Python Exercise - 8

In [1]:
```
#importing libraries
import numpy as np
import pandas as pd
```

1)WA Python function to check if a given number is prime or not

In [14]:
```
def is_prime(number):
    if number <= 1:
        return False
    elif number == 2:
        return True
    elif number % 2 == 0:
        return False


    # We only need to check up to the square root of the number.
    # If there are no divisors up to the square root, there will be no divisors above it.
    for i in range(3, int(number**0.5) + 1, 2):
        if number % i == 0:
            return False


    return True

# Get user input for the number to check
num = int(input("Enter the number: "))

if is_prime(num):
    print(f"{num} is a prime number.")
else:
    print(f"{num} is not a prime number.")
```

```
Enter the number: 89
89 is a prime number.
```

2)WA Python single function that returns the result of addition, subtraction, multiplication, division

In [15]:
```
def perform_operations(a, b):
    addition = a + b
    subtraction = a - b
    multiplication = a * b

    # To avoid division by zero, check if b is not zero before performing division.
    if b != 0:
        division = a / b
    else:
        division = None

    return addition, subtraction, multiplication, division

# Test the function
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))

result_add, result_sub, result_mul, result_div = perform_operations(num1, num2)

print(f"Addition: {result_add}")
print(f"Subtraction: {result_sub}")
print(f"Multiplication: {result_mul}")
if result_div is not None:
    print(f"Division: {result_div}")
else:
    print("Division by zero is not allowed.")
```

```
Enter the first number: 78
Enter the second number: 79
Addition: 157.0
Subtraction: -1.0
Multiplication: 6162.0
Division: 0.9873417721518988
```

3)WAPP to find sum of squares of list of numbers using calling a function from another function

In [24]:
```
def square(num):
    return num ** 2

def sum_of_squares(numbers):
    total_sum = 0
    for num in numbers:
        total_sum += square(num)
    return total_sum

# Test the function
def main():
    try:
        num_list = input("Enter a list of numbers separated by comma: ")
        num_list = [int(num) for num in num_list.split(',')]
        result = sum_of_squares(num_list)
        print("Sum of squares:", result)
    except ValueError:
        print("Invalid input. Please enter a list of numbers separated by comma.")

if __name__ == "__main__":
    main()
```

```
Enter a list of numbers separated by comma: 4,6,9
Sum of squares: 133
```

4)WA python function to demonstrate passing a variable number of arguments

In [25]:
```
def variable_arguments_example(*args, **kwargs):
    print("Number of arguments:", len(args))
    for arg in args:
        print(arg)

    print("Number of keyword arguments:", len(kwargs))
    for key, value in kwargs.items():
        print(f"{key}: {value}")

# Test the function with different arguments
variable_arguments_example(1, 2, 3)
variable_arguments_example('apple', 'banana', 'orange', 'grape')
variable_arguments_example(name='John', age=30)
variable_arguments_example(country='USA', city='New York', population=8378460)
variable_arguments_example(1, 2, 3, name='Alice', age=25)
```

```
Number of arguments: 3
1
2
3
Number of keyword arguments: 0
Number of arguments: 4
apple
banana
orange
grape
Number of keyword arguments: 0
Number of arguments: 0
Number of keyword arguments: 2
name: John
age: 30
Number of arguments: 0
Number of keyword arguments: 3
country: USA
city: New York
population: 8378460
Number of arguments: 3
1
2
3
Number of keyword arguments: 2
name: Alice
age: 25
```

5)WA python recursive function to solve towers of Hanoi problem

In [30]:
```
def hanoi(n, source, auxiliary, target):
    if n == 1:
        print(f"Move disk 1 from {source} to {target}")
        return
    else:
        hanoi(n-1, source, target, auxiliary)
        print(f"Move disk {n} from {source} to {target}")
        hanoi(n-1, auxiliary, source, target)

# Test the function
num_disks = int(input("Enter the number of disks: "))
hanoi(num_disks, 'A', 'B', 'C')
```

```
Enter the number of disks: 3
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C
```

6)WAPP to Display Fibonacci Sequence Using Recursion

In [37]:
```
def fibonacci(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    else:
        fib_seq = fibonacci(n - 1)
        fib_seq.append(fib_seq[-1] + fib_seq[-2])
        return fib_seq

# Test the function
num_terms = int(input("Enter the number of terms in the Fibonacci sequence: "))

fib_sequence = fibonacci(num_terms)
print("Fibonacci sequence:")
print(fib_sequence)
```

```
Enter the number of terms in the Fibonacci sequence: 8
Fibonacci sequence:
[0, 1, 1, 2, 3, 5, 8, 13]
```

7)WAPP to make a function that always triples the number you send in using lambda

In [38]:
```
triple_number = lambda x: x * 3

# Test the lambda function
num = int(input("Enter a number: "))
result = triple_number(num)
print(f"The triple of {num} is: {result}")
```

```
Enter a number: 7
The triple of 7 is: 21
```

8)WAPP to double all numbers using map()

In [ ]:
```

```

In [39]:
```
def double_number(num):
    return num * 2

# Test the function with a list of numbers
numbers = [1, 2, 3, 4, 5]

doubled_numbers = list(map(double_number, numbers))
print("Original Numbers:", numbers)
print("Doubled Numbers:", doubled_numbers)
```

```
Original Numbers: [1, 2, 3, 4, 5]
Doubled Numbers: [2, 4, 6, 8, 10]
```

9)WAPP to a pre-defined function product which return the product of 2 numbers passed into it & is used as an argument in the reduce function which is passed along with a list of numbers

In [40]:
```
from functools import reduce
from operator import mul

def product(x, y):
    return x * y

# Test the function with a list of numbers
numbers = [1, 2, 3, 4, 5]

result = reduce(product, numbers)
print("Product of the numbers:", result)
```

```
Product of the numbers: 120
```

10)Write a Python program which iterates the integers from 1 to 80. For multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".

In [41]:
```
for num in range(1, 81):
    if num % 3 == 0 and num % 5 == 0:
        print("FizzBuzz")
    elif num % 3 == 0:
        print("Fizz")
    elif num % 5 == 0:
        print("Buzz")
    else:
        print(num)
```

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
16
17
Fizz
19
Buzz
Fizz
22
23
Fizz
Buzz
26
Fizz
28
29
FizzBuzz
31
32
Fizz
34
Buzz
Fizz
37
38
Fizz
Buzz
41
Fizz
43
44
FizzBuzz
46
47
Fizz
49
Buzz
Fizz
52
53
Fizz
Buzz
56
Fizz
58
59
FizzBuzz
61
62
Fizz
64
Buzz
Fizz
67
68
Fizz
Buzz
71
Fizz
73
74
FizzBuzz
76
77
Fizz
```

```
79
Buzz
```

In [ ]: