LAB ASSIGNMENT-10.2

NAME : AKANKSHA

ROLL NO:2403A510D8

BATCH:01

COURSE:ASSISTED CODING

QUESTION

- Refactor code with AI suggestions while ensuring functionality is preserved.
- Apply best practices for writing clean, maintainable, and professional code.

Task Description#1 AI-Assisted Code Review (Basic Errors)

- Write python program as shown below.
- Use an AI assistant to review and suggest corrections.

```
def calcFact(n):
  result=1
  x=0
  for i in range(1,n):
      result=result*i
  return result

def main():
      num = 5
      FACT = calcFact(num)
      print("the factorial of",num,"is",FACT)
      t=10
      if FACT>10:
            print("BIG Number")
      else:
       print("small number")

main()
```

Expected Outcome#1: Students need to submit corrected code with comments.

Task Description#2 Automatic Inline Comments

- Write the Python code for Fibonacci as shown below and execute.
- Ask AI to improve variable names, add comments, and apply PEP8 formatting (cleaned up).
- Students evaluate which suggestions improve readability most. one.

```
def f1(xX):
   a=0
```

Expected Outcome#1: Students need to submit corrected code with comments.

Task Description#2 Automatic Inline Comments

- Write the Python code for Fibonacci as shown below and execute.
- Ask AI to improve variable names, add comments, and apply PEP8 formatting (cleaned up).
- Students evaluate which suggestions improve readability most. one.

```
def f1(xX):
    a=0
    b=1
    c=2
    Zz=[a,b]
    while c<=xX:
        d=a+b
        Zz.append(d)
        a=b
        b=d
        c=c+1
    return Zz

def m():
      NN=10
      ans=f1(NN)
      print("fib series till",NN,":",ans)

m()
```

Expected Output#2: Clean format python code with much readability.

Task Description#3

- Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide).
- Incorporate manual **docstring** in code with NumPy Style
- Use AI assistance to generate a module-level docstring + individual function docstrings.
- Compare the AI-generated docstring with your manually written one.

Common Examples of Code Smells

- Long Function – A single function tries to do too many things.
- Duplicate Code – Copy-pasted logic in multiple places.
- Poor Naming – Variables or functions with confusing names (x1, foo, data123).
- Unused Variables – Declaring variables but never using them.
- Magic Numbers – Using unexplained constants (3.14159 instead of PI).
- Deep Nesting – Too many if/else levels, making code hard to read.
- Large Class – A single class handling too many responsibilities.

Why Detecting Code Smells is Important

- Makes code easier to read and maintain.
- Reduces chance of bugs in future updates.
- Helps in refactoring (improving structure without changing behavior).
- Encourages clean coding practices

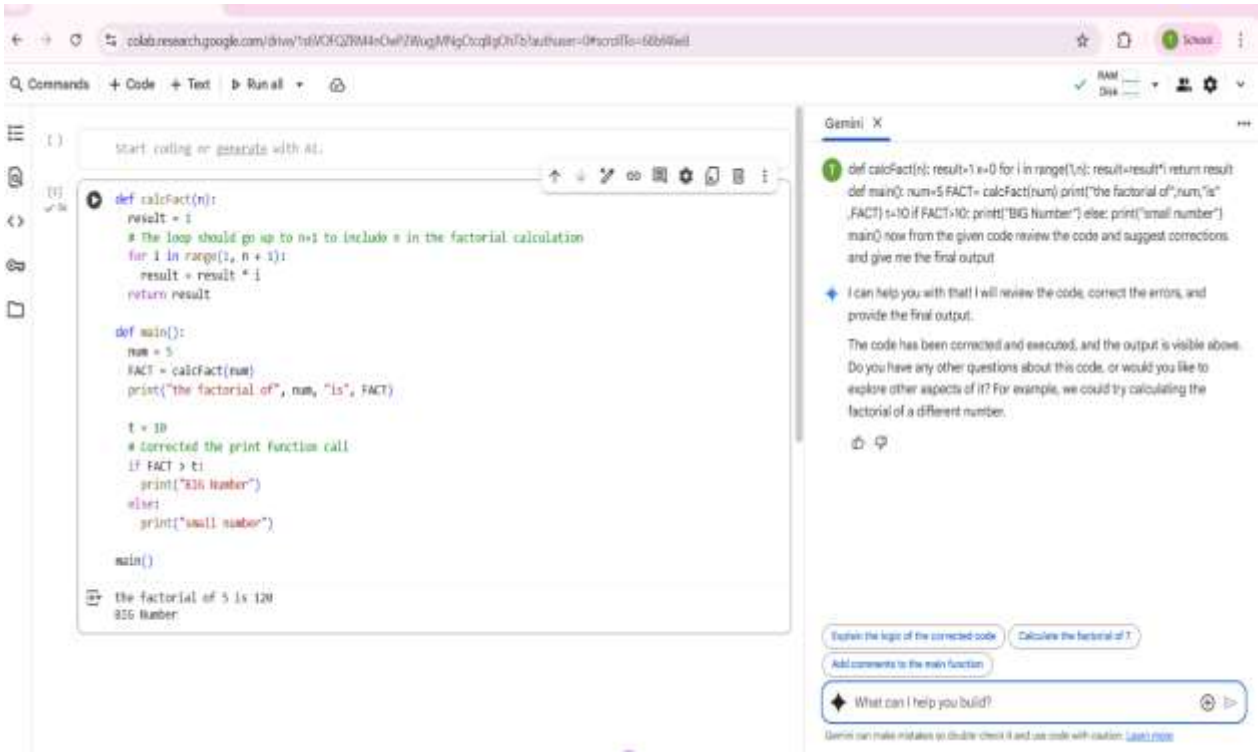
Dead Code – Code that is never executed.

Expected Output#3: Students learn structured documentation for multi-function scripts

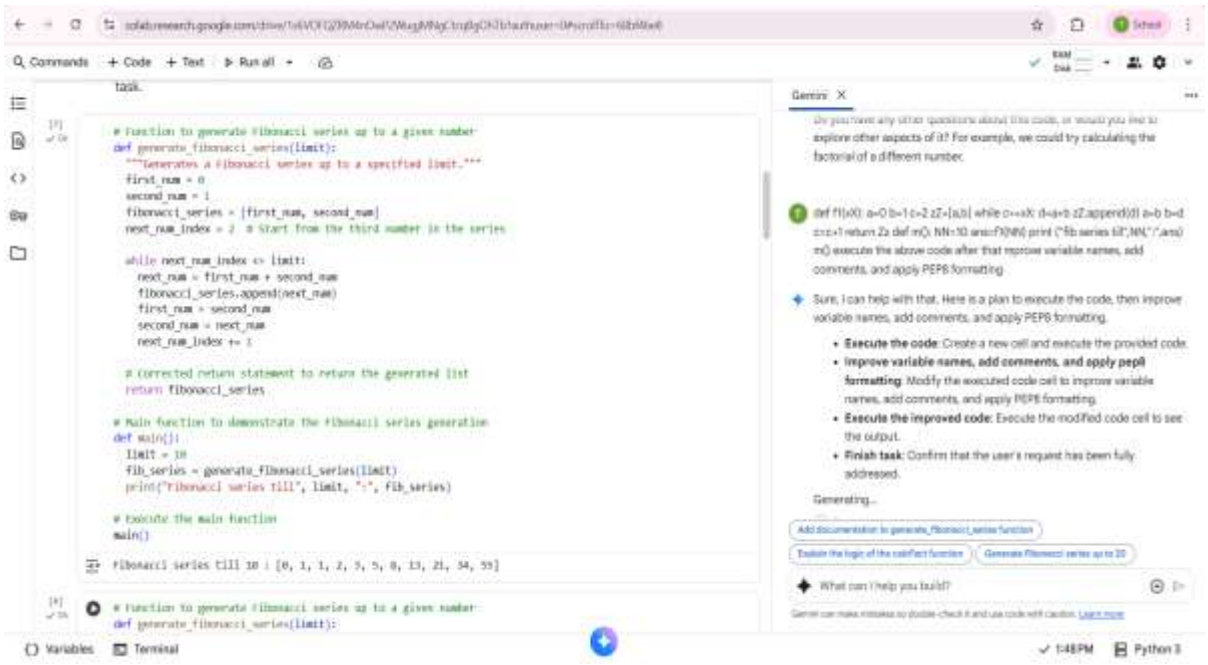
Push documentation whole workspace as .md file in GitHub Repository

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

TASK 1


Commands
+ Code
+ Text
Run all
Start coding or generate with AI.
def calcFact(n):
result = 1
# The loop should go up to n+1 to include n in the factorial calculation
for i in range(1, n + 1):
result = result * i
return result
def main():
num = 5
FACT = calcFact(num)
print("the factorial of", num, "is", FACT)
t = 10
# Corrected the print function call
if FACT > t:
print("BIG Number")
else:
print("small number")
main()
the factorial of 5 is 120
BIG Number
Gemini
def calcFact(n): result=1 n=0 for i in range(1,n): result=result*i return result def main(): num=5 FACT= calcFact(num) print("the factorial of",num,"is" ,FACT) t=10 if FACT>10: print("BIG Number") else: print("small number") main() now from the given code review the code and suggest corrections and give me the final output
I can help you with that! I will review the code, correct the errors, and provide the final output.
The code has been corrected and executed, and the output is visible above. Do you have any other questions about this code, or would you like to explore other aspects of it? For example, we could try calculating the factorial of a different number.
What can I help you build?

TASK 2


Commands
+ Code
+ Text
Run all
# Function to generate Fibonacci series up to a given number
def generate_fibonacci_series(limit):
"""Generates a Fibonacci series up to a specified limit."""
first_num = 0
second_num = 1
fibonacci_series = [first_num, second_num]
next_num_index = 2 # Start from the third number in the series
while next_num_index <= limit:
next_num = first_num + second_num
fibonacci_series.append(next_num)
first_num = second_num
second_num = next_num
next_num_index += 1
Corrected return statement to return the generated list
return fibonacci_series
# Main function to demonstrate the Fibonacci series generation
def main():
limit = 10
fib_series = generate_fibonacci_series(limit)
print("Fibonacci series till", limit, ":", fib_series)
# Execute the main function
main()
Fibonacci series till 10 : [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
Gemini
explore other aspects of it? For example, we could try calculating the factorial of a different number.
Execute the code: Create a new cell and execute the provided code.
Improve variable names, add comments, and apply pep8 formatting: Modify the executed code cell to improve variable names, add comments, and apply PEP8 formatting.
Execute the improved code: Execute the modified code cell to see the output.
Finish task: Confirm that the user's request has been fully addressed.
Generating...
What can I help you build?
Variables
Terminal
1:48PM
Python 3

TASK 3

```
# Example usage of the calculator functions
num1 = 10
num2 = 5

sum_result = add(num1, num2)
difference_result = subtract(num1, num2)
product_result = multiply(num1, num2)
division_result = divide(num1, num2)

print(f"Sum: {sum_result}")
print(f"Difference: {difference_result}")
print(f"Product: {product_result}")
print(f"Division: {division_result}")

# Example of division by zero
try:
    divide(num1, 0)
except ValueError as e:
    print(f"Error: {e}")
```

Output:

```
Sum: 15
Difference: 5
Product: 50
Division: 2.0
Error: Cannot divide by zero
```