

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.

```
ques1.py > reverse_string  
1  #generate a docstring , inline comments and Google-style documentation for String Utilities Function and compare the 3 types:  
2  def reverse_string(s):  
3      """  
4          Reverses the given string.  
5        
6      Args:  
7          s (str): The string to be reversed.  
8        
9      Returns:  
10         str: The reversed string.  
11     """  
12     # Initialize an empty string to store the reversed result  
13     reversed_str = ""  
14       
15     # Iterate through the input string in reverse order  
16     for char in s[::-1]:  
17         reversed_str += char # Append each character to the result  
18       
19     return reversed_str # Return the final reversed string  
20
```

Output:

```
# python3 -m pydoc -p 1234
```

The screenshot shows a web browser displaying the pydoc documentation for the module `ques1`. The page title is `ques1`. At the top right, there are links for `Module Index`, `Topics`, and `Keywords`, along with a search bar. Below the title, a note says `#generate a docstring , inline comments and Google-style documentation for String Utilities Function and compare the 3 types of documentation.`. A section titled `Functions` contains a single entry: `reverse_string(s)`. The documentation for this function states: `Reverses the given string.` Under `Args:`, it says `s (str): The string to be reversed.` Under `Returns:`, it says `str: The reversed string.`

```
# python3 -m pydoc ques1
```

The screenshot shows a terminal window with the command `# python3 -m pydoc ques1` entered. The output is a detailed help document for the module `ques1`. It includes sections for `NAME`, `FUNCTIONS`, and `FILE`. The `FUNCTIONS` section contains the `reverse_string(s)` function, which is described as reversing the given string. The `FILE` section shows the file path as `/Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques1.py`.

```
# python3 -m pydoc -w ques1
```

The screenshot shows a web browser displaying the generated static HTML documentation for the module `ques1`. The address bar shows the URL `127.0.0.1:5500/ques1.html`. The page has a header with back, forward, and refresh buttons, and links for `Gmail` and `YouTube`. Below the header, there is a navigation bar with `index` and a link to the source file `ques1 /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques1.py`. The main content area contains the same documentation as the previous screenshots, including the `reverse_string(s)` function and its details.

Functions

`reverse_string(s)`
Reverses the given string.

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

Returns:
`str: The reversed string.`

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.

```
ques2.py > check_password_strength
1  #generate a docstring , inline comments and Google-style documentation for Password Strength Checker and compare the 3 types
2  def check_password_strength(password):
3      """
4          Checks the strength of a given password.
5
6      Args:
7          password (str): The password to be evaluated.
8
9      Returns:
10         str: A message indicating the strength of the password.
11     """
12     # Initialize strength level
13     strength = 0
14
15     # Check for length of the password
16     if len(password) >= 8:
17         strength += 1 # Increment strength for sufficient length
18
19     # Check for presence of uppercase letters
20     if any(char.isupper() for char in password):
21         strength += 1 # Increment strength for uppercase letters
22
23     # Check for presence of lowercase letters
24     if any(char.islower() for char in password):
25         strength += 1 # Increment strength for lowercase letters
26
27     # Check for presence of digits
28     if any(char.isdigit() for char in password):
29         strength += 1 # Increment strength for digits
30
31     # Check for presence of special characters
32     if any(char in "!@#$%^&*()-_+=[]{}|;:'\".,>?/" for char in password):
33         strength += 1 # Increment strength for special characters
34
35         strength += 1 # Increment strength for special characters
36
37     # Determine the overall strength of the password
38     if strength <= 2:
39         return "Weak Password"
40     elif strength == 3:
41         return "Moderate Password"
42     else:
43         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

(FIND)
```

```
# 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
1  # generate a python function to calculate square,cube and factorial of a number and Generate docstrings and Export documentation as an HTML file
2  def calculate_square_cube_factorial(n):
3      """
4          Calculates the square, cube, and factorial of a given number.
5
6      Args:
7          n (int): The number for which to calculate the square, cube, and factorial.
8      Returns:
9          dict: A dictionary containing the square, cube, and factorial of the number.
10         """
11        # Calculate the square of the number
12        square = n ** 2
13
14        # Calculate the cube of the number
15        cube = n ** 3
16
17        # Calculate the factorial of the number
18        factorial = 1
19        for i in range(1, n + 1):
20            factorial *= i
21
22        # Return the results in a dictionary
23        return {
24            "square": square,
25            "cube": cube,
26            "factorial": factorial
27        }
```

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
```



Functions

calculate_square_cube_factorial(n)

Calculates the square, cube, and factorial of a given number.

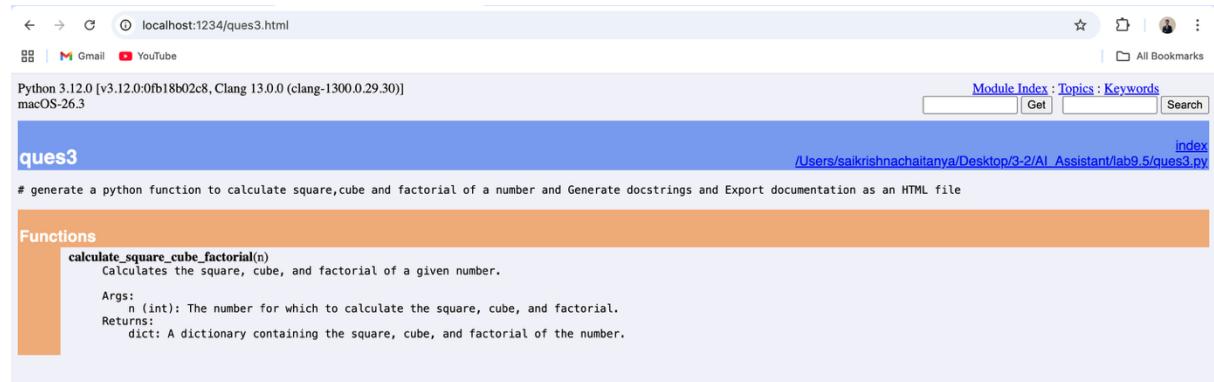
Args:

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.

```
# 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 attendance and generate attendance reports. Include docstrings and inline comments for each function.

```
ques4.py > ...
1  # generate a python Attendance Management Module with functions to mark attendance, mark absent, get attendance and generate attendance re
2  class AttendanceManagement:
3      """
4          A class to manage attendance for students.
5
6          Attributes:
7              attendance (dict): A dictionary to store attendance records.
8          """
9
10     def __init__(self):
11         """Initializes the AttendanceManagement class with an empty attendance dictionary."""
12         self.attendance = {}
13
14     def mark_attendance(self, student_name):
15         """
16             Marks a student as present.
17
18             Args:
19                 student_name (str): The name of the student to mark as present.
20             """
21         self.attendance[student_name] = "Present" # Mark the student as present
22
23
24     def get_attendance(self, student_name):
25         """
26             Retrieves the attendance status of a student.
27
28             Args:
29                 student_name (str): The name of the student whose attendance status is to be retrieved.
30
31             Returns:
32                 str: The attendance status of the student, or a message if the student is not found.
33             """
34         return self.attendance.get(student_name, "Student not found") # Return attendance status or not found message
35
36     def generate_attendance_report(self):
37         """
38             Generates a report of all students and their attendance status.
39
40             Returns:
41                 dict: A dictionary containing all students and their attendance status.
42             """
43         return self.attendance # Return the entire attendance record
44
45
46
47
48
49
50
51
```

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 attendance and generate attendance 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
```

index
ques4 /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques4.py

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

Classes

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.

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.

mark_absent(self, student_name)

 Marks a student as absent.

 Args:

 student_name (str): The name of the student to mark as absent.

mark_attendance(self, student_name)

 Marks a student as present.

 Args:

 student_name (str): The name of the student to mark as present.

Data descriptors defined here:

__dict__

 dictionary for instance variables (if defined)

__weakref__

 list of weak references to the object (if defined)

```
# python3 -m pydoc -p 1234
```

localhost:1234/ques4.html

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

Module Index : Topics : Keywords

index
ques4 /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques4.py

generate a python Attendance Management Module with functions to mark attendance, mark absent, get attendance and generate attendance reports. Include docstrings and inline commen

Classes

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.

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:

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()

```
ques5.py >  read_file
1  # generate a docstring , inline comments and Google-style documentation for python function File Handling contains def read_file(filename)
2  def read_file(filename):
3      """
4          Reads the contents of a file and returns it as a string.
5
6          Args:
7              filename (str): The name of the file to be read.
8          Returns:
9              str: The contents of the file.
10         """
11     # Open the file in read mode and ensure it is properly closed after reading
12     with open(filename, 'r') as f:
13         return f.read() # Read the entire contents of the file and return it as a string
14
15
```

Output :

```
# python3 -m pydoc ques5
```

```
# python3 -m pydoc -w ques5
```

The screenshot shows a web browser window with the URL `127.0.0.1:5500/ques5.html`. The page title is `ques5`. The content includes a `read_file` function definition and its documentation. The browser's address bar and some navigation icons are visible at the top.

```
index
ques5 /Users/saikrishnachaitanya/Desktop/3-2/AI_Assistant/lab9.5/ques5.py

# 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()

Functions

read_file(filename)
    Reads the contents of a file and returns it as a string.

    Args:
        filename (str): The name of the file to be read.
    Returns:
        str: The contents of the file.
```

```
# python3 -m pydoc -p 1234
```

The screenshot shows a web browser window with the URL `localhost:1234/ques5.html`. The page title is `ques5`. The content includes a `read_file` function definition and its documentation. The browser's address bar and some navigation icons are visible at the top. A header bar at the top right contains links for `Module Index`, `Topics`, and `Keywords`.

```
Python 3.12.0 [v3.12.0:0fb18b02c8, Clang 13.0.0 (clang-1300.0.29.30)]
macOS-26.3
index
/Users/saikrishnachaitanya/Desktop/3-2/AI\_Assistant/lab9.5/ques5.py

# 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()

Functions

read_file(filename)
    Reads the contents of a file and returns it as a string.

    Args:
        filename (str): The name of the file to be read.
    Returns:
        str: The contents of the file.
```