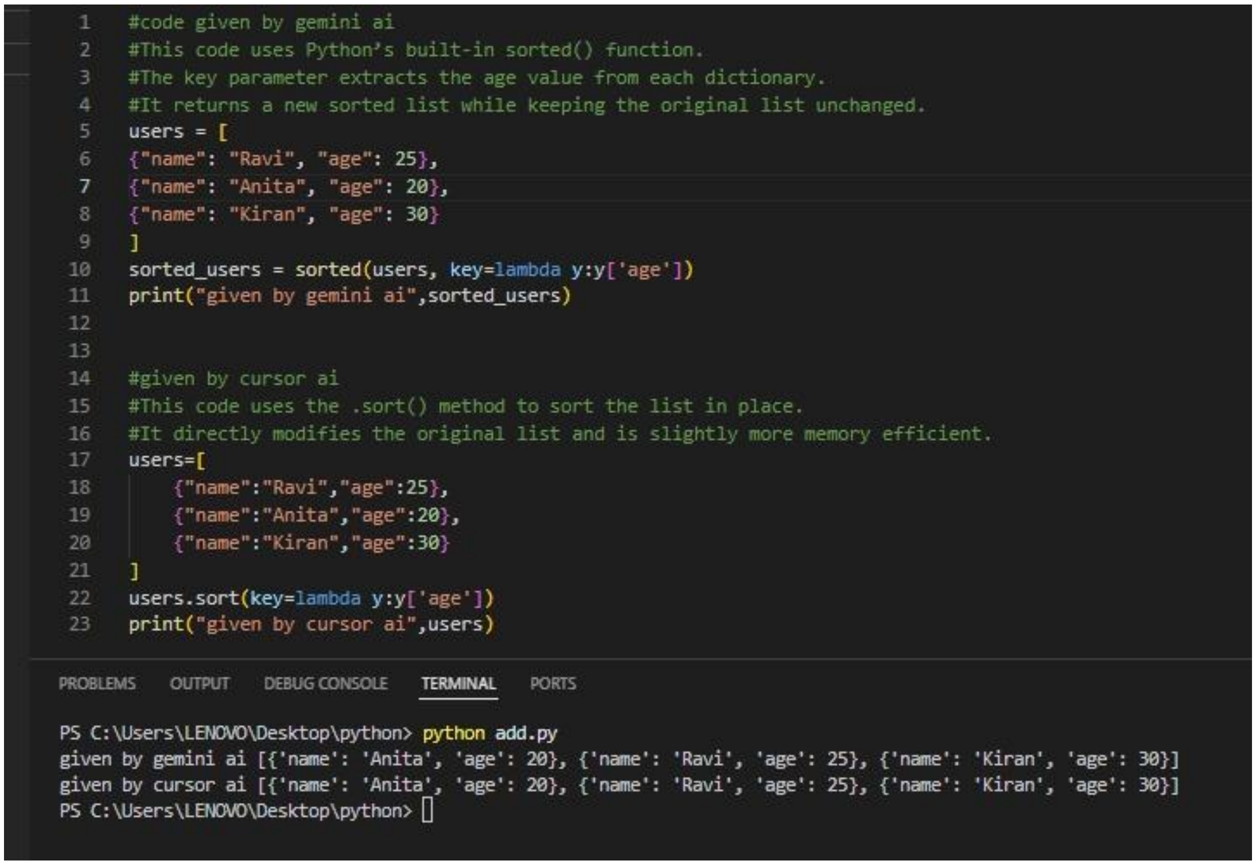
Assingment 2.4

t.akanksh

2303A51078

Task1:

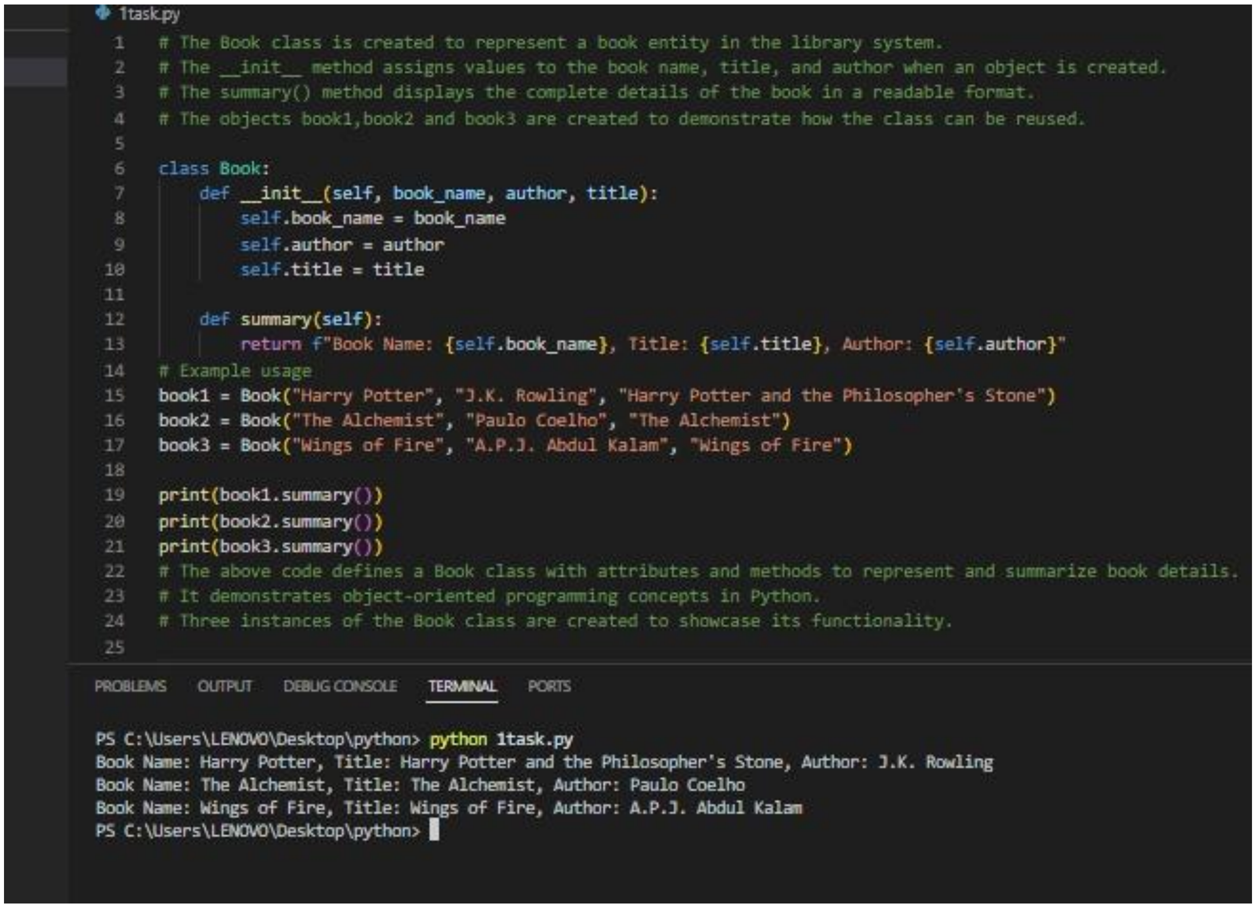


```python
#code given by gemini ai
#This code uses Python’s built-in sorted() function.
#The key parameter extracts the age value from each dictionary.
#It returns a new sorted list while keeping the original list unchanged.
users = [
{"name": "Ravi", "age": 25},
{"name": "Anita", "age": 20},
{"name": "Kiran", "age": 30}
]
sorted_users = sorted(users, key=lambda y:y['age'])
print("given by gemini ai",sorted_users)


#given by cursor ai
#This code uses the .sort() method to sort the list in place.
#It directly modifies the original list and is slightly more memory efficient.
users=[
    {"name":"Ravi","age":25},
    {"name":"Anita","age":20},
    {"name":"Kiran","age":30}
]
users.sort(key=lambda y:y['age'])
print("given by cursor ai",users)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\LENOVO\Desktop\python> python add.py
given by gemini ai [{'name': 'Anita', 'age': 20}, {'name': 'Ravi', 'age': 25}, {'name': 'Kiran', 'age': 30}]
given by cursor ai [{'name': 'Anita', 'age': 20}, {'name': 'Ravi', 'age': 25}, {'name': 'Kiran', 'age': 30}]
PS C:\Users\LENOVO\Desktop\python> 
```

Task2:

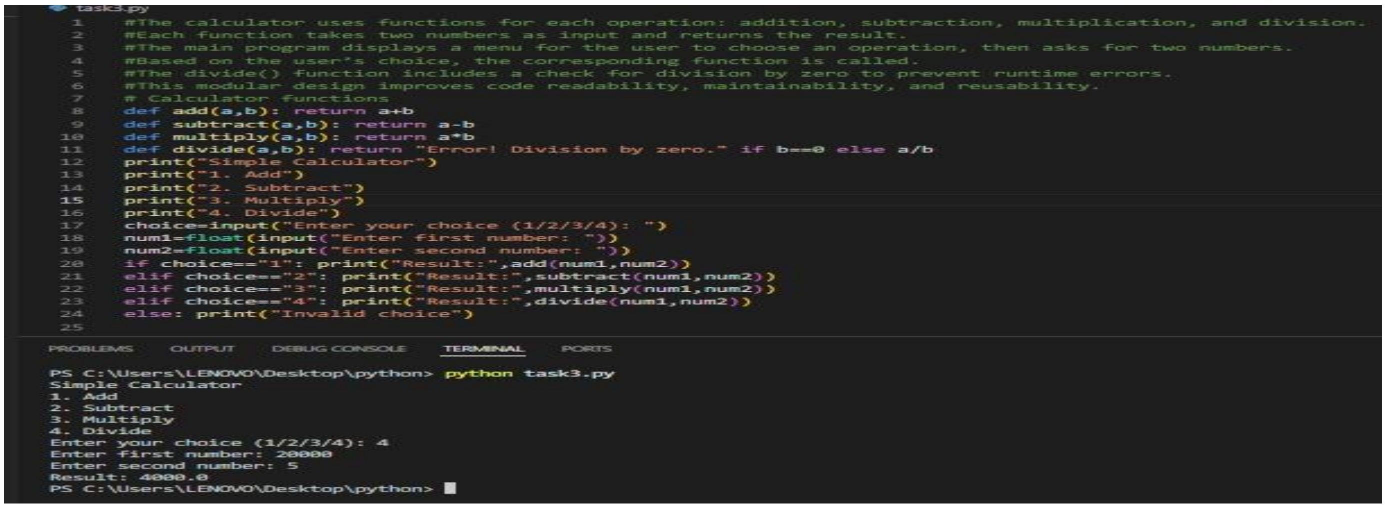


1task.py

```python
# The Book class is created to represent a book entity in the library system.
# The __init__ method assigns values to the book name, title, and author when an object is created.
# The summary() method displays the complete details of the book in a readable format.
# The objects book1,book2 and book3 are created to demonstrate how the class can be reused.

class Book:
    def __init__(self, book_name, author, title):
        self.book_name = book_name
        self.author = author
        self.title = title

    def summary(self):
        return f"Book Name: {self.book_name}, Title: {self.title}, Author: {self.author}"
# Example usage
book1 = Book("Harry Potter", "J.K. Rowling", "Harry Potter and the Philosopher's Stone")
book2 = Book("The Alchemist", "Paulo Coelho", "The Alchemist")
book3 = Book("Wings of Fire", "A.P.J. Abdul Kalam", "Wings of Fire")

print(book1.summary())
print(book2.summary())
print(book3.summary())
# The above code defines a Book class with attributes and methods to represent and summarize book details.
# It demonstrates object-oriented programming concepts in Python.
# Three instances of the Book class are created to showcase its functionality.

```

```
PS C:\Users\LENOVO\Desktop\python> python 1task.py
Book Name: Harry Potter, Title: Harry Potter and the Philosopher's Stone, Author: J.K. Rowling
Book Name: The Alchemist, Title: The Alchemist, Author: Paulo Coelho
Book Name: Wings of Fire, Title: Wings of Fire, Author: A.P.J. Abdul Kalam
PS C:\Users\LENOVO\Desktop\python>
```

task3

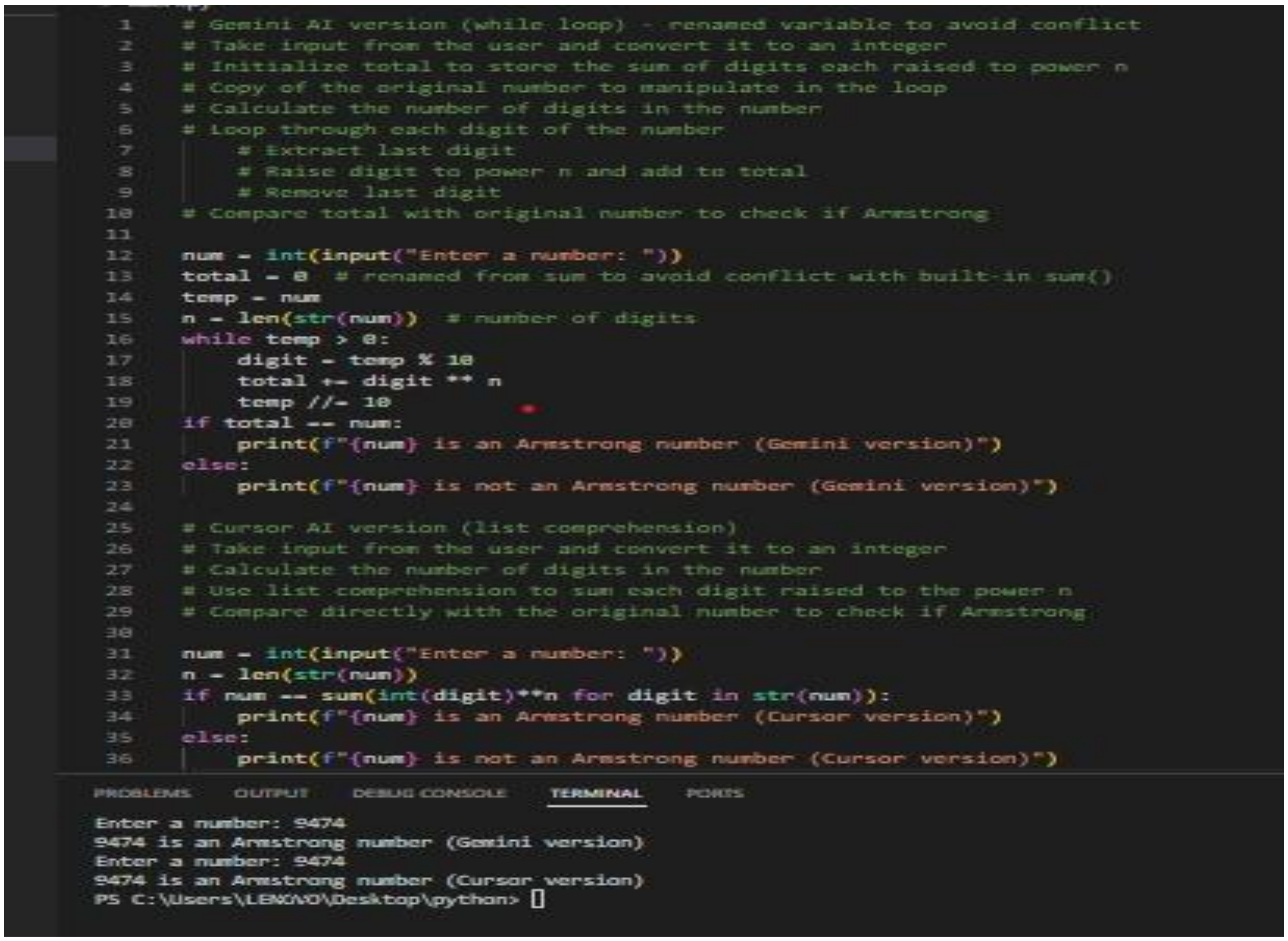


task3.py

```python
#The calculator uses functions for each operation: addition, subtraction, multiplication, and division.
#Each function takes two numbers as input and returns the result.
#The main program displays a menu for the user to choose an operation, then asks for two numbers.
#Based on the user's choice, the corresponding function is called.
#The divide() function includes a check for division by zero to prevent runtime errors.
#This modular design improves code readability, maintainability, and reusability.
# Calculator functions
def add(a,b): return a+b
def subtract(a,b): return a-b
def multiply(a,b): return a*b
def divide(a,b): return "Error! Division by zero." if b==0 else a/b
print("Simple Calculator")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")
choice=input("Enter your choice (1/2/3/4): ")
num1=float(input("Enter first number: "))
num2=float(input("Enter second number: "))
if choice=="1": print("Result:",add(num1,num2))
elif choice=="2": print("Result:",subtract(num1,num2))
elif choice=="3": print("Result:",multiply(num1,num2))
elif choice=="4": print("Result:",divide(num1,num2))
else: print("Invalid choice")

```

```
PS C:\Users\LENOVO\Desktop\python> python task3.py
Simple Calculator
1. Add
2. Subtract
3. Multiply
4. Divide
Enter your choice (1/2/3/4): 4
Enter first number: 20000
Enter second number: 5
Result: 4000.0
PS C:\Users\LENOVO\Desktop\python>
```

task4



```python
# Gemini AI version (while loop) - renamed variable to avoid conflict
# Take input from the user and convert it to an integer
# Initialize total to store the sum of digits each raised to power n
# Copy of the original number to manipulate in the loop
# Calculate the number of digits in the number
# Loop through each digit of the number
    # Extract last digit
    # Raise digit to power n and add to total
    # Remove last digit
# Compare total with original number to check if Armstrong

num = int(input("Enter a number: "))
total = 0  # renamed from sum to avoid conflict with built-in sum()
temp = num
n = len(str(num))  # number of digits
while temp > 0:
    digit = temp % 10
    total += digit ** n
    temp //= 10
if total == num:
    print(f"{num} is an Armstrong number (Gemini version)")
else:
    print(f"{num} is not an Armstrong number (Gemini version)")

# Cursor AI version (list comprehension)
# Take input from the user and convert it to an integer
# Calculate the number of digits in the number
# Use list comprehension to sum each digit raised to the power n
# Compare directly with the original number to check if Armstrong

num = int(input("Enter a number: "))
n = len(str(num))
if num == sum(int(digit)**n for digit in str(num)):
    print(f"{num} is an Armstrong number (Cursor version)")
else:
    print(f"{num} is not an Armstrong number (Cursor version)")
```

```
Enter a number: 9474
9474 is an Armstrong number (Gemini version)
Enter a number: 9474
9474 is an Armstrong number (Cursor version)
PS C:\Users\LENOVO\Desktop\python> 
```