AI ASSISTED CODING

LAB ASSIGNMENT – 6.3

Specialization	: AIML
Name of Student	: GURRALA SAI SPOORTHY
Enrollment-No	: 2403A52290
Batch No	: 01
Date	: 10-09-2025

TASK DESCRIPTION-1:

(Classes)

- Use AI to complete a Student class with attributes and a method.
- Check output
- Analyze the code generated by the AI tool

PROMPT:

GIVE A PYTHON CODE TO COMPLETE A STUDENT CLASS WITH ATTRIBUTES AND A METHOD.

CODE:

```
class Student:
    def __init__(self, name, age, grade, major):
        self.name = name
        self.age = age
        self.grade = grade
        self.major = major

    def display_info(self):
        print(f"Student Name: {self.name}")
        print(f"Student Age: {self.age}")
        print(f"Student Grade: {self.grade}")
        print(f"Student Major: {self.major}")

# Get input from the user
student_name = input("Enter student name: ")
student_age = int(input("Enter student age: "))
student_grade = input("Enter student grade: ")
student_major = input("Enter student major: ")

# Create a Student object
student1 = Student(student_name, student_age, student_grade, student_major)

# Display student information
student1.display_info()
```

OUTPUT:

```
Enter student name: manoj
Enter student age: 18
Enter student grade: 10
Enter student major: y
Student Name: manoj
Student Age: 18
Student Grade: 10
Student Major: y
```

OBSERVATION:

I observed that the code is clean and in a structured way, as shown in the above screenshot. As per my experience, it took a lot of prompts before getting this output. However, after providing an appropriate prompt by researching, I obtained the desired output.

TASK DESCRIPTION-2:

PROMPT AI TO COMPLETE A FUNCTION THAT PRINTS THE FIRST 10 MULTIPLES OF A NUMBER USING A LOOP.

- ANALYZE THE GENERATED CODE
- ASK AI TO GENERATE CODE USING OTHER CONTROLLED LOOPING

PROMPT: Write a Python code that prints the first 10 multiples of a given number by the user using a loop.

CODE:

```
# Get input from the user
number = int(input("Enter a number: "))

# Print the first 10 multiples
print(f"The first 10 multiples of {number} are:")
for i in range(1, 11):
    multiple = number * i
    print(multiple)
```

OUTPUT:

```
Enter a number: 3
The first 10 multiples of 3 are:
3
6
9
12
15
18
21
24
27
30
```

OBSERVATION:

THERE IS NOTHING TO CORRECT. THE CODE IS EXECUTED PERFECTLY BY USING THE FOR AND GENERATED THE MINIMAL CODE WHICH MEETS MY REQUIREMENT AND SIMPLE OUTPUT.

TASK DESCRIPTION 3:

Ask AI to write nested if-elif-else conditionals to classify age groups.

- Analyze the generated code
- Ask AI to generate code using other conditional statements

PROMPT:

Write a python code using nested if-elif-else conditionals to classify age groups

CODE WITH OUTPUT:

```
# Get age input from the user
age = int(input("Enter your age: "))

# Classify age groups using nested if-elif-else
if age < 0:
    print("Invalid age.")
elif age <= 12:
    print("You are a child.")
elif age <= 19:
    print("You are a teenager.")
else:
    # Nested conditionals for adult age groups
    if age <= 64:
        print("You are an adult.")
    else:
        print("You are a senior citizen.")
```

```
Enter your age: 200
You are a senior citizen.
```

UPDATED CODE USING OTHER CONDITIONAL STATEMENT

PROMPT:

UPDATE CODE USING OTHER CONDITIONAL STATEMENTS EXCEPT NESTED IF-ELIF-ELSE

CODE WITH OUTPUT:

```
# Get age input from the user
age = int(input("Enter your age: "))

# Classify age groups using different conditional statements
if age < 0:
    print("Invalid age.")
elif 0 <= age <= 12:
    print("You are a child.")
elif 13 <= age <= 19:
    print("You are a teenager.")
elif 20 <= age <= 64:
    print("You are an adult.")
else:
    print("You are a senior citizen.")
```

```
Enter your age: 4
You are a child.
```

OBSERVATION:

I observed that the code is clean and in a structured way, as shown in the above screenshot. As per my experience, it took a lot of prompts before getting this output. However, after providing an appropriate prompt by researching, I obtained the desired output.And I updated the code without using nested if-elif-else, and it worked properly

TASK DESCRIPTION:4

GENERATE A SUM_TO_N() FUNCTION TO CALCULATE THE SUM OF THE FIRST N NUMBERS

- ANALYZE THE GENERATED CODE
- GET SUGGESTIONS FROM AI WITH OTHER CONTROLLED LOOPING

PROMPT:

GENERATE A PYTHON CODE THAT CALCULATES THE SUM OF THE FIRST N NUMBERS USING THE SUM_TO_(N) FUNCTION.

GENERATED CODE:

```python
def sum_to_n(n):
  """Calculates the sum of the first n positive integers."""
  total_sum = 0
  for i in range(1, n + 1):
    total_sum += i
  return total_sum

# Get input from the user
num = int(input("Enter a positive integer (N): "))

# Calculate and print the sum
result = sum_to_n(num)
print(f"The sum of the first {num} numbers is: {result}")
```

OUTPUT:

```
Enter a positive integer (N): 3
The sum of the first 3 numbers is: 6
```

OBSERVATION:

I observed that the code is clean and in a structured way, as shown in the above screenshot. As per my experience, it took a lot of prompts before getting this output. However, after providing an appropriate prompt by researching, I obtained the desired output.

EXPLANATION:

This code calculates the sum of the first N positive integers, where N is provided by the user.

- It defines a function sum_to_n(n) that takes an integer n as input.
- Inside the function, it initializes a variable total_sum to 0.
- It then uses a for loop to iterate through numbers from 1 up to n (inclusive).
- In each iteration, it adds the current number (i) to total_sum.

- After the loop finishes, the function returns the calculated total_sum.
- The code then prompts the user to enter a positive integer (N).
- It calls the sum_to_n() function with the user's input and stores the result in the result variable.
- Finally, it prints the sum of the first N numbers.

TASK DESCRIPTION 5:

• USE AI TO BUILD A BANK ACCOUNT CLASS WITH DEPOSIT, WITHDRAW, AND BALANCE METHODS.

• ANALYZE THE GENERATED CODE

• ADD COMMENTS AND EXPLAIN CODE

PROMPT:

GENERATE A PYTHON CODE TO BUILD A BANK ACCOUNT CLASS WITH DEPOSIT, withdrawal, AND BALANCE METHODS. and all the entries should be user defined

GENERATED CODE:-

```python
    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposit of ${amount} successful. New balance: ${self.balance}")
        else:
            print("Invalid deposit amount. Amount must be positive.")

    def withdrawal(self, amount):
        if amount > 0:
            if self.balance >= amount:
                self.balance -= amount
                print(f"Withdrawal of ${amount} successful. New balance: ${self.balance}")
            else:
                print("Insufficient funds.")
        else:
            print("Invalid withdrawal amount. Amount must be positive.")

    def get_balance(self):
        return self.balance

# Get user input for creating a bank account
account_num = input("Enter account number: ")
initial_bal = float(input("Enter initial balance: "))
account1 = BankAccount(account_num, initial_bal)

# Display initial balance
print(f"Initial balance: ${account1.get_balance()}")

# Get user input for deposit
deposit_amount = float(input("Enter deposit amount: "))
account1.deposit(deposit_amount)

# Get user input for withdrawal
withdrawal_amount = float(input("Enter withdrawal amount: "))
account1.withdrawal(withdrawal_amount)

# Check final balance
print(f"Current balance: ${account1.get_balance()}")
```

OUTPUT:

```
Enter account number: 12334
Enter initial balance: 34567
Initial balance: $34567.0
Enter deposit amount: 34567
Deposit of $34567.0 successful. New balance: $69134.0
Enter withdrawal amount: 346
Withdrawal of $346.0 successful. New balance: $68788.0
Current balance: $68788.0
```

OBSERVATION:

This prompt, which provided code, is faster than the before tasks and worked perfectly

THANKYOU