# AI Assisted Coding

## Lab Assignment 9.5

Name : V Sai Krishnachaithanya

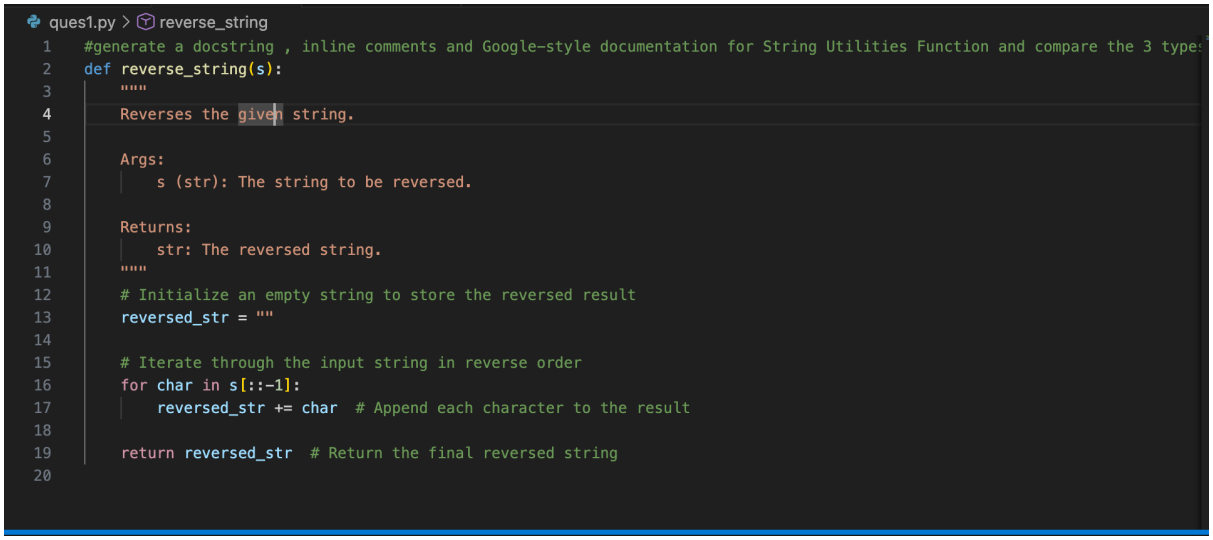
Hall Ticket no : 2303A52154

### Problem 1: String Utilities Function

Consider the following Python function:
def reverse_string(text):
return text[::-1]
Task:
1. Write documentation in:
o (a) Docstring
o (b) Inline comments
o (c) Google-style documentation
2. Compare the three documentation styles.
3. Recommend the most suitable style for a utility-based string library.

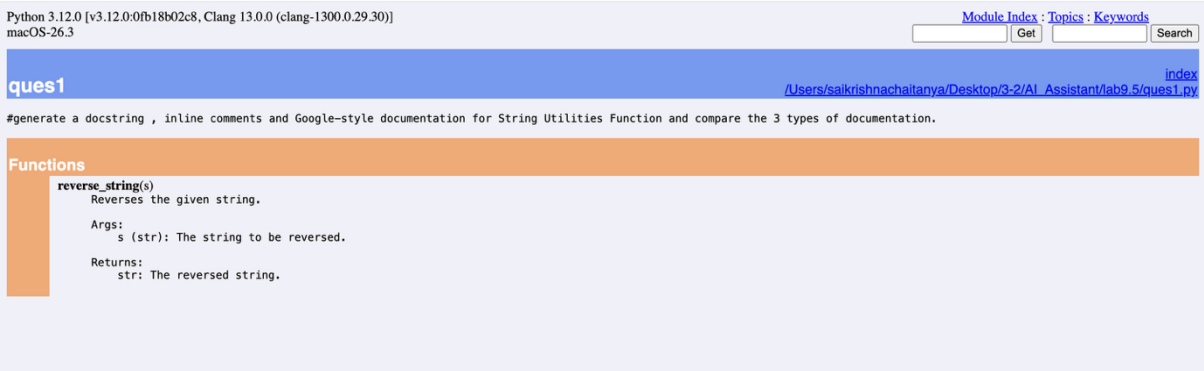
### Prompt:

generate a docstring , inline comments and Google-style documentation for String Utilities Function and compare the 3 types of documentation.



# Output:

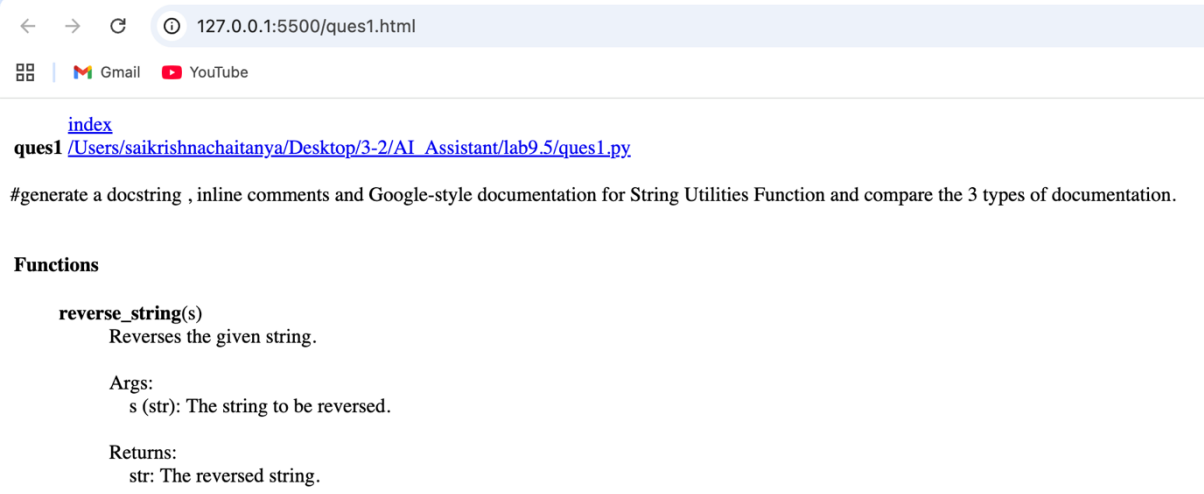
# python3 -m pydoc -p 1234




# python3 -m pydoc ques1

```
Help on module ques1:

NAME
    ques1 - #generate a docstring , inline comments and Google-style documentation for String Utilities Function and compare th
e 3 types of documentation.

FUNCTIONS
    reverse_string(s)
        Reverses the given string.

        Args:
            s (str): The string to be reversed.

        Returns:
            str: The reversed string.

FILE
    /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques1.py

~
```

# python3 -m pydoc -w ques1



## Problem 2: Password Strength Checker

Consider the function:

def check_strength(password):

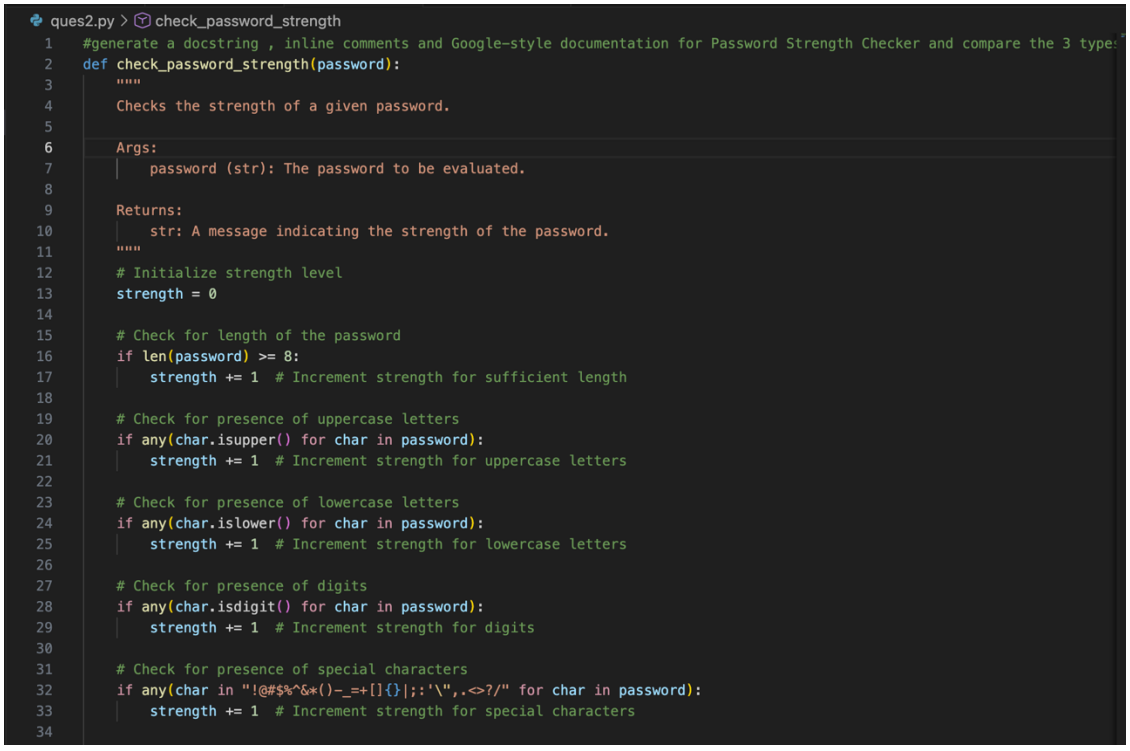
return len(password) >= 8

Task:

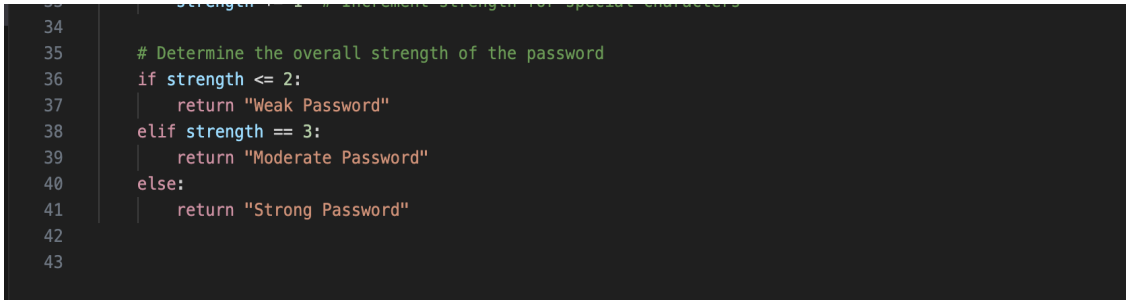
1. Document the function using docstring, inline comments, and Google style.
2. Compare documentation styles for security-related code.
3. Recommend the most appropriate style.

## Prompt:

generate a docstring , inline comments and Google-style documentation for Password Strength Checker and compare the 3 types of documentation.



```python
#generate a docstring , inline comments and Google-style documentation for Password Strength Checker and compare the 3 type:
def check_password_strength(password):
    """
    Checks the strength of a given password.

    Args:
        password (str): The password to be evaluated.

    Returns:
        str: A message indicating the strength of the password.
    """
    # Initialize strength level
    strength = 0

    # Check for length of the password
    if len(password) >= 8:
        strength += 1  # Increment strength for sufficient length

    # Check for presence of uppercase letters
    if any(char.isupper() for char in password):
        strength += 1  # Increment strength for uppercase letters

    # Check for presence of lowercase letters
    if any(char.islower() for char in password):
        strength += 1  # Increment strength for lowercase letters

    # Check for presence of digits
    if any(char.isdigit() for char in password):
        strength += 1  # Increment strength for digits

    # Check for presence of special characters
    if any(char in "!@#$%^&*()-_=+[]{}|;:'\",.<>?/" for char in password):
        strength += 1  # Increment strength for special characters
```



```python
    # Determine the overall strength of the password
    if strength <= 2:
        return "Weak Password"
    elif strength == 3:
        return "Moderate Password"
    else:
        return "Strong Password"
```

# Output:

# python3 -m pydoc ques2

```
Help on module ques2:

NAME
    ques2 - #generate a docstring , inline comments and Google-style documentation for Password Strength Checker and compare th
e 3 types of documentation.

FUNCTIONS
    check_password_strength(password)
        Checks the strength of a given password.

        Args:
            password (str): The password to be evaluated.

        Returns:
            str: A message indicating the strength of the password.

FILE
    /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques2.py

(END)
```

# python3 -m pydoc -w ques2

index
**ques2** /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques2.py

#generate a docstring , inline comments and Google-style documentation for Password Strength Checker and compare the 3 types of documentation.

**Functions**

**check_password_strength**(password)
Checks the strength of a given password.

Args:
password (str): The password to be evaluated.

Returns:
str: A message indicating the strength of the password.

# python3 -m pydoc -p 1234

Python 3.12.0 [v3.12.0:0fb18b02c8, Clang 13.0.0 (clang-1300.0.29.30)]
macOS-26.3

Module Index : Topics : Keywords
Get Search

**ques2** index /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques2.py

#generate a docstring , inline comments and Google-style documentation for Password Strength Checker and compare the 3 types of documentation.

**Functions**

**check_password_strength**(password)
Checks the strength of a given password.

Args:
password (str): The password to be evaluated.

Returns:
str: A message indicating the strength of the password.

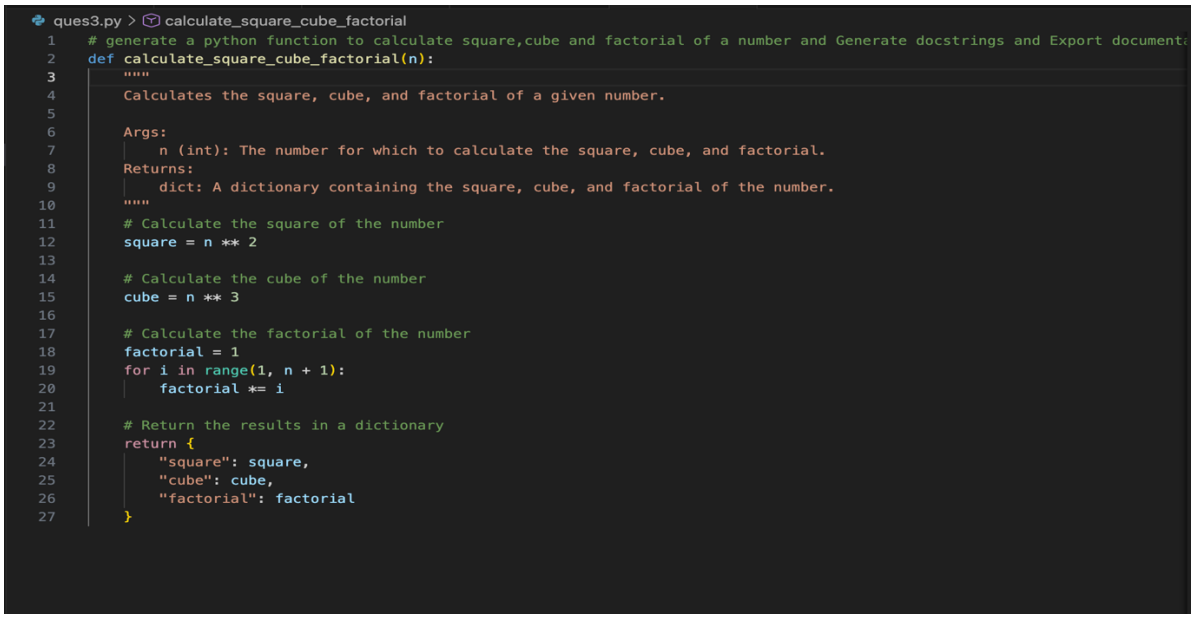
## Problem 3: Math Utilities Module

Task:

1. Create a module math_utils.py with functions:
o square(n)
o cube(n)
o factorial(n)
2. Generate docstrings automatically using AI tools.
3. Export documentation as an HTML file.

## Prompt :

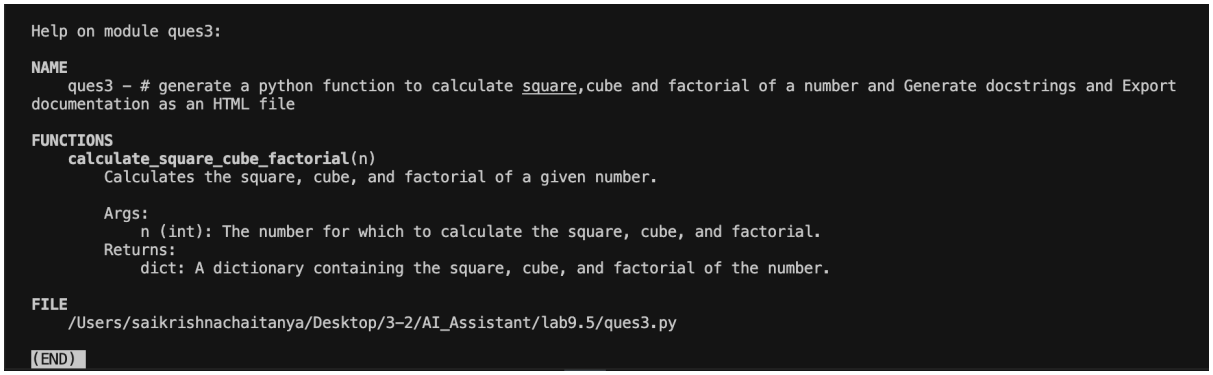
generate a python function to calculate square,cube and factorial of a number and Generate docstrings and Export documentation as an HTML file



```
ques3.py > calculate_square_cube_factorial
# generate a python function to calculate square,cube and factorial of a number and Generate docstrings and Export documenta
def calculate_square_cube_factorial(n):
    """
    Calculates the square, cube, and factorial of a given number.

    Args:
        n (int): The number for which to calculate the square, cube, and factorial.
    Returns:
        dict: A dictionary containing the square, cube, and factorial of the number.
    """
    # Calculate the square of the number
    square = n ** 2

    # Calculate the cube of the number
    cube = n ** 3

    # Calculate the factorial of the number
    factorial = 1
    for i in range(1, n + 1):
        factorial *= i

    # Return the results in a dictionary
    return {
        "square": square,
        "cube": cube,
        "factorial": factorial
    }
```

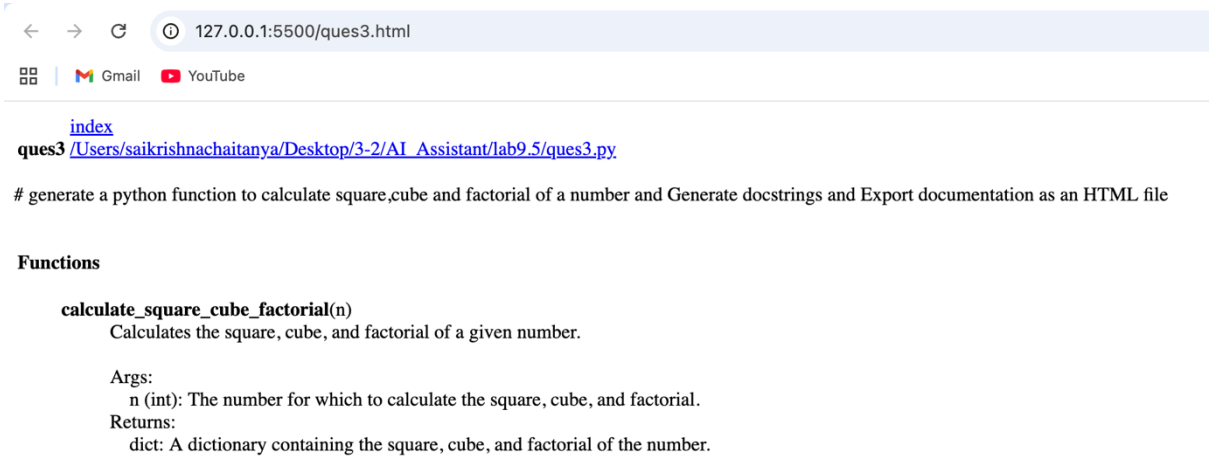
## Output:

# python3 -m pydoc ques3

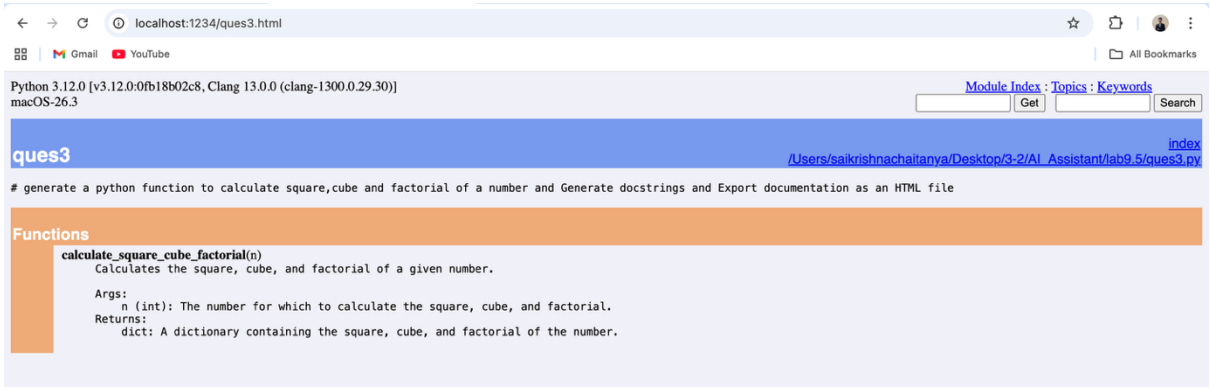


```
Help on module ques3:

NAME
    ques3 - # generate a python function to calculate square,cube and factorial of a number and Generate docstrings and Export
documentation as an HTML file

FUNCTIONS
    calculate_square_cube_factorial(n)
        Calculates the square, cube, and factorial of a given number.

        Args:
            n (int): The number for which to calculate the square, cube, and factorial.
        Returns:
            dict: A dictionary containing the square, cube, and factorial of the number.

FILE
    /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques3.py

(END)
```

# python3 -m pydoc -w ques3




# python3 -m pydoc -p 1234




# Problem 4: Attendance Management Module

Task:

1. Create a module attendance.py with functions:

mark_present(student)

mark_absent(student)

get_attendance(student)

2. Add proper docstrings.

3. Generate and view documentation in terminal and browse

# Prompt:

generate a python Attendance Management Module with functions to mark attendance, mark absent, get attendence and generate attendance reports. Include docstrings and inline comments for each function.

ques4.py > ...

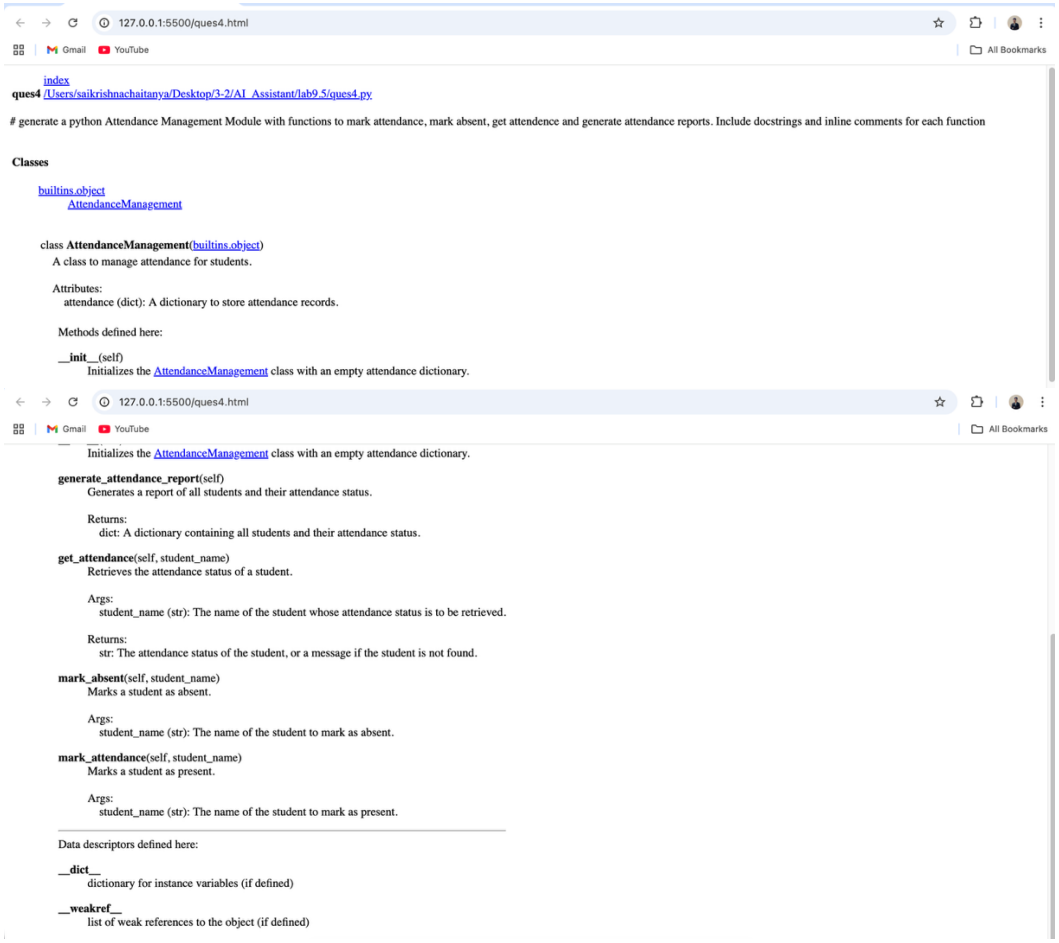
```python
# generate a python Attendance Management Module with functions to mark attendance, mark absent, get attendence and generate attendance re
class AttendanceManagement:
    """
    A class to manage attendance for students.

    Attributes:
        attendance (dict): A dictionary to store attendance records.
    """

    def __init__(self):
        """Initializes the AttendanceManagement class with an empty attendance dictionary."""
        self.attendance = {}

    def mark_attendance(self, student_name):
        """
        Marks a student as present.

        Args:
            student_name (str): The name of the student to mark as present.
        """
        self.attendance[student_name] = "Present"  # Mark the student as present


    def get_attendance(self, student_name):
        """
        Retrieves the attendance status of a student.

        Args:
            student_name (str): The name of the student whose attendance status is to be retrieved.

        Returns:
            str: The attendance status of the student, or a message if the student is not found.
        """
        return self.attendance.get(student_name, "Student not found")  # Return attendance status or not found message

    def generate_attendance_report(self):
        """
        Generates a report of all students and their attendance status.

        Returns:
            dict: A dictionary containing all students and their attendance status.
        """
        return self.attendance  # Return the entire attendance record
```

# Output:

# python3 -m pydoc ques4

```
Help on module ques4:

NAME
    ques4 - # generate a python Attendance Management Module with functions to mark attendance, mark absent, get attendence and generate atten
dance reports. Include docstrings and inline comments for each function

CLASSES
    builtins.object
        AttendanceManagement

    class AttendanceManagement(builtins.object)
     |  A class to manage attendance for students.
     |
     |  Attributes:
     |      attendance (dict): A dictionary to store attendance records.
     |
     |  Methods defined here:
     |
     |  __init__(self)
     |      Initializes the AttendanceManagement class with an empty attendance dictionary.
     |
     |  generate_attendance_report(self)
     |      Generates a report of all students and their attendance status.
     |
     |      Returns:
     |          dict: A dictionary containing all students and their attendance status.
     |
:
```

# python3 -m pydoc -w ques4




# python3 -m pydoc -p 1234

# Problem 5: File Handling Function

Consider the function:

def read_file(filename):

with open(filename, 'r') as f:

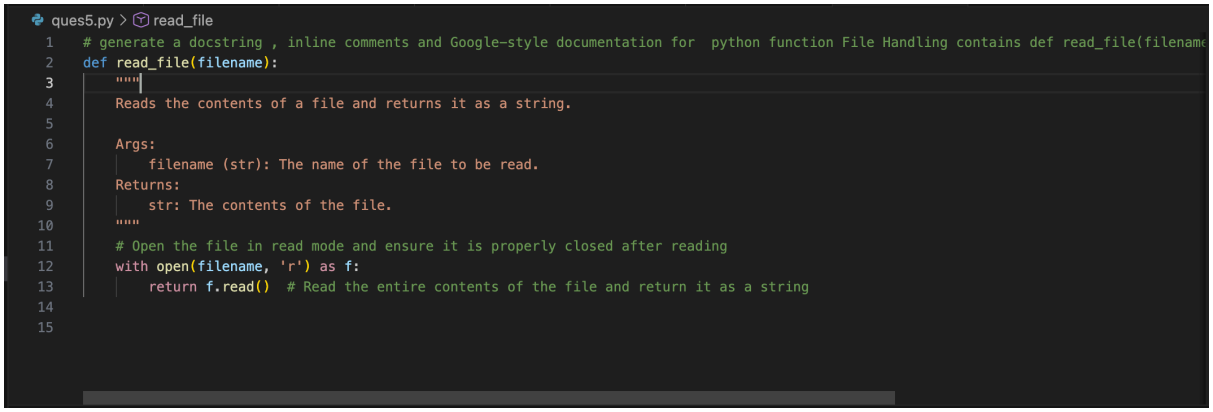
return f.read()

Task:

1. Write documentation using all three formats.
2. Identify which style best explains exception handling.
3. Justify your recommendation.

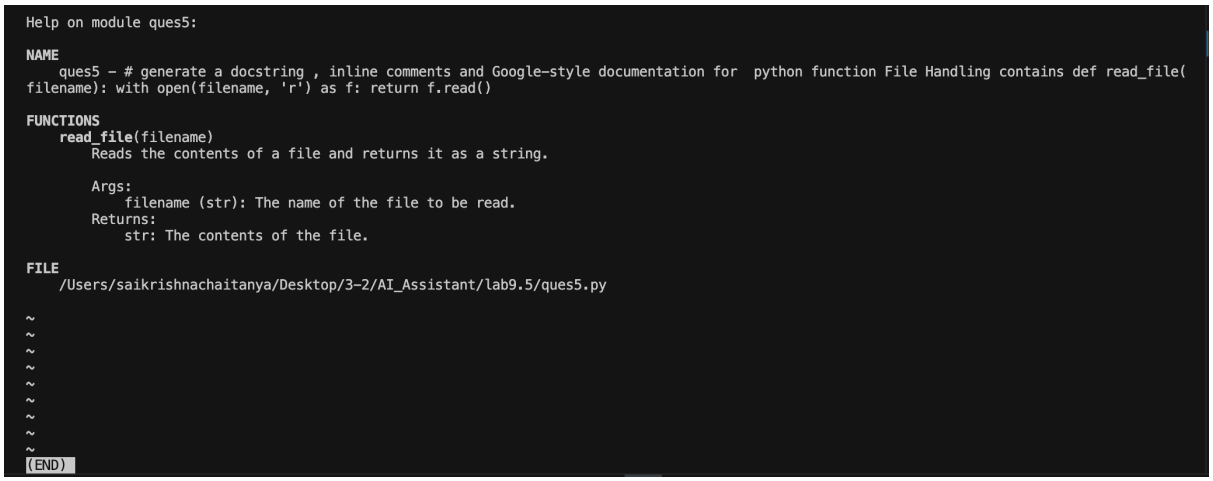
# Prompt:

generate a docstring , inline comments and Google-style documentation for  python function File Handling contains def read_file(filename): with open(filename, 'r') as f: return f.read()




# Output :

# python3 -m pydoc ques5



# # python3 -m pydoc -w ques5


# # python3 -m pydoc -p 1234