# Code cell 1

print(4.5 \* 1.5 - 2.5)

# Code cell 2

radius = eval(input("Enter a value for radius: "))

area = 3.14 \* radius \* radius

print(area)

‘’’

In this code, eval() is a built-in Python function used to evaluate the expression passed to it as a string and returns the result.

Here's how it works in the context of your code:

input("Enter a value for radius: ")

prompts the user to enter a value for the radius.

Whatever the user inputs is treated as a string.

eval() takes this string and evaluates it as a Python expression.

If the user enters, for example, "5",

eval() will evaluate it to the integer 5.

The result of eval() is then assigned to the variable radius.

So, eval() here is used to convert the user input (which is a string)

into a numerical value (specifically, an integer or float), which can then be used for calculations like finding the area of a circle.

’’’

# Code cell 3

radius = eval(input("Enter a value for radius: "))

if radius > 0:

if radius < 10:

print("Number is positive and less than 10")

else:

print("Number is negative")

print("Operation completed")

# Code cell 4

score = eval(input("Enter the score: "))

if score >= 90.0:

grade = 'A'

elif score >= 80.0:

grade = 'B'

elif score >= 70.0:

grade = 'C'

elif score >= 60.0:

grade = 'D'

else:

grade = 'F'

print(grade)

# Code cell 1

sum\_val = 0

counter = 0

while counter < 10:

sum\_val += counter

counter += 1

print("Sum of numbers from 0 to 9 = ", sum\_val)

# Code cell 2

start\_value = eval(input("Enter the start value: "))

start\_value\_original = start\_value

end\_value = eval(input("Enter the end value: "))

end\_value\_original = end\_value

if start\_value < end\_value:

sum\_val = 0

while start\_value < end\_value:

sum\_val += start\_value

start\_value += 1

else:

temp = end\_value

end\_value = start\_value

start\_value = temp

sum\_val = 0

while start\_value < end\_value:

sum\_val += start\_value

start\_value += 1

print(start\_value\_original, " to ", end\_value\_original,

" sum of numbers = ", sum\_val)

# Code cell 3

import random

number1 = random.randint(0, 10)

number2 = random.randint(0, 10)

result = number1 + number2

print(number1, "+", number2, " = ?")

guess = eval(input("Answer: "))

if result == guess:

print("Congratulations!")

else:

print("Wrong answer")

# Code cell 4

for i in range(0, 1, 2):

print(i)

‘’’

The range() function is called with three arguments:

start (1 in this case): The starting number of the sequence (inclusive).

stop (2 in this case): The ending number of the sequence (exclusive).

step (3 in this case): The difference between each two consecutive numbers in the sequence.

However, the sequence generated by this range()

call doesn't produce any numbers because the start value is greater than or equal to the stop value.

So, when you execute this loop, it doesn't print anything because there are no values generated by the range() function to iterate over. In essence, it generates an empty sequence, resulting in no output.

‘’’

# Code cell 5

for i in range(1, 6):

for j in range(1, 6):

print("Nested Loop")

# Code cell 6

for i in range(1, 100):

if i == 50:

break

print(i)

# Code cell 7

for i in range(1, 100):

if i == 50:

continue

print(i)

# Code cell 1

def sum(a, b):

return a + b

def printFunction():

print("Message")

print("Result= ", sum(10, 10))

print("Result= ", printFunction())

# Code cell 2

def sum(s1, s2):

sum\_val = 0

for i in range(s1, s2 + 1):

sum\_val += i

return sum\_val

def main():

s1 = eval(input("Enter the first number: "))

s2 = eval(input("Enter the second number: "))

print("Sum of the entered numbers = ", sum(s1, s2))

main()

# Code cell 3

def sumDigit(number):

sum\_val = 0

while number != 0:

remainder = number % 10

sum\_val += remainder

number -= remainder

number /= 10

return sum\_val

def main():

number = eval(input("Enter a number: "))

print("Sum of digits of the entered number = ", int(sumDigit(number)))

main()

# Code cell 4

def sumDigit(number):

original\_number = number

sum\_val = 0

while number != 0:

remainder = number % 10

sum\_val += remainder

number -= remainder

number /= 10

print("Sum of digits of the entered number", original\_number, "= ", sum\_val)

def main():

number = eval(input("Enter a number: "))

sumDigit(number)

main()

★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★

# Code cell 5

import random

def createArray(size):

print("Creating an array...")

array = []

for i in range(size):

array.append(random.randint(0, 10))

print(array)

return array

def printArray(array):

print("Printing the array")

for i in range(len(array)):

print(array[i], end=" ")

print()

def sumArray(array):

print("Performing operations on the array...")

sum\_val = 0

for i in range(len(array)):

sum\_val += array[i]

return sum\_val

def main():

size = eval(input("Enter the size of the array: "))

array = createArray(size)

printArray(array)

print("Sum of array elements = ", sumArray(array))

main()

‘’’

In the given code, len() is a built-in Python function used to return the number of elements in an object. Here's how it works in the context of your code:

len(array) is used inside the printArray() and sumArray() functions to determine the length of the input array.

In printArray(), len(array) returns the number of elements in the array. This value is used in the for loop to iterate over each element of the array.

In sumArray(), len(array) similarly returns the number of elements in the array. This value is used in the for loop to iterate over each element of the array and calculate their sum.

So, in summary, len(array) is used to find the number of elements in the array array, which is crucial for iterating over the elements in the array in both the printing and summing functions.

‘’’

★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★

# Code cell 6

import random

def createArray(size):

print("Creating an array...")

array = []

for i in range(size):

array.append(random.randint(0, 10))

print(array)

return array

def printArray(array):

print("Printing the array")

for i in range(len(array)):

print(array[i], end=" ")

print()

sumArray(array)

def sumArray(array):

print("Performing operations on the array...")

sum\_val = 0

for i in range(len(array)):

sum\_val += array[i]

print("Sum of array elements = ", sum\_val)

def main():

size = eval(input("Enter the size of the array: "))

printArray(createArray(size))

main()

# Code cell 7

import random

def createMatrix():

print("Creating a matrix")

rows = eval(input("Enter the number of rows: "))

columns = eval(input("Enter the number of columns: "))

matrix = []

for row in range(rows):

matrix.append([])

for column in range(columns):

matrix[row].append(random.randint(0, 10))

print("Matrix created")

print(matrix)

return matrix

def printMatrix(matrix):

print("Printing the matrix...")

for row in range(len(matrix)):

for column in range(len(matrix[row])):

print(matrix[row][column], end=" ")

print()

def calculateMatrix(matrix):

print("Performing operations on the matrix...")

sum\_odd = 0

sum\_even = 0

for row in range(len(matrix)):

for column in range(len(matrix[row])):

if matrix[row][column] % 2 == 0:

sum\_even += matrix[row][column]

else:

sum\_odd += matrix[row][column]

return sum\_odd, sum\_even

def main():

matrix = createMatrix()

printMatrix(matrix)

odd\_sum, even\_sum = calculateMatrix(matrix)

print("Sum of odd numbers = ", odd\_sum)

print("Sum of even numbers = ", even\_sum)

main()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import random

def sum(a, b):

return a + b

def printFunction():

print("Message")

print("Result= ", sum(10, 10))

print("Result= ", printFunction())

def sum(s1, s2):

sum\_val = 0

for i in range(s1, s2 + 1):

sum\_val += i

return sum\_val

def main():

s1 = eval(input("Enter the first number: "))

s2 = eval(input("Enter the second number: "))

print("Sum of the entered numbers = ", sum(s1, s2))

main()

def sumDigit(number):

sum\_val = 0

while number != 0:

remainder = number % 10

sum\_val += remainder

number -= remainder

number /= 10

return sum\_val

def main():

number = eval(input("Enter a number: "))

print("Sum of digits of the entered number = ", int(sumDigit(number)))

main()

def sumDigit(number):

original\_number = number

sum\_val = 0

while number != 0:

remainder = number % 10

sum\_val += remainder

number -= remainder

number /= 10

print("Sum of digits of the entered number", original\_number, "= ", sum\_val)

def main():

number = eval(input("Enter a number: "))

sumDigit(number)

main()

def createArray(size):

print("Creating an array...")

array = []

for i in range(size):

array.append(random.randint(0, 10))

print(array)

return array

def printArray(array):

print("Printing the array")

for i in range(len(array)):

print(array[i], end=" ")

print()

def sumArray(array):

print("Performing operations on the array...")

sum\_val = 0

for i in range(len(array)):

sum\_val += array[i]

return sum\_val

def main():

size = eval(input("Enter the size of the array: "))

array = createArray(size)

printArray(array)

print("Sum of array elements = ", sumArray(array))

main()

def createArray(size):

print("Creating an array...")

array = []

for i in range(size):

array.append(random.randint(0, 10))

print(array)

return array

def printArray(array):

print("Printing the array")

for i in range(len(array)):

print(array[i], end=" ")

print()

sumArray(array)

def sumArray(array):

print("Performing operations on the array...")

sum\_val = 0

for i in range(len(array)):

sum\_val += array[i]

print("Sum of array elements = ", sum\_val)

def main():

size = eval(input("Enter the size of the array: "))

printArray(createArray(size))

main()

def createMatrix():

print("Creating a matrix")

rows = eval(input("Enter the number of rows: "))

columns = eval(input("Enter the number of columns: "))

matrix = []

for row in range(rows):

matrix.append([])

for column in range(columns):

matrix[row].append(random.randint(0, 10))

print("Matrix created")

print(matrix)

return matrix

def printMatrix(matrix):

print("Printing the matrix...")

for row in range(len(matrix)):

for column in range(len(matrix[row])):

print(matrix[row][column], end=" ")

print()

def calculateMatrix(matrix):

print("Performing operations on the matrix...")

sum\_odd = 0

sum\_even = 0

for row in range(len(matrix)):

for column in range(len(matrix[row])):

if matrix[row][column] % 2 == 0:

sum\_even += matrix[row][column]

else:

sum\_odd += matrix[row][column]

return sum\_odd, sum\_even

def main():

matrix = createMatrix()

printMatrix(matrix)

odd\_sum, even\_sum = calculateMatrix(matrix)

print("Sum of odd numbers = ", odd\_sum)

print("Sum of even numbers = ", even\_sum)

main()

# Cell 1

print("Turkish control")

result = 4.5 \* 5 - 2 - 6

print("Turkish control")

# Cell 2

number = eval(input("Please enter a number: "))

area = 3.14 \* number \* number

print("Radius of the circle: ", number, " Area: ", area)

# Cell 3

base = eval(input("Enter the base value: "))

exponent = eval(input("Enter the exponent value: "))

print(base, "^", exponent, "= ", (base \*\* exponent))

# Cell 4

print(3 + 4 \* 4 + 5 \* (4 + 3) - 1)

# Cell 5

number = eval(input("Enter a number: "))

if number % 2 == 0:

if number < 100:

print("The number is even and less than 100")

print("Operation Completed")

# Cell 6

number = eval(input("Enter a number: "))

if number % 2 == 0:

print("The number is even and less than 100")

else:

print('The number is odd')

print("Operation Completed")

# Cell 7

score = eval(input('Enter the score: '))

if score >= 90.0:

grade = 'A'

elif score >= 80.0:

grade = 'B'

elif score >= 70.0:

grade = 'C'

elif score >= 60.0:

grade = 'D'

else:

grade = 'F'

print(grade)

# Cell 1

sum = 0

counter = 0

while counter < 10:

sum += counter

counter += 1

print("Sum of numbers between 0 and 10 = ", str(sum))

‘’’’

In the provided code, the str() function converts the integer value of sum to a string before it is passed as an argument to the print() function.

Although in this specific case, converting sum to a string using str() is not necessary because the print() function automatically converts its arguments to strings before printing them. However, adding str() ensures that the conversion is explicit, which can sometimes be useful for clarity or when dealing with more complex expressions.

So, in this context, the str() function doesn't have any significant impact on the code's functionality. It simply converts the integer sum to a string, making it suitable for printing.

‘’’’

# Cell 2

start\_value = eval(input("Enter the start value: "))

start\_value\_fixed = start\_value

end\_value = eval(input("Enter the end value: "))

if start\_value < end\_value:

sum = 0

while start\_value < end\_value:

sum += start\_value

start\_value += 1

else:

temp = end\_value

end\_value = start\_value

start\_value = temp

sum = 0

while start\_value < end\_value:

sum += start\_value

start\_value += 1

print(start\_value\_fixed, " to ", end\_value, " sum of numbers = ", sum)

‘’’

This code seems to be intended to calculate the sum of numbers between two given values (inclusive of the start value and exclusive of the end value). Here's an analysis of the code:

Input Gathering: The code first prompts the user to enter the start and end values. It uses the input() function to get user input and eval() to evaluate the input as Python code, converting it into a numerical value.

Condition Check: It checks if the start value is less than the end value. If it is, it proceeds with the calculation directly. Otherwise, it swaps the values so that the start value is always less than the end value.

Summation Calculation: It initializes a variable sum to store the sum of numbers. Then, it enters a while loop where it iterates from the start value to the end value (exclusive) and accumulates the sum of each number.

Output: Finally, it prints the start value, end value, and the sum of numbers between them.

While the code achieves its goal of calculating the sum of numbers between two given values, there are a few potential improvements:

It's generally recommended to avoid using eval() as it can be unsafe and inefficient. Instead, you can use int() to convert the user input directly to integers.

The code could be simplified by avoiding the need to swap values. Instead, you could initialize start\_value as the minimum of the two inputs and end\_value as the maximum.

Using more descriptive variable names can improve code readability.

Here's an updated version of the code incorporating these suggestions:

The Fixed code.1:

start\_value = int(input("Enter the start value: "))

end\_value = int(input("Enter the end value: "))

start = min(start\_value, end\_value)

end = max(start\_value, end\_value)

sum\_of\_numbers = 0

while start < end:

sum\_of\_numbers += start

start += 1

print(start\_value, " to ", end\_value, " sum of numbers = ", sum\_of\_numbers)

The fixed code.2:

start\_value = eval(input("Enter the start value: "))

start\_value\_fixed = start\_value

end\_value = eval(input("Enter the end value: "))

end\_value\_fixed = end\_value

if start\_value <= end\_value:

sum = 0

while start\_value <= end\_value:

sum += start\_value

start\_value += 1

else:

temp = end\_value

end\_value = start\_value

start\_value = temp

sum = 0

while start\_value <= end\_value:

sum += start\_value

start\_value += 1

print(start\_value\_fixed, " to ", end\_value\_fixed, " sum of numbers = ", sum)

‘’’

# Cell 3

import random

number1 = random.randint(0, 10)

number2 = random.randint(0, 10)

if number1 < number2:

print(number2, "-", number1, "=?")

answer = eval(input("Your answer: "))

result = number2 - number1

else:

print(number1, "-", number2, "=?")

answer = eval(input("Your answer: "))

result = number1 - number2

if answer == result:

print("Congratulations!")

else:

print("Wrong Answer")

# Cell 4

for i in range(1, 6):

print("Nested Loop")

‘’’

for i in range(1, 6):

print("Nested Loop",i)

‘’’

# Cell 5

for i in range(1, 5):

j = 0

while j < i:

print(j, end="")

j += 1

print()

‘’’

In Python, the print() function is used to output text or values to the console. By default, each call to print() will print its arguments followed by a newline character (\n), which moves the cursor to the next line after printing.

However, the print() function also has optional parameters that allow you to modify its behavior. One of these parameters is end. The end parameter specifies what to print at the end of the output. By default, end='\n', meaning that a newline character is printed at the end.

In the code you provided:

python

Copy code

for i in range(1, 5):

j = 0

while j < i:

print(j, end="")

j += 1

print()

The end="" part is used to specify that nothing should be printed at the end of each print() statement within the inner while loop. Instead of automatically moving to the next line, the cursor will stay at the end of the printed output. This means that subsequent calls to print() will continue printing on the same line.

So, the output of each iteration of the inner loop will be printed side by side, without a newline character separating them. After the inner loop finishes printing the numbers, print() without any arguments is called, which moves to the next line, ensuring that each iteration of the outer loop starts on a new line.

Here's how the output of the code looks like:

yaml

Copy code

0

01

012

0123

Each number sequence is printed on a new line due to the print() statement outside the inner loop. The end="" part ensures that the numbers within each line are printed without any separation or newline characters between them.

SECOND EXPLANATION

Outer Loop (for i in range(1, 5):):

This loop iterates over the numbers 1 through 4. For each iteration:

i takes on the values 1, 2, 3, and 4 successively.

Inner Loop (while j < i:):

Inside the outer loop, an inner loop is used. This loop runs j from 0 up to the current value of i.

For example, in the first iteration of the outer loop (when i is 1), the inner loop runs once with j being 0. In the second iteration (when i is 2), the inner loop runs twice with j being 0 and 1, and so on.

Printing (print(j, end="")):

Inside the inner loop, print(j, end="") prints the current value of j without adding a newline character (end="").

So, during each iteration of the inner loop, the current value of j is printed without any newline characters separating them.

Newline (print()):

After the inner loop finishes executing, a print() statement without any arguments is used to print an empty line, which effectively adds a newline character. This moves the cursor to the next line before the next iteration of the outer loop.

Now, let's see how this unfolds with the first few iterations of the outer loop:

When i is 1:

The inner loop runs once with j being 0, printing "0" without a newline character.

After the inner loop, a newline character is printed, moving to the next line.

When i is 2:

The inner loop runs twice with j being 0 and 1, printing "01" without a newline character.

After the inner loop, a newline character is printed, moving to the next line.

When i is 3:

The inner loop runs three times with j being 0, 1, and 2, printing "012" without a newline character.

After the inner loop, a newline character is printed, moving to the next line.

When i is 4:

The inner loop runs four times with j being 0, 1, 2, and 3, printing "0123" without a newline character.

After the inner loop, a newline character is printed, moving to the next line.

This process continues until the outer loop completes all iterations, resulting in the output:

yaml

Copy code

0

01

012

0123

Each line contains the concatenated sequence of numbers printed without any newline characters between them. The newline characters are added by the print() statement without arguments after each inner loop.

‘’’

# Cell 6

for i in range(1, 100):

if i == 50:

continue

print(i)

# Cell 1

sum = 0

counter = 0

while counter < 10:

sum += counter

counter += 1

print("Sum of numbers between 0 and 10 = ", str(sum))

# Cell 2

start\_value = eval(input("Enter the start value: "))

start\_value\_fixed = start\_value

end\_value = eval(input("Enter the end value: "))

if start\_value < end\_value:

sum = 0

while start\_value < end\_value:

sum += start\_value

start\_value += 1

else:

temp = end\_value

end\_value = start\_value

start\_value = temp

sum = 0

while start\_value < end\_value:

sum += start\_value

start\_value += 1

print(start\_value\_fixed, " to ", end\_value, " sum of numbers = ", sum)

# Cell 3

import random

number1 = random.randint(0, 10)

number2 = random.randint(0, 10)

print(number1, "-", number2, "=?")

result = eval(input("Your answer = ? "))

result\_sub = number1 - number2

if result == result\_sub:

print("Congratulations!")

else:

print("Wrong")

‘’’

es, you can modify the code to generate float numbers by using random.uniform() instead of random.randint(). random.uniform(a, b) generates a random floating-point number N such that a <= N <= b.

Here's the modified code:

python

Copy code

import random

# Generate random floating-point numbers between 0 and 10

number1 = random.uniform(0, 10)

number2 = random.uniform(0, 10)

# Round the numbers to two decimal places for better readability

number1 = round(number1, 2)

number2 = round(number2, 2)

print(number1, "-", number2, "=?")

result = float(input("Your answer = ? "))

result\_sub = round(number1 - number2, 2) # Round the result to two decimal places

if result == result\_sub:

print("Congratulations!")

else:

print("Wrong")

In this modified version:

random.uniform(0, 10) is used to generate random floating-point numbers between 0 and 10.

The round() function is used to round the generated numbers to two decimal places for better readability and comparison accuracy.

The input() function returns a string, which is converted to a float using float().

The result\_sub calculation also rounds the result to two decimal places for comparison with the user's input.

‘’’

# Cell 4

import random

correct\_count = 0

question\_count = 3

counter = 0

while counter < question\_count:

number1 = random.randint(0, 10)

number2 = random.randint(0, 10)

print(number1, "-", number2, "=?")

result = eval(input("Your answer = ? "))

result\_sub = number1 - number2

if result == result\_sub:

print("Congratulations!")

correct\_count += 1

else:

print("Wrong")

counter += 1

print(correct\_count, "/", question\_count)

‘’’

WITH LETTER GRADE

import random

correct\_count = 0

question\_count = 3

counter = 0

while counter < question\_count:

number1 = random.randint(0, 10)

number2 = random.randint(0, 10)

print(number1, "-", number2, "=?")

result = eval(input("Your answer = ? "))

result\_sub = number1 - number2

if result == result\_sub:

print("Congratulations!")

correct\_count += 1

else:

print("Wrong")

counter += 1

percentage\_correct = round(float((correct\_count / question\_count) \* 100))

# Determine letter grade based on percentage

if percentage\_correct >= 90:

letter\_grade = 'A'

elif percentage\_correct >= 80:

letter\_grade = 'B'

elif percentage\_correct >= 70:

letter\_grade = 'C'

elif percentage\_correct >= 60:

letter\_grade = 'D'

else:

letter\_grade = 'F'

print("Correct answers:", correct\_count, "/", question\_count)

print("Percentage:", percentage\_correct, "%")

print("Letter grade:", letter\_grade)

‘’’

# Cell 5

for i in range(10):

print(i)

# Cell 6

for i in range(1, 6):

for j in range(1, 6):

print("Nested Loops")

‘’’

count = 0

for i in range(1, 6):

for j in range(1, 6):

count += 1

print("Nested Loops",count)

‘’’

# Cell 7

for i in range(1, 5):

j = 0

while j < i:

print(j, end=" ")

j += 1

print()

# Cell 8

# This code will produce a TypeError because the range() function cannot accept floats as step values

# Fixing it by using integers for the step value

for i in range(0, 10, 1):

print(i)

# Cell 9

# The code in this cell will produce an infinite loop because the break condition is never met

# Fixing it by adding a break statement when i reaches 50

for i in range(0, 100):

if i == 50:

break

print(i)

# Cell 10

# The code in this cell will skip printing 50 because of the continue statement

# Fixing it by removing the continue statement

for i in range(0, 100):

if i == 50:

continue

print(i)

import random

def create\_list\_with\_user\_input():

dizi = []

diziElemanSayisi = int(input("Enter the size of the list: "))

sum = 0

for i in range(diziElemanSayisi):

value = int(input(f"Enter element {i+1} of the list: "))

dizi.append(value)

sum += dizi[i]

avg = sum / diziElemanSayisi

print("List:", dizi)

print("Sum of list elements:", sum)

print("Average of list elements:", avg)

print("Last element of the list:", dizi[-1])

def create\_list\_with\_random\_numbers():

dizi = []

diziElemanSayisi = int(input("Enter the size of the list: "))

for i in range(diziElemanSayisi):

value = random.randint(0, 50)

dizi.append(value)

print("Random list:", dizi)

def find\_smallest\_and\_largest():

dizi = []

diziElemanSayisi = int(input("Enter the size of the list: "))

for i in range(diziElemanSayisi):

value = random.randint(0, 50)

dizi.append(value)

enKucuk = dizi[0]

enBuyuk = dizi[0]

for i in range(diziElemanSayisi):

if dizi[i] < enKucuk:

enKucuk = dizi[i]

if dizi[i] > enBuyuk:

enBuyuk = dizi[i]

print("List:", dizi)

print("Smallest element in the list:", enKucuk)

print("Largest element in the list:", enBuyuk)

def sort\_list():

liste = []

listeElemanSayisi = int(input("Enter the size of the list: "))

for i in range(listeElemanSayisi):

value = random.randint(0, 50)

liste.append(value)

print("Original list:", liste)

print("Sorting the list in ascending order...")

liste.sort()

print("Sorted list in ascending order:", liste)

print("Sorting the list in descending order...")

liste.sort(reverse=True)

print("Sorted list in descending order:", liste)

def create\_matrix\_with\_user\_input():

matrix = []

rowNumber = int(input("Enter the number of rows: "))

columnNumber = int(input("Enter the number of columns: "))

for row in range(rowNumber):

matrix.append([])

for column in range(columnNumber):

value = int(input(f"Enter element at position ({row+1}, {column+1}): "))

matrix[row].append(value)

print("Matrix:", matrix)

def create\_matrix\_with\_random\_numbers():

matrix = []

rowNumber = int(input("Enter the number of rows: "))

columnNumber = int(input("Enter the number of columns: "))

print("Creating a matrix with random numbers...")

for row in range(rowNumber):

matrix.append([])

for column in range(columnNumber):

value = random.randint(0, 9)

matrix[row].append(value)

print("Random matrix:", matrix)

if \_\_name\_\_ == "\_\_main\_\_":

create\_list\_with\_user\_input()

create\_list\_with\_random\_numbers()

find\_smallest\_and\_largest()

sort\_list()

create\_matrix\_with\_user\_input()

create\_matrix\_with\_random\_numbers()

EXAMPLE RESULT:

Enter the size of the list: 3

Enter element 1 of the list: 1

Enter element 2 of the list: 2

Enter element 3 of the list: 3

Enter the size of the list: 3

List: [1, 2, 3]

Sum of list elements: 6

Average of list elements: 2.0

Last element of the list: 3

Random list: [31, 23, 49]

Enter the size of the list: 3

List: [17, 4, 31]

Smallest element in the list: 4

Largest element in the list: 31

Enter the size of the list: 3

Original list: [36, 21, 3]

Sorting the list in ascending order...

Sorted list in ascending order: [3, 21, 36]

Sorting the list in descending order...

Sorted list in descending order: [36, 21, 3]

Enter the number of rows: 3

Enter the number of columns: 3

Enter element at position (1, 1): 1

Enter element at position (1, 2): 2

Enter element at position (1, 3): 3

Enter element at position (2, 1): 1

Enter element at position (2, 2): 2

Enter element at position (2, 3): 3

Enter element at position (3, 1): 1

Enter element at position (3, 2): 2

Enter element at position (3, 3): 3

Matrix: [[1, 2, 3], [1, 2, 3], [1, 2, 3]]

Enter the number of rows: 3

Enter the number of columns: 3

Creating a matrix with random numbers...

Random matrix: [[0, 2, 5], [1, 9, 7], [3, 0, 0]]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This code defines several functions to perform various tasks related to lists and matrices, and then executes these functions in the `\_\_main\_\_` block. Here's an analysis of each function:

1. \*\*create\_list\_with\_user\_input():\*\*

- This function prompts the user to input the size of the list and the elements of the list. It calculates the sum, average, and prints the list, sum, average, and last element of the list.

2. \*\*create\_list\_with\_random\_numbers():\*\*

- This function prompts the user to input the size of the list and generates a list of random numbers between 0 and 50. It then prints the random list.

3. \*\*find\_smallest\_and\_largest():\*\*

- This function prompts the user to input the size of the list and generates a list of random numbers between 0 and 50. It then finds the smallest and largest elements in the list and prints them.

4. \*\*sort\_list():\*\*

- This function prompts the user to input the size of the list and generates a list of random numbers between 0 and 50. It then sorts the list in ascending and descending order and prints both sorted lists.

5. \*\*create\_matrix\_with\_user\_input():\*\*

- This function prompts the user to input the number of rows and columns of the matrix and the elements of the matrix. It then prints the matrix.

6. \*\*create\_matrix\_with\_random\_numbers():\*\*

- This function prompts the user to input the number of rows and columns of the matrix and generates a matrix with random numbers between 0 and 9. It then prints the random matrix.

The code is structured to allow the user to interactively create lists and matrices either by providing input or by generating random numbers, and to perform various operations on them, such as finding the smallest and largest elements, sorting lists, and creating matrices.

# Example 1

liste=[]

listeBoyutu=eval(input("Liste boyutunu giriniz : "))

for i in range(listeBoyutu):

print((i+1),". ", end=" ")

value=eval(input("Liste elemanini giriniz : "))

liste.append(value)

liste.append(10)

print(liste)

‘’’

liste = []

listeBoyutu = int(input("Liste boyutunu giriniz: "))

for i in range(listeBoyutu):

value = int(input(f"Girilen listenin {i+1}. elemanını giriniz: "))

liste.append(value)

# Append 10 after all user inputs (this line is outside the loop)

liste.append(10)

print("Son liste:", liste)

‘’’

# Example 2

import random

liste=[]

listeBoyutu=eval(input("Liste boyutunu giriniz : "))

sum=0

for i in range(listeBoyutu):

value=random.randint(0,10)

sum+=value

liste.append(value)

avg=sum/listeBoyutu

print(liste)

print("Liste elemanlarinin toplami = ", sum)

print("Liste elemanlarinin ortalamasi = ", avg)

kucukSay=0

buyukSay=0

for i in range(listeBoyutu):

if liste[i]<avg:

kucukSay+=1

else:

buyukSay+=1

print("Ortalamanin altinda kalan eleman sayisi = ", kucukSay)

print("Ortalamanin ustunde kalan eleman sayisi = ", buyukSay)

‘’’

FIRST KIND OF IMPROVE:

import random

from collections import Counter

listeBoyutu = int(input("Liste boyutunu giriniz: "))

# Generate a list of random integers

liste = [random.randint(0, 10) for \_ in range(listeBoyutu)]

# Calculate sum and average of the list

liste\_sum = 0

for value in liste:

liste\_sum += value # Add each element to the sum

liste\_avg = liste\_sum / listeBoyutu

# Count occurrences of elements below and above the average

counter = Counter(x < liste\_avg for x in liste)

kucukSay = counter[True]

buyukSay = counter[False]

# Print the results

print("Liste:", liste)

print("Liste elemanlarının toplamı =", liste\_sum)

print("Liste elemanlarının ortalaması =", liste\_avg)

print("Ortalamının altında kalan eleman sayısı =", kucukSay)

print("Ortalamının üstünde kalan eleman sayısı =", buyukSay)

SECOND KIND OF IMPROVE:

import random

liste = []

listeBoyutu = int(input("Liste boyutunu giriniz: "))

# Generate list of random numbers and calculate sum

liste = [random.randint(0, 10) for \_ in range(listeBoyutu)]

liste\_sum = 0 # Initialize the sum variable to 0

for value in liste:

liste\_sum += value # Add each element to the sum

# Calculate average

avg = liste\_sum / listeBoyutu

print("Liste:", liste)

print("Liste elemanlarının toplamı:", liste\_sum)

print("Liste elemanlarının ortalaması:", avg)

# Count elements below and above average

kucukSay = len([1 for value in liste if value < avg])

buyukSay = len([1 for value in liste if value > avg])

print("Ortalamadan küçük eleman sayısı:", kucukSay)

print("Ortalamadan büyük eleman sayısı:", buyukSay)

del avg

‘’’

# Example 3

import random

liste=[]

listeBoyutu=eval(input("Liste boyutunu giriniz : "))

print("Liste olusturuluyor...")

for i in range(listeBoyutu):

value=random.randint(0,100)

liste.append(value)

print("Liste yazdiriliyor...")

for i in range(listeBoyutu):

print(liste[i], end=" ")

print()

enKucuk=liste[0]

enBuyuk=liste[0]

for i in range(len(liste)):

if (liste[i]<enKucuk):

enKucuk=liste[i]

if (liste[i]>enBuyuk):

enBuyuk =liste[i]

print("Listenin en kucuk elemani =", enKucuk)

print("Listenin en buyuk elemani =", enBuyuk)

# Example 4

import random

liste=[]

listeBoyutu=eval(input("Liste boyutunu giriniz : "))

print("Liste olusturuluyor...")

for i in range(listeBoyutu):

value=random.randint(0,100)

liste.append(value)

print("Liste yazdiriliyor...")

for i in range(listeBoyutu):

print(liste[i], end=" ")

print()

print("Liste elemanlari kucukten buyuge siralaniyor....")

for i in range(len(liste)):

for j in range(len(liste)):

if (liste[i]<liste[j]):

temp=liste[j]

liste[j]=liste[i]

liste[i]=temp

print(liste)

print("Liste elemanlari buyukten kucuge siralaniyor....")

for i in range(len(liste)):

for j in range(len(liste)):

if (liste[i]>liste[j]):

temp=liste[j]

liste[j]=liste[i]

liste[i]=temp

print(liste)

‘’’

import random

liste=[]

listeBoyutu=eval(input("Liste boyutunu giriniz : "))

print("Liste olusturuluyor...")

for i in range(listeBoyutu):

value=random.randint(0,100)

liste.append(value)

print("Liste yazdiriliyor...")

for i in range(listeBoyutu):

print(liste[i], end=" ")

print()

print("Liste elemanlari kucukten buyuge siralaniyor....")

for i in range(listeBoyutu):

print(liste[i], end=" ")

print(liste[j], end=" ")

print()

for j in range(listeBoyutu):

print(liste[i], end=" ")

print(liste[j], end=" ")

print()

if (liste[i]<liste[j]):

temp=liste[j]

liste[j]=liste[i]

liste[i]=temp

print(liste)

print("Liste elemanlari buyukten kucuge siralaniyor....")

for i in range(listeBoyutu):

print(liste[i], end=" ")

print(liste[j], end=" ")

print()

for j in range(listeBoyutu):

print(liste[i], end=" ")

print(liste[j], end=" ")

print()

if (liste[i]>liste[j]):

temp=liste[j]

liste[j]=liste[i]

liste[i]=temp

print(liste)

Result:

Liste boyutunu giriniz : 3

Liste olusturuluyor...

Liste yazdiriliyor...

3 98 15

Liste elemanlari kucukten buyuge siralaniyor....

3 15

3 3

3 98

98 15

3 15

3 98

98 98

98 15

15 15

15 3

15 98

98 98

[3, 15, 98]

Liste elemanlari buyukten kucuge siralaniyor....

3 98

3 3

3 15

3 98

15 98

15 3

3 3

3 98

98 98

98 15

15 3

3 3

[98, 15, 3]

Value Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Step | Action | List | i | j |
| 1 | Initialize list | [] | - | - |
| 2 | Append random value | [3] | 0 | - |
| 3 | Append random value | [3, 98] | 1 | - |
| 4 | Append random value | [3, 98, 15] | 2 | - |
| 5 | Print list | [3, 98, 15] | - | - |
| 6 | Start sorting in ascending order | [3, 98, 15] | 0 | 0 |
| 7 | Compare and swap if necessary | [3, 98, 15] | 0 | 1 |
| 8 | Compare and swap if necessary | [3, 98, 15] | 0 | 2 |
| 9 | Compare and swap if necessary | [3, 15, 98] | 1 | 0 |
| 10 | Compare and swap if necessary | [3, 15, 98] | 1 | 1 |
| 11 | Compare and swap if necessary | [3, 15, 98] | 1 | 2 |
| 12 | Compare and swap if necessary | [3, 15, 98] | 2 | 0 |
| 13 | Compare and swap if necessary | [3, 15, 98] | 2 | 1 |
| 14 | Compare and swap if necessary | [3, 15, 98] | 2 | 2 |
| 15 | Print sorted list | [3, 15, 98] | - | - |
| 16 | Start sorting in descending order | [3, 15, 98] | 0 | 0 |
| 17 | Compare and swap if necessary | [98, 15, 3] | 0 | 1 |
| 18 | Compare and swap if necessary | [98, 15, 3] | 0 | 2 |
| 19 | Compare and swap if necessary | [98, 15, 3] | 1 | 0 |
| 20 | Compare and swap if necessary | [98, 15, 3] | 1 | 1 |
| 21 | Compare and swap if necessary | [98, 15, 3] | 1 | 2 |
| 22 | Compare and swap if necessary | [98, 15, 3] | 2 | 0 |
| 23 | Compare and swap if necessary | [98, 15, 3] | 2 | 1 |
| 24 | Compare and swap if necessary | [98, 15, 3] | 2 | 2 |
| 25 | Print sorted list | [98, 15, 3] | - | - |

‘’’

# Example 5

matrix=[]

rowNumber=eval(input("Satir gir"))

columnNumber=eval(input("Sutun gir"))

for row in range(rowNumber):

matrix.append([])

for column in range(columnNumber):

value=eval(input(" elemanini giriniz : "))

matrix[row].append(value)

print(matrix)

‘’’

num\_rows = input("Enter the number of rows: ")

num\_cols = input("Enter the number of columns: ")

if not num\_rows.isdigit() or not num\_cols.isdigit():

print("Invalid input. Please enter an integer.")

else:

num\_rows = int(num\_rows)

num\_cols = int(num\_cols)

matrix = []

print("Enter the elements of the matrix:")

for row in range(num\_rows):

matrix.append([int(input(f"Enter element for row {row+1}, column {col+1}: ")) for col in range(num\_cols)])

print("The matrix is:")

for row in matrix:

print(row)

‘’’

# Example 6

import random

matrix=[]

rowNumber=eval(input("Satir gir"))

columnNumber=eval(input("Sutun gir"))

for row in range(rowNumber):

matrix.append([])

for column in range(columnNumber):

value=random.randint(0,100)

matrix[row].append(value)

print(matrix)

# Example 7

import random

matrix=[]

rowNumber=eval(input("Satir gir"))

columnNumber=eval(input("Sutun gir"))

print("Matris olusturuluyor....")

for row in range(rowNumber):

matrix.append([])

for column in range(columnNumber):

value=random.randint(0,9)

matrix[row].append(value)

print("Matris yazdiriliyor....")

for row in range(rowNumber):

for column in range(columnNumber):

print(matrix[row][column], end=" ")

print()

#Cell 1

def sum(a, b):

return a + b

def printMessage():

print("İslem tamamlandi")

sum(5, 6)

printMessage()

print("Girilen sayilarin toplami = ", sum(5, 6))

print("Islem Ozeti = ", printMessage())

result = sum(5, 6)

print("Sonuc = ", result)

resultOut = printMessage()

print(resultOut)

def sumNumbers(s1, s2):

sum = 0

for i in range(s1, s2 + 1):

sum += i

return sum

def main():

s1 = eval(input("İlk deger giriniz : "))

s2 = eval(input("İkinci deger giriniz : "))

print("Girilen iki araliktaki sayilarin toplami = ", sumNumbers(s1, s2))

main()

‘’’

def sum(a, b):

return a + b

def printMessage():

return "İslem tamamlandi"

print("Girilen sayilarin toplami = ", sum(5, 6))

print("Islem Ozeti = ", printMessage())

result = sum(5, 6)

print("Sonuc = ", result)

resultOut = printMessage()

print(resultOut)

def sumNumbers(s1, s2):

total = 0

for i in range(s1, s2 + 1):

total += i

return total

def main():

s1 = int(input("İlk deger giriniz : "))

s2 = int(input("Ikinci deger giriniz : "))

print("Girilen iki araliktaki sayilarin toplami = ", sumNumbers(s1, s2))

main()

‘’’

#Cell 2

def sumDigit(sayi):

sum = 0

while sayi != 0:

kalan = sayi % 10

sum += kalan

sayi -= kalan

sayi /= 10

return sum

def main():

sayi = eval(input("Lutfen bir sayi giriniz : "))

print("Girilen sayinin rakamlari toplami = ", int(sumDigit(sayi)))

main()

#Cell 3

def getDizi(boyut):

print("Dizi olusturuluyor....")

dizi = []

for i in range(boyut):

dizi.append(random.randint(0, 10))

print("Dizi rasgele sayilardan olusturuldu.")

print(dizi)

return dizi

def printDizi(dizi):

print("Dizi yazdiriliyor...")

for i in range(len(dizi)):

print(dizi[i], end=" ")

print()

def calculateDizi(dizi):

print("Dizi uzerinde islem yapiliyor....")

sum = 0

for i in range(len(dizi)):

sum += dizi[i]

return sum

def main():

boyut = eval(input("Dizi boyutunu giriniz : "))

dizi = getDizi(boyut)

printDizi(dizi)

print("Dizi elemanlarinin toplami = ", calculateDizi(dizi))

main()

‘’’

import random

def generate\_random\_array(size):

print("Generating array....")

array = [random.randint(0, 10) for \_ in range(size)]

print("Array created with random numbers.")

print(array)

return array

def print\_array(array):

print("Printing array...")

print(' '.join(map(str, array)))

def calculate\_array\_sum(array):

print("Calculating sum of array....")

total = 0

for num in array:

total += num

return total

def main():

size = int(input("Enter array size : "))

array = generate\_random\_array(size)

print\_array(array)

print("Sum of array elements = ", calculate\_array\_sum(array))

main()

‘’’

#Cell 4

import random

def getMatrix():

print("Matrix olusturuluyor....")

rowNumber = eval(input("Satir gir : "))

columnNumber = eval(input("Sutun gir : "))

matrix = []

for row in range(rowNumber):

matrix.append([])

for column in range(columnNumber):

matrix[row].append(random.randint(0, 10))

print("Matrix olusturuldu...")

print(matrix)

return matrix

def printMatrix(matrix):

print("Matrix Yazdiriliyor...")

for row in range(len(matrix)):

for column in range(len(matrix[row])):

print(matrix[row][column], end=" ")

print()

def calculateMatrix(matrix):

sumOdd = 0

sumEven = 0

for row in range(len(matrix)):

for column in range(len(matrix[row])):

if matrix[row][column] % 2 == 0:

sumEven += matrix[row][column]

else:

sumOdd += matrix[row][column]

return sumOdd, sumEven

def main():

matrix = getMatrix()

printMatrix(matrix)

tek, cift = calculateMatrix(matrix)

print("Olusan Matrisin tek elemanlarinin toplami = ", tek)

print("Olusan Matrisin cift elemanlarinin toplami = ", cift)

main()

‘’’

import random

def generate\_random\_matrix():

print("Generating matrix....")

row\_number = int(input("Enter number of rows : "))

column\_number = int(input("Enter number of columns : "))

matrix = [[random.randint(0, 10) for \_ in range(column\_number)] for \_ in range(row\_number)]

print("Matrix created...")

print(matrix)

return matrix

def print\_matrix(matrix):

print("Printing matrix...")

for row in matrix:

print(' '.join(map(str, row)))

def calculate\_odd\_sum(matrix):

print("Calculating sum of odd numbers in the matrix....")

total\_odd = 0

for row in matrix:

for element in row:

if element % 2 != 0:

total\_odd += element

return total\_odd

def calculate\_even\_sum(matrix):

print("Calculating sum of even numbers in the matrix....")

total\_even = 0

for row in matrix:

for element in row:

if element % 2 == 0:

total\_even += element

return total\_even

def main():

matrix = generate\_random\_matrix()

print\_matrix(matrix)

odd\_sum = calculate\_odd\_sum(matrix)

even\_sum = calculate\_even\_sum(matrix)

print("Sum of odd elements in the matrix = ", odd\_sum)

print("Sum of even elements in the matrix = ", even\_sum)

main()

‘’’

import random

# Function to create a predefined matrix

def create\_predefined\_matrix():

# Define the matrix

matrix = [

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

[6, 7, 0, 0, 0],

[0, 1, 0, 0, 0],

[1, 0, 0, 0, 8],

[0, 0, 9, 0, 3]

]

return matrix

# Function to access a specific element in the matrix

def access\_specific\_element(matrix):

# Access and return the element at row 4, column 2

return matrix[4][2]

# Function to create a matrix based on user input

def create\_matrix\_user\_input():

# Get the number of rows and columns from the user

numberOfRows = int(input("Enter the number of rows: "))

numberOfColumns = int(input("Enter the number of columns: "))

# Initialize an empty matrix

matrix = []

# Loop over the rows

for row in range(numberOfRows):

# Add an empty row to the matrix

matrix.append([])

# Loop over the columns

for column in range(numberOfColumns):

# Get a value from the user and add it to the current row

value = int(input("Enter an element and press Enter: "))

matrix[row].append(value)

return matrix

# Function to create a matrix with random values

def create\_random\_matrix():

# Get the number of rows and columns from the user

numberOfRows = int(input("Enter the number of rows: "))

numberOfColumns = int(input("Enter the number of columns: "))

# Initialize an empty matrix

matrix = []

# Loop over the rows

for row in range(numberOfRows):

# Add an empty row to the matrix

matrix.append([])

# Loop over the columns

for column in range(numberOfColumns):

# Generate a random number and add it to the current row

matrix[row].append(random.randint(0, 100))

return matrix

# Function to print a matrix

def print\_matrix(matrix):

# Loop over the rows

for row in range(len(matrix)):

# Loop over the columns

for column in range(len(matrix[row])):

# Print the current element, followed by a space

print(matrix[row][column], end=" ")

# Print a newline to move to the next row

print()

# Function to calculate the sum of all elements in a matrix

def calculate\_matrix\_sum(matrix):

# Initialize the total sum to 0

total = 0

# Loop over the rows

for row in range(len(matrix)):

# Loop over the columns

for column in range(len(matrix[row])):

# Add the current element to the total sum

total += matrix[row][column]

return total

# Function to calculate the sum of the digits in a number

def sum\_digits(number):

# Initialize the total sum to 0

total = 0

# While the number is not 0

while number != 0:

# Get the last digit of the number

digit = number % 10

# Add the digit to the total sum

total += digit

# Remove the last digit from the number

number //= 10

return total

# Function to create a matrix with random values based on user input

def create\_matrix\_with\_user\_input():

# Get the number of rows and columns from the user

numberOfRows = int(input("Enter the number of rows: "))

numberOfColumns = int(input("Enter the number of columns: "))

# Initialize an empty matrix

matrix = []

# Loop over the rows

for row in range(numberOfRows):

# Add an empty row to the matrix

matrix.append([])

# Loop over the columns

for column in range(numberOfColumns):

# Generate a random number and add it to the current row

matrix[row].append(random.randint(0, 10))

return matrix

# Main function

def main():

# Create a predefined matrix

predefined\_matrix = create\_predefined\_matrix()

# Access a specific element in the predefined matrix

specific\_element = access\_specific\_element(predefined\_matrix)

print("Specific element:", specific\_element)

# Create a matrix based on user input

user\_input\_matrix = create\_matrix\_user\_input()

print("User input matrix:")

print\_matrix(user\_input\_matrix)

# Create a matrix with random values

random\_matrix = create\_random\_matrix()

print("Random matrix:")

print\_matrix(random\_matrix)

# Print the random matrix

print("Random matrix:")

print\_matrix(random\_matrix)

# Calculate the sum of all elements in the random matrix

random\_matrix\_sum = calculate\_matrix\_sum(random\_matrix)

print("Sum of all elements in random matrix:", random\_matrix\_sum)

# Calculate the sum of the digits in a number

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

digits\_sum = sum\_digits(number)

print("Sum of digits in number:", digits\_sum)

# Perform matrix operations with user input

user\_input\_matrix = create\_matrix\_with\_user\_input()

print("User input matrix:")

print\_matrix(user\_input\_matrix)

user\_input\_matrix\_sum = calculate\_matrix\_sum(user\_input\_matrix)

print("Sum of all elements in user input matrix:", user\_input\_matrix\_sum)

# If this script is the main script, run the main function

if \_\_name\_\_ == "\_\_main\_\_":

main()

# Example 1

NUMBER\_OF\_ELEMENTS = 5

numbers = [] # list operator

sum = 0

for i in range(NUMBER\_OF\_ELEMENTS):

value = eval(input("Enter a new number: "))

numbers.append(value)

sum = sum + value

average = sum / NUMBER\_OF\_ELEMENTS

count = 0

for i in range(NUMBER\_OF\_ELEMENTS):

if numbers[i] > average: #index = 0, 1, 2, 3, 4

count += 1

print("Average is", average)

print("Number of elements above the average is", count)

print("Length of list", len(numbers))

print("Min element of list", min(numbers))

print("Max element of list", max(numbers))

‘’’

NUMBER\_OF\_ELEMENTS = 5

numbers = [] # list operator

total = 0

for i in range(NUMBER\_OF\_ELEMENTS):

value = float(input("Enter a new number: "))

numbers.append(value)

total = total + value

average = total / NUMBER\_OF\_ELEMENTS

count = 0

for i in range(NUMBER\_OF\_ELEMENTS):

if numbers[i] > average: #index = 0, 1, 2, 3, 4

count += 1

print("Average is", average)

print("Number of elements above the average is", count)

print("Length of list", len(numbers))

print("Min element of list", min(numbers))

print("Max element of list", max(numbers))

‘’’

# Example 2

list3 = list(["red", "green", "blue"])

print("Min element of list", min(list3))

print("Max element of list", max(list3))

# Example 3

list1 = [2, 3, 4, 1, 32, 4]

list1.append(19)

print(list1)

print(list1.count(4))

list2 = [99, 54]

list1.extend(list2)

print(list1)

list1.insert(1, 25)

print(list1)

# Example 4

list1 = [2, 25, 3, 4, 1, 32, 4, 19, 99, 54]

list1.pop(2)

list1.pop()

list1.remove(32)

list1.reverse()

print(list1)

list1.sort()

print(list1)

items = "Jane,John,Peter,Susan".split(",")

print(items)

‘’’

pop(): This is a built-in function in Python that removes and returns an element at the given index from the list. If no index is specified, pop() removes and returns the last item in the list. In your code:

list1.pop(2) removes the third element (since Python uses 0-based indexing), which is 3 in this case.

list1.pop() without an argument removes the last element, which is 54 after the previous operation.

remove(): This function removes the first occurrence of a value in the list. list1.remove(32) removes the first 32 found in the list.

reverse(): This function reverses the order of the items in the list. list1.reverse() reverses the entire list.

sort(): This function sorts the items in the list in ascending order (from low to high). list1.sort() sorts the list.

split(): This is a string method that splits a string into a list where each word is a list item. The split occurs at the specified separator. In your code, "Jane,John,Peter,Susan".split(",") splits the string at each comma, resulting in a list of names.

Here’s a step-by-step breakdown of what your code does:

list1 = [2, 25, 3, 4, 1, 32, 4, 19, 99, 54]

list1.pop(2) # Removes the third element (3), list1 is now [2, 25, 4, 1, 32, 4, 19, 99, 54]

list1.pop() # Removes the last element (54), list1 is now [2, 25, 4, 1, 32, 4, 19, 99]

list1.remove(32) # Removes the first occurrence of 32, list1 is now [2, 25, 4, 1, 4, 19, 99]

list1.reverse() # Reverses the list, list1 is now [99, 19, 4, 1, 4, 25, 2]

print(list1) # Prints the reversed list

list1.sort() # Sorts the list in ascending order, list1 is now [1, 2, 4, 4, 19, 25, 99]

print(list1) # Prints the sorted list

items = "Jane,John,Peter,Susan".split(",") # Splits the string into a list of names

print(items) # Prints the list of names

‘’’

# Example 1

day = input("Enter the day of the week: ").capitalize()

match day:

case "Saturday" | "Sunday":

print(f"{day} is a weekend.")

case "Monday" | "Tuesday" | "Wednesday" | "Thursday" | "Friday":

print(f"{day} is a weekday.")

case \_:

print("That's not a valid day of the week.")

‘’’

The capitalize() function is a built-in Python method that can be used with strings. It converts the first character of a string to uppercase (if it’s a letter) and makes all other characters in the string lowercase.

‘’’

# Example 2

gun = input("Bir gün giriniz")

match gun:

case "Pazartesi" | "Sali" | "Carsamba" | "Persembe" | "Cuma":

print("Günlerden Haftaici")

case "Cumartesi" | "Pazar":

print("Günlerden haftasonu")

case \_:

print("Geçersiz gün")

# Example 3

sum = 0

number = 0

while number < 20:

number += 1

sum += number

if sum >= 100:

continue

print("The number is", number)

print("The sum is", sum)

# Example 4

sum = 0

number = 0

while number < 20:

number += 1

if number == 10 or number == 11:

continue

sum += number

print("The sum is", sum)

# Example 5

for i in range(0, 100):

if i == 50:

continue

print(i)

# Example 1

dizi=[]

diziElemanSayisi=eval(input("Dizi boyutunu giriniz : "))

for i in range(diziElemanSayisi):

value = eval(input("Enter a new number: "))

dizi.append(value)

# Example 2

import random

dizi=[]

diziElemanSayisi=eval(input("Dizi boyutunu giriniz : "))

for i in range(diziElemanSayisi):

value = random.randint(0,50)

dizi.append(value)

for i in range(diziElemanSayisi):

print(dizi[i] , end=" ")

# Example 3

import random

dizi=[]

diziElemanSayisi=eval(input("Dizi boyutunu giriniz : "))

for i in range(diziElemanSayisi):

value = random.randint(0,50)

dizi.append(value)

for i in range(diziElemanSayisi):

print(dizi[i] , end=" ")

print("en kucuk eleman : ")

enKucuk=dizi[0]

for i in range(diziElemanSayisi):

if dizi[i]<enKucuk:

enKucuk=dizi[i]

print("En Kucuk Eleman : ", enKucuk)

# Example 4

import random

dizi=[]

diziElemanSayisi=eval(input("Dizi boyutunu giriniz : "))

for i in range(diziElemanSayisi):

value = random.randint(0,50)

dizi.append(value)

for i in range(diziElemanSayisi):

print(dizi[i] , end=" ")

print("Sıralanıyor....")

for i in range(diziElemanSayisi):

for j in range (diziElemanSayisi):

if dizi[i]<dizi[j]:

temp=dizi[j]

dizi[j]=dizi[i]

dizi[i]=temp

print(dizi)

# Example 5

import random

matrix=[]

rowNumber=eval(input("Satir sayisini giriniz : "))

columnNuber=eval(input("Sutun sayisini giriniz : "))

print("Matris olusturuluyor...")

for row in range(rowNumber):

matrix.append([])

for column in range (columnNuber):

matrix[row].append(random.randint(0,50))

print("Matris Yazdiriliyor...")

for row in range(rowNumber):

for column in range(columnNuber):

print(matrix[row][column],end=" ")

print()

print("En kucuk bulunuyor")

enKucuk=matrix[0][0]

for row in range(rowNumber):

for column in range(columnNuber):

if(matrix[row][column] < enKucuk):

enKucuk=matrix[row][column]

print(enKucuk)

|  |
| --- |
| Yorum satırları ve kod içi İngilizce kısımlar doğru kod yazımında alıştırma olması için eklenmiştir. İlerleyen süreçte yorum satırları da İngilizce olarak eklenecektir.  # 4.1  # Matematik kütüphanesi eklenir.  import math  # Değişkenler float tanımlı bir şekilde kullanıcıdan denklemin katsayıları istenir.  a = float(input("Enter the coefficient a: "))  b = float(input("Enter the coefficient b: "))  c = float(input("Enter the coefficient c: "))  # Discriminant hesaplanır.  discriminant = b\*\*2 - 4\*a\*c  # Gerekli koşul durumları kontrol edilir, rootlar hesaplanır, kullanıcıya çıktı verilir.  if discriminant > 0:  root1 = (-b + math.sqrt(discriminant)) / (2\*a)  root2 = (-b - math.sqrt(discriminant)) / (2\*a)  print(f"The equation has two roots: {root1} and {root2}")  # F-sözdizimi, çıkış mesajlarına root'ların değerlerini eklemek için kullanıldı.  elif discriminant == 0:  root = -b / (2\*a)  print(f"The equation has one root: {root}")  else:  print("The equation has no real roots.")  The equation has no real roots.  # 4.2  # Rastgele sayı üretmek için gerekli kütüphane eklenir.  import random  # Rastgele olarak üç sayı tanımlanır.  num1 = random.randint(0, 9)  num2 = random.randint(0, 9)  num3 = random.randint(0, 9)  # Kulanıcıdan numaraların toplamını girmesi istenir, "user\_sum" ile hafızaya alınır ve kullanıcıya girdiği cevap gösterilir.  print(f"Enter the sum of {num1}, {num2}, and {num3}")  user\_sum = int(input("Your answer: "))  # Gerçek cevap hesaplanır.  pc\_sum = num1 + num2 + num3  # Gerekli koşul durumları kontrol edilerek kullanıcıya çıktı verilir.  if user\_sum == pc\_sum:  print("Correct!")  else:  print(f"Wrong. The correct answer is {pc\_sum}.")  Enter the sum of 7, 0, and 5  Correct!  # 4.3  # Kullanıcıdan sayılar istenir.  a = float(input("Enter a: "))  b = float(input("Enter b: "))  c = float(input("Enter c: "))  d = float(input("Enter d: "))  e = float(input("Enter e: "))  f = float(input("Enter f: "))  # Önce 0 olma durumu kontrol edilerek kod temiz tutulur. Ardından koşullar kontrol edilip kullanıcıya çıktı verilir.  if a \* d - b \* c == 0:  print("The equation has no solution.")  else:  # Hesaplamalar yapılarak kullanıcıya sonuçlar çıktı olarak verilir.  x = (e \* d - b \* f) / (a \* d - b \* c)  y = (a \* f - e \* c) / (a \* d - b \* c)  print(f"x is {x} and y is {y}")  x is -4.0 and y is 4.5  # 4.4  # Gerekli kütüphane eklenir.  import random  # 0 ile 100 arası rastgele iki sayı tanımlanır.  num1 = random.randint(0, 99)  num2 = random.randint(0, 99)  # Kullanıcıdan toplamlarının girilmesi istenir.  print(f"Enter the sum of {num1} and {num2}")  # Kulanıcıdan numaraların toplamını girmesi istenir, "user\_sum" ile hafızaya alınır ve kullanıcıya girdiği cevap gösterilir.  user\_sum = int(input("Your answer: "))  # Gerekli koşul durumları kontrol edilip kullanıcıya çıktı verilir.  if user\_sum == num1 + num2:  print(True)  else:  print(False)  Enter the sum of 48 and 52  False  # 4.5  # Haftanın günleri bir dizi/liste içine, ilk gün sıfır ile başlayacağından dolayı, alınır.  days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]  # Kullanıcıya haftanın hangi gününde olunduğu sorulur. "today" değişkeni ile hafızaya alınır.  today = int(input("Enter today's day (Sunday is 0, Monday is 1, ..., and Saturday is 6): "))  # Prompt the user to enter the number of days after today  elapsed\_days = int(input("Enter the number of days elapsed since today: "))  # Calculate the future day of the week  future\_day = (today + elapsed\_days) % 7  # Display the result  print(f"Today is {days[today]} and the future day is {days[future\_day]}")  Today is Wednesday and the future day is Monday  # 4.6  # Kullanıcıdan boy değerleri metre ve cm, ağırlık değeri kilogram cinsinden istenir.  height\_m = float(input("Please enter the height in meters: "))  height\_cm = float(input("Please enter remaining height in centimeters: "))  weight = float(input("Please enter weight in kg: "))  # Boyu santimetre cinsinden metre cinsine çevirip BMI hesabı yapıp ekrana yazdırılır.  height\_total = height\_m + height\_cm/100  BMI = weight/(height\_total\*\*2)  print("BMI: ",BMI)  # Koşullar ile kullanıcının durumu karşılaştırılıp kullanıcıya verilir.  if BMI < 18:  print("You are underweight")  elif BMI < 25:  print("You are normal")  elif BMI <30:  print("You are overweight")  else:  print("You are obese")  BMI: 23.671253629592222  You are normal  # 4.7  # Kullanıcıdan düzenlenmesi istenen miktar alınır.  amount = eval(input("Enter an amount, for example, 11.56: "))  # Miktarı cents'lere dönüştürülür.  remainingAmount = int(amount \* 100)  # Banknot ve madeni para miktarları hesaplanır.  dollars = remainingAmount // 100  remainingAmount %= 100  quarters = remainingAmount // 25  remainingAmount %= 25  dimes = remainingAmount // 10  remainingAmount %= 10  nickels = remainingAmount // 5  remainingAmount %= 5  pennies = remainingAmount  # Sonuçlar kullanıcıya koşullar ile kontrol edilip sunulur.  print("Your amount", amount, "consists of:")  if dollars == 1:  print("\t", dollars, "dollar")  else:  print("\t", dollars, "dollars")  if quarters == 1:  print("\t", quarters, "quarter")  else:  print("\t", quarters, "quarters")  if dimes == 1:  print("\t", dimes, "dime")  else:  print("\t", dimes, "dimes")  if nickels == 1:  print("\t", nickels, "nickel")  else:  print("\t", nickels, "nickels")  if pennies == 1:  print("\t", pennies, "pennie")  else:  print("\t", pennies, "penies")  Your amount 14.67 consists of:  14 dollars  2 quarters  1 dime  1 nickel  2 penies  # 4.8  num1 = float(input("Please enter first number:"))  num2 = float(input("Please enter second number:"))  num3 = float(input("Please enter third number:"))  sorted\_nums = sorted([num1,num2,num3])  print("The numbers in increasing order are:",sorted\_nums[0],sorted\_nums[1],sorted\_nums[2])  The numbers in increasing order are: 134.0 232.0 2345.0  # 4.9  # Kullanıcıdan ilk ve ikinci paketlerin ağırlık ve fiyat bilgisi alınır.  weight1 = float(input("Enter the weight of the first package: "))  price1 = float(input("Enter the price of the first package: "))  weight2 = float(input("Enter the weight of the second package: "))  price2 = float(input("Enter the price of the second package: "))  # Gerekli hesaplamalar yapılır.  unit\_price1 = price1 / weight1  unit\_price2 = price2 / weight2  # Gerekli koşullar ile kullanıcıya çıktı verilir.  if unit\_price1 < unit\_price2:  print("Package 1 has the better price.")  elif unit\_price1 > unit\_price2:  print("Package 2 has the better price.")  else:  print("Both packages have the same price.")  Package 2 has the better price.  # 4.10  import random  # Yüzden küçük rastgele iki sayı üretilip hafızaya alınır.  number1 = random.randint(0, 99)  number2 = random.randint(0, 99)  # Kullanıcıdan hesaplama sonucu istenir.  answer = eval(input("What is " + str(number1) + " \* " + str(number2) + "? "))  # Kullanıcan alınan cevap kontrol edilip kullanıcıya çıktı verilir.  if number1 \* number2 == answer:  print("You are correct!")  else:  print("Your answer is wrong.\n", number1, '\*', number2, "is", number1 \* number2, '.')  You are correct!  # 4.11  # Kullanıcıdan ay ve yılı girmesi istenir.  month = int(input("Enter the month (1-12): "))  year = int(input("Enter the year: "))  # Önce şubat ayı kontrol edilir, ardından diğer kontroller ile birlikte kullanıcıya çıktı verilir.  if month == 2:  # Burada en kısa nasıl yapabılabilceğini araştırıp bulduklarımı koda  if year % 4 == 0 and year % 100 != 0 or year % 400 == 0:  print(f"February {year} has 29 days")  else:  print(f"February {year} has 28 days")  elif month == 4 or month == 6 or month == 9 or m == 11:  print(f"Month {month} of year {year} has 30 days")  elif month == 1 or month == 3 or month == 5 or month == 7 or month == 8 or month == 10 or month == 12:  print(f"Month {month} of year {year} has 31 days")  else:  print("Invalid month entered.")  February 12312412 has 29 days  # 4.12  # Kullanıcıdan bir sayı istenir.  number = int(input("Enter an integer: "))  # Kullanıcıdan alınan sayının bölünüp bölünmeme durumu kontrol edilir.  divisible\_by\_5\_and\_6 = (number % 5 == 0) and (number % 6 == 0)  divisible\_by\_5\_or\_6 = (number % 5 == 0) or (number % 6 == 0)  divisible\_byOnly\_5\_or\_6 = (number % 5 == 0) != (number % 6 == 0)  # Kullanıcıya sonuçlar verilir.  print(f"Is {number} divisible by 5 and 6? {divisible\_by\_5\_and\_6}")  print(f"Is {number} divisible by 5 or 6? {divisible\_by\_5\_or\_6}")  print(f"Is {number} divisible by 5 or 6, but not both? {divisible\_byOnly\_5\_or\_6}")  Is 15 divisible by 5 and 6? False  Is 15 divisible by 5 or 6? True  Is 15 divisible by 5 or 6, but not both? True  # 4.13  # "sys.exit()" fonksiyonunu kullandığımız için SYS modülünü import ettik.  # Sys modülü kullanıcı tarafından geçersiz bir durum girilirse, sys.exit() programdan çıkmak için kullanılır.  import sys  status = eval(input(  "(0-single filer, 1-married jointly,\n" +  "2-married separately, 3-head of household)\n" +  "Enter the filing status: "))  income = eval(input("Enter the taxable income: "))  tax = 0  if status == 0:  if income <= 8350:  tax = income \* 0.10  elif income <= 33950:  tax = 8350 \* 0.10 + (income - 8350) \* 0.15  elif income <= 82250:  tax = 8350 \* 0.10 + (33950 - 8350) \* 0.15 + \  (income - 33950) \* 0.25  elif income <= 171550:  tax = 8350 \* 0.10 + (33950 - 8350) \* 0.15 + \  (82250 - 33950) \* 0.25 + (income - 82250) \* 0.28  elif income <= 372950:  tax = 8350 \* 0.10 + (33950 - 8350) \* 0.15 + \  (82250 - 33950) \* 0.25 + (171550 - 82250) \* 0.28 + \  (income - 171550) \* 0.33  else:  tax = 8350 \* 0.10 + (33950 - 8350) \* 0.15 + \  (82250 - 33950) \* 0.25 + (171550 - 82250) \* 0.28 + \  (372950 - 171550) \* 0.33 + (income - 372950) \* 0.35  elif status == 1:  if income <= 16700:  tax = income \* 0.10  elif income <= 67900:  tax = 16700 \* 0.10 + (income - 16700) \* 0.15  elif income <= 137050:  tax = 16700 \* 0.10 + (67900 - 16700) \* 0.15 + \  (income - 67900) \* 0.25  elif income <= 208850:  tax = 16700 \* 0.10 + (67900 - 16700) \* 0.15 + \  (137050 - 67900) \* 0.25 + (income - 137050) \* 0.28  elif income <= 372950:  tax = 16700 \* 0.10 + (67900 - 16700) \* 0.15 + \  (137050 - 67900) \* 0.25 + (208850 - 137050) \* 0.28 + \  (income - 208850) \* 0.33  else:  tax = 16700 \* 0.10 + (67900 - 16700) \* 0.15 + \  (137050 - 67900) \* 0.25 + (208850 - 137050) \* 0.28 + \  (372950 - 208850) \* 0.33 + (income - 372950) \* 0.35  elif status == 2:  print("Left as exercise")  elif status == 3:  print("Left as exercise")  else:  print("Error: invalid status")  sys.exit()  print("Tax is", format(tax, ".2f"))  Tax is 1832.50  # 4.14  import random  coin\_flip = random.randint(0, 1)  guess = input("Guess the result (0 for heads, 1 for tails): ")  if guess not in ['0', '1']:  print("Invalid input. Please enter 0 for heads or 1 for tails.")  else:  guess = int(guess)  if coin\_flip == guess:  print("Congratulations! You guessed correctly.")  else:  print("Sorry, you guessed incorrectly.")  Congratulations! You guessed correctly.  # 4.15  import random  lottery = random.randint(100, 999)  guess = int(input("Enter your lottery pick (three digits): "))  lotteryDigit1 = lottery // 100  lotteryDigit2 = (lottery % 100) // 10  lotteryDigit3 = lottery % 10  guessDigit1 = guess // 100  guessDigit2 = (guess % 100) // 10  guessDigit3 = guess % 10  print("The lottery number is", lottery)  if guess == lottery:  print("Exact match: you win $10,000")  elif (guessDigit1 == lotteryDigit1 or guessDigit1 == lotteryDigit2 or guessDigit1 == lotteryDigit3) \  and (guessDigit2 == lotteryDigit1 or guessDigit2 == lotteryDigit2 or guessDigit2 == lotteryDigit3) \  and (guessDigit3 == lotteryDigit1 or guessDigit3 == lotteryDigit2 or guessDigit3 == lotteryDigit3):  print("Match all digits: you win $3,000")  elif (guessDigit1 == lotteryDigit1 or guessDigit1 == lotteryDigit2 or guessDigit1 == lotteryDigit3) \  or (guessDigit2 == lotteryDigit1 or guessDigit2 == lotteryDigit2 or guessDigit2 == lotteryDigit3) \  or (guessDigit3 == lotteryDigit1 or guessDigit3 == lotteryDigit2 or guessDigit3 == lotteryDigit3):  print("Match one digit: you win $1,000")  else:  print("Sorry, no match")  The lottery number is 539  Match one digit: you win $1,000  # 4.16  import random  random\_letter = chr(random.randint(65, 90))  print("Random uppercase letter:", random\_letter)  Random uppercase letter: W  # 4.17  import random  options = ['scissor', 'rock', 'paper']  computer\_choice = random.choice(options)  print("Enter your choice:")  print("0: Scissor")  print("1: Rock")  print("2: Paper")  user\_choice = int(input("Your choice (0, 1, or 2): "))  print("You chose:", options[user\_choice])  print("The computer chose:", computer\_choice)  if user\_choice == options.index(computer\_choice):  print("It's a draw!")  elif (user\_choice == 0 and computer\_choice == 'paper') or \  (user\_choice == 1 and computer\_choice == 'scissor') or \  (user\_choice == 2 and computer\_choice == 'rock'):  print("The computer wins!")  else:  print("You win!")  Enter your choice:  0: Scissor  1: Rock  2: Paper  You chose: paper  The computer chose: scissor  You win!  # 4.18  exchange\_rate = float(input("Enter the exchange rate from dollars to RMB: "))  option = int(input("Enter 0 to convert dollars to RMB and 1 vice versa: "))  if option == 0:  dollar\_amount = float(input("Enter the dollar amount: "))  rmb\_amount = dollar\_amount \* exchange\_rate  print(f"${dollar\_amount} is {rmb\_amount} yuan")  elif option == 1:  rmb\_amount = float(input("Enter the RMB amount: "))  dollar\_amount = rmb\_amount / exchange\_rate  print(f"{rmb\_amount} yuan is ${dollar\_amount:.2f}")  else:  print("Incorrect input")  10000.0 yuan is $1468.43  # 4.19  edges = input("Enter three edges (separated by commas): ").split(',')  edges = [int(edge) for edge in edges]  if edges[0] + edges[1] > edges[2] and edges[1] + edges[2] > edges[0] and edges[0] + edges[2] > edges[1]:  perimeter = sum(edges)  print("The perimeter is", perimeter)  else:  print("The input is invalid")  The perimeter is 3  # 4.20  temperature = float(input("Enter the temperature in Fahrenheit: "))  wind\_speed = float(input("Enter the wind speed in miles per hour: "))  if temperature > 41 or wind\_speed < 2:  if temperature > 41:  print("Temperature is invalid. It must be 41°F or lower.")  if wind\_speed < 2:  print("Wind speed is invalid. It must be 2 mph or higher.")  else:  wind\_chill = 35.74 + 0.6215 \* temperature - 35.75 \* (wind\_speed \*\* 0.16) + 0.4275 \* temperature \* (wind\_speed \*\* 0.16)  print("The wind-chill temperature is:", round(wind\_chill, 2), "°F")  The wind-chill temperature is: 18.38 °F |

Yorum satırları ve kod içi İngilizce kısımlar doğru kod yazımında alıştırma olması için eklenmiştir. İlerleyen süreçte yorum satırları da İngilizce olarak eklenecektir.

# 5.1

count\_positive = 0

count\_negative = 0

total = 0

count = 0

# Burada for yerine while döngüsü kodu kısaltmak için kullanıldı.

while True:

number = int(input("Enter an integer, the input ends if it is 0: "))

if number == 0:

break

if number > 0:

count\_positive += 1

elif number < 0:

count\_negative += 1

total += number

count += 1

if count == 0:

print("You didn't enter any number")

else:

average = total / count

print(f"The number of positives is {count\_positive}")

print(f"The number of negatives is {count\_negative}")

print(f"The total is {total}")

print(f"The average is {average:.2f}")

The number of positives is 5

The number of negatives is 1

The total is 12

The average is 2.00

# 5.2

# Python'un güzel yanlarından biri de geniş kütüphane seçekleri ve kolay entegre edilebilmesi.

import random

import time

correctCount = 0

# Burada on soru çok kalabalık bir görüntü oluşturduğu için beşe indirildi.

NUMBER\_OF\_QUESTIONS = 5

# NUMBER\_OF\_QUESTIONS = 10

startTime = time.time()

for questionNumber in range(NUMBER\_OF\_QUESTIONS):

number1 = random.randint(1, 15)

number2 = random.randint(1, 15)

answer = eval(input("What is " + str(number1) + " + " + str(number2) + "? "))

if number1 + number2 == answer:

print("You are correct!")

correctCount += 1

else:

print("Your answer is wrong.\n", number1, "+", number2, "is", number1 + number2)

endTime = time.time()

testTime = int(endTime - startTime)

print("Correct count is", correctCount, "out of", NUMBER\_OF\_QUESTIONS, "\nTest time is", testTime, "seconds")

You are correct!

You are correct!

Your answer is wrong.

15 + 15 is 30

You are correct!

You are correct!

Correct count is 4 out of 5

Test time is 11 seconds

# 5.3

print(f'{"Kilograms":<10} {"Pounds"}')

for i in range(1, 200, 2):

pounds = i \* 2.2

print(f'{i:<10} {pounds:.1f}')

Kilograms Pounds

1 2.2

3 6.6

5 11.0

7 15.4

9 19.8

11 24.2

13 28.6

15 33.0

17 37.4

19 41.8

21 46.2

23 50.6

25 55.0

27 59.4

29 63.8

31 68.2

33 72.6

35 77.0

37 81.4

39 85.8

41 90.2

43 94.6

45 99.0

47 103.4

49 107.8

51 112.2

53 116.6

55 121.0

57 125.4

59 129.8

61 134.2

63 138.6

65 143.0

67 147.4

69 151.8

71 156.2

73 160.6

75 165.0

77 169.4

79 173.8

81 178.2

83 182.6

85 187.0

87 191.4

89 195.8

91 200.2

93 204.6

95 209.0

97 213.4

99 217.8

101 222.2

103 226.6

105 231.0

107 235.4

109 239.8

111 244.2

113 248.6

115 253.0

117 257.4

119 261.8

121 266.2

123 270.6

125 275.0

127 279.4

129 283.8

131 288.2

133 292.6

135 297.0

137 301.4

139 305.8

141 310.2

143 314.6

145 319.0

147 323.4

149 327.8

151 332.2

153 336.6

155 341.0

157 345.4

159 349.8

161 354.2

163 358.6

165 363.0

167 367.4

169 371.8

171 376.2

173 380.6

175 385.0

177 389.4

179 393.8

181 398.2

183 402.6

185 407.0

187 411.4

189 415.8

191 420.2

193 424.6

195 429.0

197 433.4

199 437.8

# 5.4

print(f'{"Miles":<10} {"Kilometers"}')

for i in range(1, 11):

kilometers = i \* 1.609

print(f'{i:<10} {kilometers:.3f}')

Miles Kilometers

1 1.609

2 3.218

3 4.827

4 6.436

5 8.045

6 9.654

7 11.263

8 12.872

9 14.481

10 16.090

# 5.5

print(f'{"Kilograms":<10} {"Pounds":<10} | {"Pounds":<10} {"Kilograms"}')

for i in range(1, 200, 2):

pounds = i \* 2.2

kilograms = (i \* 10 + 15) \* 0.45

print(f'{i:<10} {pounds:<10.1f} | {(i\*10 + 15):<10} {kilograms:.2f}')

Kilograms Pounds | Pounds Kilograms

1 2.2 | 25 11.25

3 6.6 | 45 20.25

5 11.0 | 65 29.25

7 15.4 | 85 38.25

9 19.8 | 105 47.25

11 24.2 | 125 56.25

13 28.6 | 145 65.25

15 33.0 | 165 74.25

17 37.4 | 185 83.25

19 41.8 | 205 92.25

21 46.2 | 225 101.25

23 50.6 | 245 110.25

25 55.0 | 265 119.25

27 59.4 | 285 128.25

29 63.8 | 305 137.25

31 68.2 | 325 146.25

33 72.6 | 345 155.25

35 77.0 | 365 164.25

37 81.4 | 385 173.25

39 85.8 | 405 182.25

41 90.2 | 425 191.25

43 94.6 | 445 200.25

45 99.0 | 465 209.25

47 103.4 | 485 218.25

49 107.8 | 505 227.25

51 112.2 | 525 236.25

53 116.6 | 545 245.25

55 121.0 | 565 254.25

57 125.4 | 585 263.25

59 129.8 | 605 272.25

61 134.2 | 625 281.25

63 138.6 | 645 290.25

65 143.0 | 665 299.25

67 147.4 | 685 308.25

69 151.8 | 705 317.25

71 156.2 | 725 326.25

73 160.6 | 745 335.25

75 165.0 | 765 344.25

77 169.4 | 785 353.25

79 173.8 | 805 362.25

81 178.2 | 825 371.25

83 182.6 | 845 380.25

85 187.0 | 865 389.25

87 191.4 | 885 398.25

89 195.8 | 905 407.25

91 200.2 | 925 416.25

93 204.6 | 945 425.25

95 209.0 | 965 434.25

97 213.4 | 985 443.25

99 217.8 | 1005 452.25

101 222.2 | 1025 461.25

103 226.6 | 1045 470.25

105 231.0 | 1065 479.25

107 235.4 | 1085 488.25

109 239.8 | 1105 497.25

111 244.2 | 1125 506.25

113 248.6 | 1145 515.25

115 253.0 | 1165 524.25

117 257.4 | 1185 533.25

119 261.8 | 1205 542.25

121 266.2 | 1225 551.25

123 270.6 | 1245 560.25

125 275.0 | 1265 569.25

127 279.4 | 1285 578.25

129 283.8 | 1305 587.25

131 288.2 | 1325 596.25

133 292.6 | 1345 605.25

135 297.0 | 1365 614.25

137 301.4 | 1385 623.25

139 305.8 | 1405 632.25

141 310.2 | 1425 641.25

143 314.6 | 1445 650.25

145 319.0 | 1465 659.25

147 323.4 | 1485 668.25

149 327.8 | 1505 677.25

151 332.2 | 1525 686.25

153 336.6 | 1545 695.25

155 341.0 | 1565 704.25

157 345.4 | 1585 713.25

159 349.8 | 1605 722.25

161 354.2 | 1625 731.25

163 358.6 | 1645 740.25

165 363.0 | 1665 749.25

167 367.4 | 1685 758.25

169 371.8 | 1705 767.25

171 376.2 | 1725 776.25

173 380.6 | 1745 785.25

175 385.0 | 1765 794.25

177 389.4 | 1785 803.25

179 393.8 | 1805 812.25

181 398.2 | 1825 821.25

183 402.6 | 1845 830.25

185 407.0 | 1865 839.25

187 411.4 | 1885 848.25

189 415.8 | 1905 857.25

191 420.2 | 1925 866.25

193 424.6 | 1945 875.25

195 429.0 | 1965 884.25

197 433.4 | 1985 893.25

199 437.8 | 2005 902.25

# 5.6

print(f'{"Miles":<10} {"Kilometers":<10} | {"Kilometers":<10} {"Miles"}')

for i in range(1, 11):

kilometers = i \* 1.609

miles = (i \* 5 + 15) \* 0.621

print(f'{i:<10} {kilometers:<10.3f} | {(i\*5 + 15):<10} {miles:.3f}')

Miles Kilometers | Kilometers Miles

1 1.609 | 20 12.420

2 3.218 | 25 15.525

3 4.827 | 30 18.630

4 6.436 | 35 21.735

5 8.045 | 40 24.840

6 9.654 | 45 27.945

7 11.263 | 50 31.050

8 12.872 | 55 34.155

9 14.481 | 60 37.260

10 16.090 | 65 40.365

# 5.7

import math

print(f'{"Degree":<10} {"Sin":<10} {"Cos"}')

for i in range(0, 361, 10):

radians = math.radians(i)

sin\_value = math.sin(radians)

cos\_value = math.cos(radians)

print(f'{i:<10} {sin\_value:<10.4f} {cos\_value:.4f}')

Degree Sin Cos

0 0.0000 1.0000

10 0.1736 0.9848

20 0.3420 0.9397

30 0.5000 0.8660

40 0.6428 0.7660

50 0.7660 0.6428

60 0.8660 0.5000

70 0.9397 0.3420

80 0.9848 0.1736

90 1.0000 0.0000

100 0.9848 -0.1736

110 0.9397 -0.3420

120 0.8660 -0.5000

130 0.7660 -0.6428

140 0.6428 -0.7660

150 0.5000 -0.8660

160 0.3420 -0.9397

170 0.1736 -0.9848

180 0.0000 -1.0000

190 -0.1736 -0.9848

200 -0.3420 -0.9397

210 -0.5000 -0.8660

220 -0.6428 -0.7660

230 -0.7660 -0.6428

240 -0.8660 -0.5000

250 -0.9397 -0.3420

260 -0.9848 -0.1736

270 -1.0000 -0.0000

280 -0.9848 0.1736

290 -0.9397 0.3420

300 -0.8660 0.5000

310 -0.7660 0.6428

320 -0.6428 0.7660

330 -0.5000 0.8660

340 -0.3420 0.9397

350 -0.1736 0.9848

360 -0.0000 1.0000

# 5.8

import math

print(f'{"Number":<10} {"Square Root"}')

for i in range(0, 21, 2):

sqrt\_value = math.sqrt(i)

print(f'{i:<10} {sqrt\_value:.4f}')

Number Square Root

0 0.0000

2 1.4142

4 2.0000

6 2.4495

8 2.8284

10 3.1623

12 3.4641

14 3.7417

16 4.0000

18 4.2426

20 4.4721

# 5.9

tuition = 10000

year = 0

while year < 10:

tuition \*= 1.05

year += 1

print(f"Tuition in 10 years: ${tuition:.2f}")

total\_cost = 0

while year < 14:

tuition \*= 1.05

total\_cost += tuition

year += 1

print(f"Total costs for 4 years of education that started 10 years later: ${total\_cost:.2f}")

Tuition in 10 years: $16288.95

Total cost for 4 years' worth of tuition starting 10 years from now: $73717.76

# 5.10

# "score.txt" file elimizde olmadığı için buradad farklı bir yola gidilip direkt kullanıcıdan bilgi alınmaya çalışıldı.

number\_of\_students = int(input("Enter the number of students: "))

highest\_score = 0

# Buradaki döngü yapısıyla verilen sayı kadar dönüş yapılır.

for \_ in range(number\_of\_students):

score = int(input("Enter a student's score: "))

if score > highest\_score:

highest\_score = score

print(f"The highest score is {highest\_score}")

The highest score is 64

# 5.11

number\_of\_students = int(input("Enter the number of students: "))

highest\_score = 0

second\_highest\_score = 0

for \_ in range(number\_of\_students):

score = int(input("Enter a student's score: "))

if score > highest\_score:

second\_highest\_score = highest\_score

highest\_score = score

elif score > second\_highest\_score:

second\_highest\_score = score

print(f"The highest score is {highest\_score}")

print(f"The second highest score is {second\_highest\_score}")

The highest score is 99

The second highest score is 54

# 5.12

count = 0

for number in range(100, 1001):

if number % 5 == 0 and number % 6 == 0:

print(number, end=' ')

count += 1

if count % 10 == 0:

print()

120 150 180 210 240 270 300 330 360 390

420 450 480 510 540 570 600 630 660 690

720 750 780 810 840 870 900 930 960 990

# 5.13

count = 0

for number in range(100, 201):

if (number % 5 == 0 or number % 6 == 0) and not (number % 5 == 0 and number % 6 == 0):

print(number, end=' ')

count += 1

if count % 10 == 0:

print()

100 102 105 108 110 114 115 125 126 130

132 135 138 140 144 145 155 156 160 162

165 168 170 174 175 185 186 190 192 195

198 200

# 5.14

n = 0

while n\*\*2 <= 12000:

n += 1

print(f"The smallest integer n such that n^2 is greater than 12,000 is {n}")

The smallest integer n such that n^2 is greater than 12,000 is 110

# 5.15

n = 0

while (n+1)\*\*3 < 12000:

n += 1

print(f"The largest integer n such that n^3 is less than 12,000 is {n}")

The largest integer n such that n^3 is less than 12,000 is 22

Yorum satırlarının ve kod içi yazımların İngilizce olması doğru kod yazımında alıştırma olması için eklenmiştir.

# 10.1

def assignGrades(scores):

best = max(scores) # First we assign the best score.

grades = []

for score in scores: # Then we scan all the scores again and grade them.

if score >= best - 10:

grades.append('A')

# The .append() function is used for adding an item to the end of an existing list.

elif score >= best - 20:

grades.append('B')

elif score >= best - 30:

grades.append('C')

elif score >= best - 40:

grades.append('D')

else:

grades.append('F')

return grades # The return key is a structure used when writing functions with parameters.

scores = [86, 43, 57, 71]

grades = assignGrades(scores)

for i, grade in enumerate(grades):

'''

The enumerate() function is a powerful tool, we used this tool for repeating a

collection (list, bunch or string) while keeping track of the index of each item.

'''

print(f'Student {i} score is {scores[i]} and grade is {grade}')

Student 0 score is 86 and grade is A

Student 1 score is 43 and grade is F

Student 2 score is 57 and grade is C

Student 3 score is 71 and grade is B

# 10.2

'''

I tried to do it in a long way, but when I did some research, I saw it was a special method used for reverse a list.

In ths problem we used the same powerful tool; enumarate(), thus the code is short and clear.

'''

def readIntegerList(numbers):

for i, num in enumerate(reversed(numbers)):

print(f"{num} is at index {i}")

myList = [1, 2, 3, 4, 5, 6, 7, 8, 9]

readIntegerList(myList)

9 is at index 0

8 is at index 1

7 is at index 2

6 is at index 3

5 is at index 4

4 is at index 5

3 is at index 6

2 is at index 7

1 is at index 8

# 10.3

def countNumbers(numbers):

counts = {} # We used dictionary and will be used store to count of occurrences of each number.

for num in numbers: # This "for" loop that iterates through each number in the input list "numbers".

'''This "if else conditions" check if the current number already exists in the "counts" dictionary, if so, increment its occurrence count

and the current number does not exist in the "counts" dictionary, add it with an occurrence count of 1'''

if num in counts:

counts[num] += 1

else:

counts[num] = 1

# print (counts.keys()) # Control Area

# print (counts.values())

return counts

print("Enter integers between 1 and 100 (separated by spaces!):")

user\_input = input().split() # Here we take user inputs, splits it into individual elements, stores in the variable "user\_input"

numbers = []

'''This "for" loop converts each element to an integer using "int()" and appends each element to "numbers()" '''

for num in user\_input:

num\_int = int(num)

if 1 <= num\_int <= 100:

numbers.append(num\_int)

occurrences = countNumbers(numbers) # Here we call function that we write with "numbers()" list and we send the returned "counts" to "occourrences"

for num, count in occurrences.items(): # And lastly we give the user the output

print(f"{num} occurs {count} times")

Enter integers between 1 and 100 (separated by spaces!):

1 occurs 4 times

2 occurs 6 times

3 occurs 3 times

4 occurs 2 times

5 occurs 1 times

6 occurs 2 times

7 occurs 2 times

8 occurs 1 times

9 occurs 1 times

11 occurs 2 times

12 occurs 2 times

13 occurs 1 times

31 occurs 1 times

22 occurs 1 times

import random

#10.1

def grade\_assignment(scores):

best\_score = max(scores)

grades = []

for score in scores:

if score >= best\_score - 10:

grades.append('A')

elif score >= best\_score - 20:

grades.append('B')

elif score >= best\_score - 30:

grades.append('C')

elif score >= best\_score - 40:

grades.append('D')

else:

grades.append('F')

return grades

#10.2

def reverse\_display(integer\_list):

reversed\_list = integer\_list[::-1]

for integer in reversed\_list:

print(integer)

#10.3

occurrences = {}

while True:

number = int(input("1 ile 100 arasında bir tamsayı girin (0 çıkış): "))

if number == 0:

break

if 1 <= number <= 100:

occurrences[number] = occurrences.get(number, 0) + 1

else:

print("Geçersiz giriş! Lütfen 1 ile 100 arasında bir sayı girin.")

print("\nGirilen sayıların tekrar sayıları:")

for number, count in occurrences.items():

print(f"{number}: {count} kez")

#10.4

scores = list(map(int, input("Puanları boşluklarla ayırarak girin: ").split()))

total\_score = sum(scores)

num\_scores = len(scores)

average\_score = total\_score / num\_scores

above\_average\_count = sum(score >= average\_score for score in scores)

below\_average\_count = num\_scores - above\_average\_count

print(f"Ortalama: {average\_score:.2f}")

print(f"Ortalamanın üstünde veya eşit puan sayısı: {above\_average\_count}")

print(f"Ortalamadan küçük puan sayısı: {below\_average\_count}")

#10.5

numbers = input("Sayıları boşluklarla ayırarak girin: ").split()

distinct\_numbers = sorted(set(numbers), key=int)

print("Tekrar edilmeyen sayılar:")

for number in distinct\_numbers:

print(number, end=" ")

#10.6

#10.7

counts = [0] \* 10

for \_ in range(1000):

random\_number = random.randint(0, 9)

counts[random\_number] += 1

for i in range(10):

print(f"{i}: {counts[i]} adet")

#10.8

def indexOfSmallestElement(lst):

min\_value = min(lst)

return lst.index(min\_value)

numbers = [5, 2, 7, 1, 9, 3]

print("En küçük elemanın indeksi:", indexOfSmallestElement(numbers))

#10.9

#10.10

def reverseList(lst):

return lst[::-1]

def main():

numbers = input("Sayı listesini boşluklarla ayırarak girin: ").split()

numbers = [int(num) for num in numbers]

reversed\_numbers = reverseList(numbers)

print("Girilen sayılar:", numbers)

print("Tersine çevrilmiş sayılar:", reversed\_numbers)

main()

#10.11

def shuffle(lst):

shuffled\_list = lst[:]

for i in range(len(shuffled\_list)):

j = random.randint(0, i)

shuffled\_list[i], shuffled\_list[j] = shuffled\_list[j], shuffled\_list[i]

return shuffled\_list

def main():

numbers = input("Sayı listesini boşluklarla ayırarak girin: ").split()

numbers = [int(num) for num in numbers]

shuffled\_numbers = shuffle(numbers)

print("Girilen sayılar:", numbers)

print("Karıştırılmış sayılar:", shuffled\_numbers)

main()

#10.12

def gcd(numbers):

def find\_gcd(x, y):

while y:

x, y = y, x % y

return x

if len(numbers) < 2:

raise ValueError("Listede en az iki sayı olmalıdır.")

result = find\_gcd(numbers[0], numbers[1])

for num in numbers[2:]:

result = find\_gcd(result, num)

return result

def main():

numbers = [int(input(f"{i+1}. sayıyı girin: ")) for i in range(5)]

greatest\_common\_divisor = gcd(numbers)

print("Girilen sayıların en büyük ortak böleni:", greatest\_common\_divisor)

main()

#10.13

def eliminateDuplicates(lst):

unique\_list = list(set(lst))

return unique\_list

def main():

numbers = input("Tamsayı listesini boşluklarla ayırarak girin: ").split()

numbers = [int(num) for num in numbers]

unique\_numbers = eliminateDuplicates(numbers)

print("Tekrarlanan değerlerden arındırılmış liste:", unique\_numbers)

main()

#10.14

def selection\_sort(lst):

n = len(lst)

for i in range(n-1, 0, -1):

max

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#10.1

def grade\_assignment(scores):

best\_score = max(scores)

grades = []

for score in scores:

if score >= best\_score - 10:

grades.append('A')

elif score >= best\_score - 20:

grades.append('B')

elif score >= best\_score - 30:

grades.append('C')

elif score >= best\_score - 40:

grades.append('D')

else:

grades.append('F')

return grades

# Kullanıcıdan not listesini al

score\_list = list(map(int, input("Not listesini girin (virgülle ayırarak): ").split(',')))

# Notları atayıp ekrana yazdır

assigned\_grades = grade\_assignment(score\_list)

print("Notlar:", assigned\_grades)

#10.2

def reverse\_display(integer\_list):

reversed\_list = integer\_list[::-1]

for integer in reversed\_list:

print(integer)

# Kullanıcıdan tamsayı listesini al

integer\_list = list(map(int, input("Tamsayı listesini girin (virgülle ayırarak): ").split(',')))

# Tersine çevrilmiş listeyi görüntüle

print("Okuma sırasının tersine göre tamsayılar:")

reverse\_display(integer\_list)

#10.3

# Boş bir sözlük oluşturuluyor

occurrences = {}

# Kullanıcıdan sayı girişi alınıyor

while True:

number = int(input("1 ile 100 arasında bir tamsayı girin (0 çıkış): "))

if number == 0:

break

if 1 <= number <= 100:

# Girilen sayının sözlükteki değeri artırılıyor

occurrences[number] = occurrences.get(number, 0) + 1

else:

print("Geçersiz giriş! Lütfen 1 ile 100 arasında bir sayı girin.")

# Sonuçlar ekrana yazdırılıyor

print("\nGirilen sayıların tekrar sayıları:")

for number, count in occurrences.items():

print(f"{number}: {count} kez")

#10.4

# Kullanıcıdan puanları bir dize olarak alıyoruz ve boşluklardan bölerek bir liste oluşturuyoruz

scores = input("Puanları boşluklarla ayırarak girin: ").split()

# Puanları integer'a dönüştürüyoruz

scores = [int(score) for score in scores]

# Puanların toplamını ve puan sayısını hesaplıyoruz

total\_score = sum(scores)

num\_scores = len(scores)

# Ortalamayı hesaplıyoruz

average\_score = total\_score / num\_scores

# Ortalamadan büyük veya eşit puanların sayısını ve küçük puanların sayısını sayıyoruz

above\_average\_count = sum(score >= average\_score for score in scores)

below\_average\_count = num\_scores - above\_average\_count

# Sonuçları ekrana yazdırıyoruz

print(f"Ortalama: {average\_score:.2f}")

print(f"Ortalamanın üstünde veya eşit puan sayısı: {above\_average\_count}")

print(f"Ortalamadan küçük puan sayısı: {below\_average\_count}")

#10.5

# Kullanıcıdan sayıları bir dize olarak alıyoruz ve boşluklardan bölerek bir liste oluşturuyoruz

numbers = input("Sayıları boşluklarla ayırarak girin: ").split()

# Sayıları bir küme (set) veri yapısına dönüştürerek yinelenenleri kaldırıyoruz

distinct\_numbers = set(numbers)

# Kümeyi listeye dönüştürerek sıralı bir şekilde elde ediyoruz

distinct\_numbers = sorted(distinct\_numbers, key=int)

# Distinct numbers'ları ekrana yazdırıyoruz

print("Tekrar edilmeyen sayılar:")

for number in distinct\_numbers:

print(number, end=" ")

#10.6 - Not provided

#10.7

import random

# 0'dan 9'a kadar olan sayıların sayılarını tutacak bir liste oluşturuyoruz

counts = [0] \* 10

# 1,000 rastgele tamsayı oluşturup listedeki ilgili indeksi artırıyoruz

for \_ in range(1000):

random\_number = random.randint(0, 9)

counts[random\_number] += 1

# Sonuçları ekrana yazdırıyoruz

for i in range(10):

print(f"{i}: {counts[i]} adet")

#10.8

def indexOfSmallestElement(lst):

# En küçük değerin indeksini bulmak için min() fonksiyonunu kullanıyoruz

min\_value = min(lst)

# En küçük değerin indeksini döndürüyoruz

return lst.index(min\_value)

# Fonksiyonu kullanarak bir örnek

numbers = [5, 2, 7, 1, 9, 3]

print("En küçük elemanın indeksi:", indexOfSmallestElement(numbers))

#10.9 - Not provided

#10.10

def reverseList(lst):

return lst[::-1]

def main():

# Kullanıcıdan sayı listesini al

numbers = input("Sayı listesini boşluklarla ayırarak girin: ").split()

numbers = [int(num) for num in numbers] # Girdiyi tamsayıya dönüştür

# Sayıları tersine çevir

reversed\_numbers = reverseList(numbers)

# Sonuçları ekrana yazdır

print("Girilen sayılar:", numbers)

print("Tersine çevrilmiş sayılar:", reversed\_numbers)

# Test programını çağır

main()

#10.11

import random

def shuffle(lst):

shuffled\_list = lst[:] # Orjinal listeyi değiştirmemek için kopyasını oluştur

for i in range(len(shuffled\_list)):

# Rastgele bir indeks seç ve bu indeksteki elemanı şu andaki indeksle değiştir

j = random.randint(0, i)

shuffled\_list[i], shuffled\_list[j] = shuffled\_list[j], shuffled\_list[i]

return shuffled\_list

def main():

# Kullanıcıdan sayı listesini al

numbers = input("Sayı listesini boşluklarla ayırarak girin: ").split()

numbers = [int(num) for num in numbers] # Girdiyi tamsayıya dönüştür

# Sayıları karıştır

shuffled\_numbers = shuffle(numbers)

# Sonuçları ekrana yazdır

print("Girilen sayılar:", numbers)

print("Karıştırılmış sayılar:", shuffled\_numbers)

# Test programını çağır

main()

#10.12

def gcd(numbers):

# İlk iki sayının GCD'sini bulmak için yardımcı bir fonksiyon tanımlayalım

def find\_gcd(x, y):

while y:

x, y = y, x % y

return x

# Listede en az iki eleman olduğundan emin olalım

if len(numbers) < 2:

raise ValueError("Listede en az iki sayı olmalıdır.")

# Listede en az iki eleman olduğu için ilk iki elemanın GCD'sini alalım

result = find\_gcd(numbers[0], numbers[1])

# GCD'yi diğer elemanlarla tekrar tekrar bulalım

for num in numbers[2:]:

result = find\_gcd(result, num)

return result

def main():

# Kullanıcıdan beş sayı alalım

numbers = [int(input(f"{i+1}. sayıyı girin: ")) for i in range(5)]

# Sayıların en büyük ortak bölenini bulalım

greatest\_common\_divisor = gcd(numbers)

# Sonucu ekrana yazdıralım

print("Girilen sayıların en büyük ortak böleni:", greatest\_common\_divisor)

# Test programını çağıralım

main()

#10.13

def eliminateDuplicates(lst):

# Liste elemanlarını bir küme içine koyarak tekrarlayanları kaldıralım

unique\_list = list(set(lst))

return unique\_list

def main():

# Kullanıcıdan bir tamsayı listesi alalım

numbers = input("Tamsayı listesini boşluklarla ayırarak girin: ").split()

numbers = [int(num) for num in numbers] # Girdiyi tamsayıya dönüştür

# Tekrarlanan değerleri kaldıralım

unique\_numbers = eliminateDuplicates(numbers)

# Sonucu ekrana yazdıralım

print("Tekrarlanan değerlerden arındırılmış liste:", unique\_numbers)

# Test programını çağıralım

main()

#10.14

def selection\_sort(lst):

n = len(lst)

for i in range(n-1, 0, -1):

max\_index = i

for j in range(i):

if lst[j] > lst[max\_index]:

max\_index = j

lst[max\_index], lst[i] = lst[i], lst[max\_index]

def main():

numbers = [float(input(f"Enter number {i+1}: ")) for i in range(10)]

selection\_sort(numbers)

print("Sorted numbers:", numbers)

main()

#10.15

def isSorted(lst):

# Check if the list is sorted in increasing order

return all(lst[i] <= lst[i+1] for i in range(len(lst)-1))

def main():

# Prompt the user to enter a list

user\_input = input("Enter a list of numbers separated by spaces: ")

numbers = list(map(float, user\_input.split()))

# Check if the list is sorted

if isSorted(numbers):

print("The list is sorted in increasing order.")

else:

print("The list is not sorted in increasing order.")

# Test the program

main()

import math

# 11.1

def sumColumn(matris, sutun\_indeksi):

toplam = 0

for satir in matris:

toplam += satir[sutun\_indeksi]

return toplam

def main():

# Matrisin boyutlarını kullanıcıdan al

satir\_sayisi = int(input("Matrisin satır sayısını girin: "))

sutun\_sayisi = int(input("Matrisin sütun sayısını girin: "))

# Boş bir matris oluştur

matris = []

# Matris elemanlarını kullanıcıdan al

print("Matris elemanlarını satır satır girin:")

for i in range(satir\_sayisi):

satir = []

for j in range(sutun\_sayisi):

eleman = float(input(f"Eleman [{i}][{j}]: "))

satir.append(eleman)

matris.append(satir)

# Matrisi ekrana yazdır

print("\nMatris:")

for satir in matris:

print(satir)

# Her sütunun toplamını hesapla ve ekrana yazdır

for j in range(sutun\_sayisi):

sutun\_toplami = sumColumn(matris, j)

print(f"{j}. sütunun toplamı: {sutun\_toplami}")

# Programı çalıştır

main()

# 11.2

def sumMajorDiagonal(matris):

toplam = 0

for i in range(len(matris)):

toplam += matris[i][i]

return toplam

def main():

# Matrisin boyutlarını kullanıcıdan al

boyut = int(input("Kare matrisin boyutunu girin: "))

# Boş bir matris oluştur

matris = []

# Matris elemanlarını kullanıcıdan al

print("Matris elemanlarını girin:")

for i in range(boyut):

satir = []

for j in range(boyut):

eleman = int(input(f"Eleman [{i}][{j}]: "))

satir.append(eleman)

matris.append(satir)

# Matrisi ekrana yazdır

print("\nMatris:")

for satir in matris:

print(satir)

# Ana diyagonaldeki elemanların toplamını hesapla ve ekrana yazdır

ana\_diyagonal\_toplami = sumMajorDiagonal(matris)

print(f"\nAna diyagonaldeki elemanların toplamı: {ana\_diyagonal\_toplami}")

# Programı çalıştır

main()

# 11.3

def display\_students\_in\_order(students):

sorted\_students = sorted(students, key=lambda x: x[1])

print("Öğrencilerin doğru cevap sayısına göre sıralanmış listesi:")

for student in sorted\_students:

print(f"Öğrenci {student[0]}: {student[1]} doğru cevap")

def main():

students = []

while True:

student\_name = input("Öğrencinin adını girin (Çıkmak için 'q' tuşuna basın): ")

if student\_name == 'q':

break

correct\_answers = int(input(f"{student\_name} için doğru cevap sayısını girin: "))

students.append((student\_name, correct\_answers))

display\_students\_in\_order(students)

# Programı çalıştır

main()

# 11.4

def display\_employees\_in\_order(hours\_table):

total\_hours = [(i, sum(hours)) for i, hours in enumerate(hours\_table)]

sorted\_hours = sorted(total\_hours, key=lambda x: x[1], reverse=True)

print("Çalışanlar ve toplam saatleri:")

for employee, total\_hour in sorted\_hours:

print(f"Çalışan {employee + 1}: {total\_hour} saat")

def main():

# Örnek olarak 8 çalışanın haftalık saatleri

hours\_table = [

[40, 35, 45, 37, 38, 40, 42],

[38, 36, 42, 41, 39, 40, 41],

[37, 34, 45, 36, 38, 40, 42],

[39, 38, 44, 40, 36, 40, 41],

[40, 39, 46, 37, 40, 40, 40],

[40, 40, 42, 38, 38, 40, 42],

[40, 35, 44, 40, 38, 40, 43],

[39, 37, 43, 39, 38, 40, 40]

]

display\_employees\_in\_order(hours\_table)

# Programı çalıştır

main()

# 11.5

def findLargest(row):

return max(row)

def main():

# Matrisin boyutlarını kullanıcıdan al

satir\_sayisi = int(input("Matrisin satır sayısını girin: "))

sutun\_sayisi = int(input("Matrisin sütun sayısını girin: "))

# Boş bir matris oluştur

matris = []

# Matris elemanlarını kullanıcıdan al

print("Matris elemanlarını satır satır girin:")

for i in range(satir\_sayisi):

satir = []

for j in range(sutun\_sayisi):

eleman = int(input(f"Eleman [{i}][{j}]: "))

satir.append(eleman)

matris.append(satir)

# Matrisi ekrana yazdır

print("\nMatris:")

for satir in matris:

print(satir)

# Her satırdaki en büyük elemanı bul ve ekrana yazdır

for i, satir in enumerate(matris):

en\_buyuk = findLargest(satir)

print(f"{i}. satırdaki en büyük eleman: {en\_buyuk}")

# Programı çalıştır

main()

# 11.6

def displaySortedNumbers(num1, num2, num3):

sorted\_numbers = sorted([num1, num2, num3])

for number in sorted\_numbers:

print(number, end=" ")

def main():

# Kullanıcıdan üç sayı al

num1 = float(input("1. Sayıyı girin: "))

num2 = float(input("2. Sayıyı girin: "))

num3 = float(input("3. Sayıyı girin: "))

# Sayıları sırala ve ekrana yazdır

print("Sıralanmış sayılar:")

displaySortedNumbers(num1, num2, num3)

# Programı çalıştır

main()

# 11.7

def distance(x1, y1, x2, y2):

return ((x2 - x1) \*\* 2 + (y2 - y1) \*\* 2) \*\* 0.5

def main():

# İki noktanın koordinatlarını kullanıcıdan al

x1, y1 = eval(input("İlk noktanın koordinatlarını girin (x1, y1): "))

x2, y2 = eval(input("İkinci noktanın koordinatlarını girin (x2, y2): "))

# Mesafeyi hesapla ve ekrana yazdır

mesafe = distance(x1, y1, x2, y2)

print(f"İki nokta arasındaki mesafe: {mesafe}")

# Programı çalıştır

main()

# 11.8

def reverse(number):

reversed\_number = 0

while number != 0:

son\_basamak = number % 10

reversed\_number = reversed\_number \* 10 + son\_basamak

number //= 10

return reversed\_number

def main():

# Kullanıcıdan bir sayı al

sayi = int(input("Bir sayı girin: "))

# Sayının tersini hesapla ve ekrana yazdır

ters\_sayi = reverse(sayi)

print(f"Girilen sayının tersi: {ters\_sayi}")

# Programı çalıştır

main()

# 11.9

def isPrime(number):

if number <= 1:

return False

for i in range(2, number):

if number % i == 0:

return False

return True

def main():

# Kullanıcıdan bir sayı al

sayi = int(input("Bir sayı girin: "))

# Sayının asal olup olmadığını kontrol et ve sonucu ekrana yazdır

if isPrime(sayi):

print(f"{sayi} sayısı asaldır.")

else:

print(f"{sayi} sayısı asal değildir.")

# Programı çalıştır

main()

# 11.10

def displayPattern(n):

for i in range(1, n + 1):

# Boşlukları ekle

for j in range(n - i):

print(" ", end="")

# Sayıları ekle

for j in range(i, 0, -1):

print(j, end=" ")

print()

def main():

# Kullanıcıdan bir sayı al

n = int(input("Bir sayı girin: "))

# Deseni ekrana yazdır

displayPattern(n)

# Programı çalıştır

main()

# 11.11

def getNumbers():

numbers = []

while True:

num = int(input("Bir sayı girin (0 çıkış): "))

if num == 0:

break

numbers.append(num)

return numbers

def mean(numbers):

return sum(numbers) / len(numbers)

def deviation(numbers):

m = mean(numbers)

return (sum([(x - m) \*\* 2 for x in numbers]) / (len(numbers) - 1)) \*\* 0.5

def main():

# Sayıları al

numbers = getNumbers()

# Ortalama ve sapmayı hesapla

m = mean(numbers)

dev = deviation(numbers)

# Sonuçları ekrana yazdır

print(f"Ortalama: {m:.2f}")

print(f"Sapma: {dev:.5f}")

# Programı çalıştır

main()

# 11.12

def reverse(number):

reversed\_num = 0

while number != 0:

remainder = number % 10

reversed\_num = reversed\_num \* 10 + remainder

number = number // 10

return reversed\_num

def main():

# Kullanıcıdan bir sayı al

num = int(input("Bir sayı girin: "))

# Sayıyı tersine çevir

reversed\_num = reverse(num)

# Sonucu ekrana yazdır

print(f"Girdiğiniz sayının tersi: {reversed\_num}")

# Programı çalıştır

main()

# 11.13

def checkPalindrome(word):

return word == word[::-1]

def main():

# Kullanıcıdan bir kelime al

word = input("Bir kelime girin: ")

# Palindrom kontrolü yap

if checkPalindrome(word):

print("Bu kelime bir palindromdur.")

else:

print("Bu kelime bir palindrom değildir.")

# Programı çalıştır

main()

# 11.14

def countLetters(word):

letter\_count = {}

for letter in word:

if letter.isalpha():

letter = letter.lower()

if letter in letter\_count:

letter\_count[letter] += 1

else:

letter\_count[letter] = 1

return letter\_count

def main():

# Kullanıcıdan bir kelime al

word = input("Bir kelime girin: ")

# Harf sayısını hesapla

letter\_count = countLetters(word)

# Sonucu ekrana yazdır

for letter, count in letter\_count.items():

print(f"'{letter}' harfi {count} kez geçiyor.")

# Programı çalıştır

main()

# 11.15

def sumDigits(n):

total = 0

while n > 0:

total += n % 10

n //= 10

return total

def main():

# Kullanıcıdan bir sayı al

num = int(input("Bir sayı girin: "))

# Rakamların toplamını hesapla

digit\_sum = sumDigits(num)

# Sonucu ekrana yazdır

print(f"Girdiğiniz sayının rakamları toplamı: {digit\_sum}")

# Programı çalıştır

main()