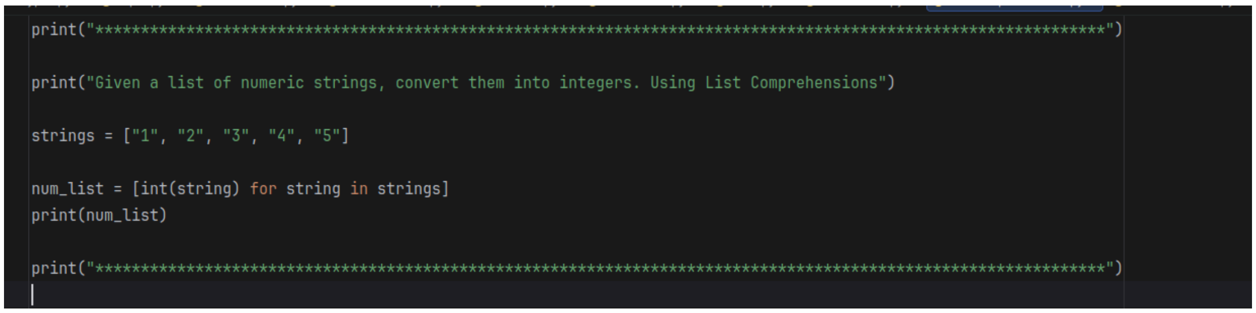

```python
print("Write a Python function that takes a list and returns a new list with distinct and sorted elements from the first list.")


def distint_sorted_elements(list) :
    distinct_ele = set()
    for i in range(len(list)) :
        distinct_ele.add(list[i])

    return sorted(distinct_ele)

a = [4, 1, 2, 3, 3, 1, 3, 4, 5, 1, 7]
set = distint_sorted_elements(a)
print(set)
```

LIST COMPREHENSIONS

LIST COMPREHENSIONS

1. Given a list of numeric strings, convert them into integers. Using List Comprehensions
   strings = ["1", "2", "3", "4", "5"]

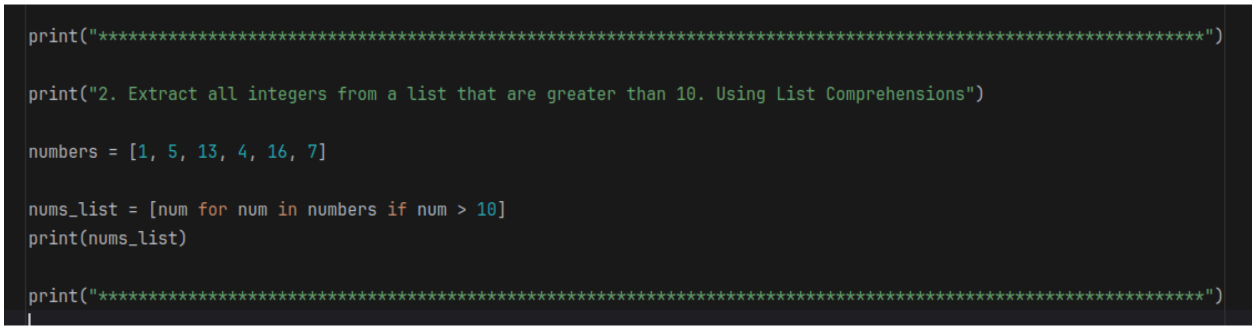
   #Expected output : [1, 2, 3, 4, 5]



```python
print("****************************************************************************************************")

print("Given a list of numeric strings, convert them into integers. Using List Comprehensions")

strings = ["1", "2", "3", "4", "5"]

num_list = [int(string) for string in strings]
print(num_list)

print("****************************************************************************************************")
```

2. Extract all integers from a list that are greater than 10. Using List Comprehensions
   numbers = [1, 5, 13, 4, 16, 7]

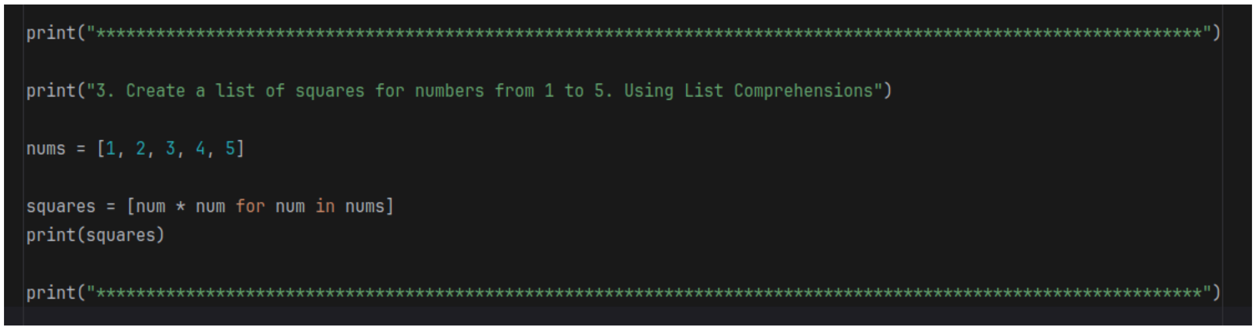
   #Expected output :[13, 16]



```python
print("****************************************************************************************************")

print("2. Extract all integers from a list that are greater than 10. Using List Comprehensions")

numbers = [1, 5, 13, 4, 16, 7]

nums_list = [num for num in numbers if num > 10]
print(nums_list)

print("****************************************************************************************************")
```

3. Create a list of squares for numbers from 1 to 5. Using List Comprehensions

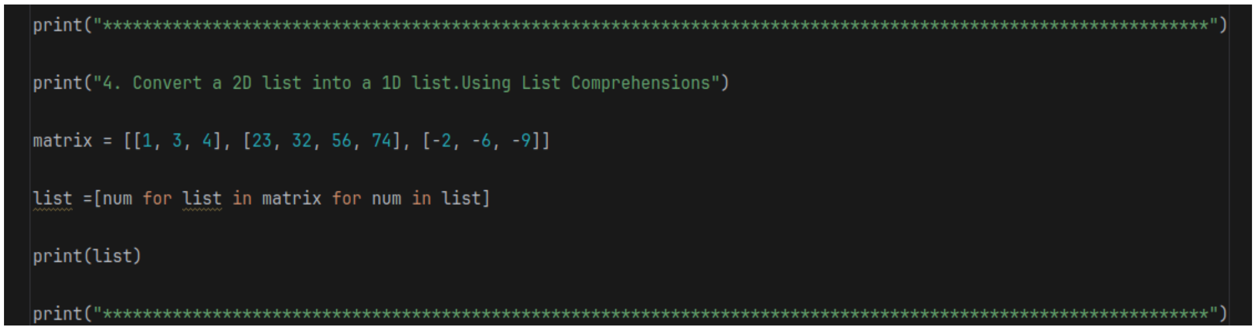
   #Expected output :[1, 4, 9, 16, 25]



```python
print("****************************************************************************************************")

print("3. Create a list of squares for numbers from 1 to 5. Using List Comprehensions")

nums = [1, 2, 3, 4, 5]

squares = [num * num for num in nums]
print(squares)

print("****************************************************************************************************")
```

4. Convert a 2D list into a 1D list.Using List Comprehensions
   matrix = [[1, 3, 4], [23, 32, 56, 74], [-2, -6, -9]]

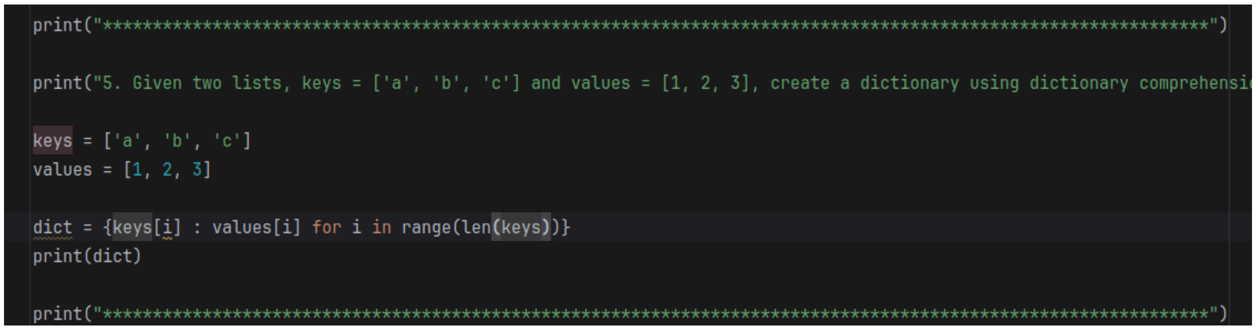
   #Expected output : [1, 3, 4, 23, 32, 56, 74, -2, -6, -9]



```
print("**************************************************************************************************")

print("4. Convert a 2D list into a 1D list.Using List Comprehensions")

matrix = [[1, 3, 4], [23, 32, 56, 74], [-2, -6, -9]]

list =[num for list in matrix for num in list]

print(list)

print("**************************************************************************************************")
```

5. Given two lists, `keys = ['a', 'b', 'c']` and `values = [1, 2, 3]`, create a dictionary using dictionary comprehension.

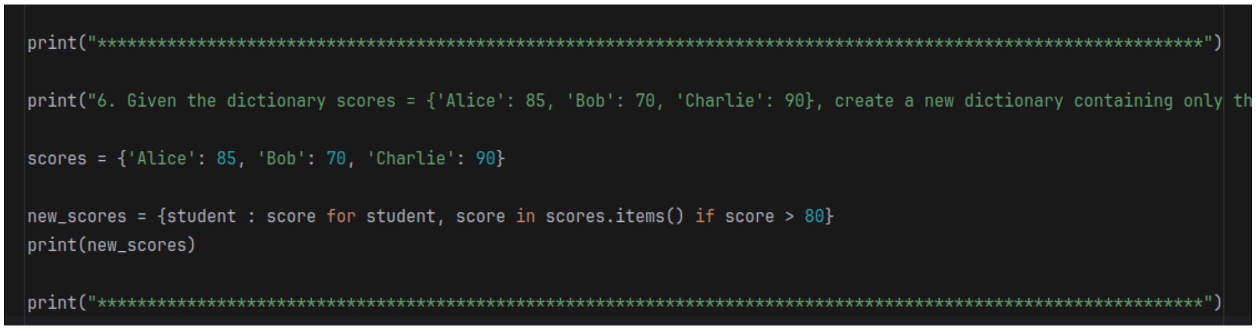
   #Expected output : {'a': 1, 'b': 2, 'c': 3}



```
print("**************************************************************************************************")

print("5. Given two lists, keys = ['a', 'b', 'c'] and values = [1, 2, 3], create a dictionary using dictionary comprehensi

keys = ['a', 'b', 'c']
values = [1, 2, 3]

dict = {keys[i] : values[i] for i in range(len(keys))}
print(dict)

print("**************************************************************************************************")
```

6. Given the dictionary `scores = {'Alice': 85, 'Bob': 70, 'Charlie': 90}`, create a new dictionary containing only the students who scored above 80

   #Expected output : {'Alice': 85, 'Charlie': 90}



```
print("**************************************************************************************************")

print("6. Given the dictionary scores = {'Alice': 85, 'Bob': 70, 'Charlie': 90}, create a new dictionary containing only th

scores = {'Alice': 85, 'Bob': 70, 'Charlie': 90}

new_scores = {student : score for student, score in scores.items() if score > 80}
print(new_scores)

print("**************************************************************************************************")
```