

Assignment-9.5

2303A51666

Sony Dodla

Batch-23

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.

```
reverse_string.py > ...  
1 #Docstring  
2 def reverse_string(text):  
3     """  
4     This function takes a string as input and returns the reverse of that string.  
5     """  
6     return text[::-1]  
7 print(reverse_string.__doc__)  
8  
9 #google style documentation  
10 def reverse_string(text):  
11     """  
12     Reverses the given string.  
13     Args:  
14     | text (str): Input string to be reversed.  
15     Returns:  
16     | str: Reversed string.  
17     """  
18     return text[::-1]  
19 print(reverse_string.__doc__)  
20  
21 #Inline comments  
22 def reverse_string(text):  
23     # Use slicing with step -1 to reverse the string  
24     return text[::-1]  
25 print(reverse_string.__doc__) # This will print None since there is no docstring defined for the function
```

```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/reverse_string.py"  
This function takes a string as input and returns the reverse of that string.
```

```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/reverse_string.py"  
None
```

```

PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/Desktop/AI ASSISTANT CODING/reverse_string.py"

Reverses the given string.
Args:
    text (str): Input string to be reversed.
Returns:
    str: 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.

```

check_strength.py > ...
1  #Docstring
2  def check_strength(password):
3      """
4      Checks if password length is at least 8 characters.
5      """
6      return len(password) >= 8
7  print(check_strength.__doc__)
8
9
10 #Inline comments
11 def check_strength(password):
12     # Password must be at least 8 characters long
13     return len(password) >= 8
14 print(check_strength.__doc__) # This will print None since there is no docstring defined for the function
15
16
17 #Google style documentation
18 def check_strength(password):
19     """
20     Checks whether the password satisfies minimum length requirement.
21     Args:
22         password (str): Password string.
23     Returns:
24         bool: True if password length >= 8, else False.
25     """
26     return len(password) >= 8
27 print(check_strength.__doc__)

```

```

PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" op/AI ASSISTANT CODING/check_strength.py"

Checks if password length is at least 8 characters.
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" op/AI ASSISTANT CODING/check_strength.py"
None
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" op/AI ASSISTANT CODING/check_strength.py"

Checks whether the password satisfies minimum length requirement.
Args:
    password (str): Password string.
Returns:
    bool: True if password length >= 8, else False.

```

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.

```

math_utils.py > ...
1  def square(n):
2      """
3      Returns the square of a number.
4      demonstrates how to use docstrings in Python.
5      Parameters:
6      n (int): The number to be squared.
7      Returns:int: The square of n.
8      """
9      return n * n
10
11  def cube(n):
12      """
13      Returns the cube of a number.
14      demonstrates how to use docstrings in Python.
15      Parameters:
16      n (int): The number to be cubed.
17      Returns:int: The cube of n.
18      """
19      return n * n * n
20
21  def factorial(n):
22      """
23      Returns the factorial of a number.
24      demonstrates how to use docstrings in Python.
25      Parameters:
26      n (int): The number to calculate the factorial of.
27      Returns:int: The factorial of n.
28      """
29      if n == 0: # Check if n is 0 and return 1 if it is because factorial of 0 is 1
30          return 1 # Factorial of 0 is defined to be 1
31      else:
32          return n * factorial(n - 1) # Recursive call to calculate factorial of n
33  print(square.__doc__)
34  print(cube.__doc__)
35  print(factorial.__doc__)

```

```

PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python39-64/Scripts/python.exe" "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python39-64/Scripts/python.exe" op/AI ASSISTANT CODING/math_utils.py"

Returns the square of a number.
demonstrates how to use docstrings in Python.
Parameters:
n (int): The number to be squared.
Returns:int: The square of n.

Returns the cube of a number.
demonstrates how to use docstrings in Python.
Parameters:
n (int): The number to be cubed.
Returns:int: The cube of n.

Returns the factorial of a number.
demonstrates how to use docstrings in Python.
Parameters:
n (int): The number to calculate the factorial of.
Returns:int: The factorial of n.

```

Problem 4: Attendance Management Module

Task:

1. Create a module attendance.py with functions:

- o mark_present(student)
- o mark_absent(student)
- o get_attendance(student)

2. Add proper docstrings.

3. Generate and view documentation in terminal and browse

```
attendance.py > mark_absent
1  attendance = {}
2  def mark_present(student):
3      """
4      Marks a student as present in the attendance record.
5      Parameters:
6      student (str): The name of the student to be marked as present.
7      """
8      attendance[student] = "Present"
9
10 def mark_absent(student):
11     """
12     Marks a student as absent in the attendance record.
13     Parameters:
14     student (str): The name of the student to be marked as absent.
15     """
16     attendance[student] = "Absent"
17
18 def get_attendance(student):
19     """
20     Returns the attendance status of a student.
21     Parameters:
22     student (str): The name of the student whose attendance is to be retrieved.
23     Returns:
24     str: The attendance status of the student.
25     """
26     return attendance.get(student, "Not found")
27 import attendance
28 help(attendance)
```

```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> python -m pydoc -w attendance
Help on module attendance:

NAME
    attendance

FUNCTIONS
    get_attendance(student)
        Returns the attendance status of a student.
        Parameters:
        student (str): The name of the student whose attendance is to be retrieved.
        Returns:
        str: The attendance status of the student.

    mark_absent(student)
        Marks a student as absent in the attendance record.
        Parameters:
-- More --
```

Functions

```
def get_attendance(student):
    """Returns the attendance status of a student.

    Parameters:
        student (str): The name of the student whose attendance is to be retrieved.
    Returns:
        str: The attendance status of the student.
    """
    # ... (implementation) ...

def mark_absent(student):
    """Marks a student as absent in the attendance record.

    Parameters:
        student (str): The name of the student to be marked as absent.
    """
    # ... (implementation) ...

def mark_present(student):
    """Marks a student as present in the attendance record.

    Parameters:
        student (str): The name of the student to be marked as present.
    """
    # ... (implementation) ...
```

Data

```
attendance = {}
```

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.

read_file.py > ...

```
1  # DocString style:
2  def read_file(course):
3      """
4      Reads the content of a file and returns it as a string.
5      Parameters:
6      filename (str): The name of the file to be read.
7      Returns:
8      str: The content of the file.
9      Raises:
10     FileNotFoundError: If the specified file does not exist.
11     IOError: If an I/O error occurs while reading the file.
12     """
13
14     try:
15         with open(course, 'r') as f:
16             return f.read()
17     except FileNotFoundError:
18         print(f"Error: The file '{course}' was not found.")
19         raise
20     except IOError as e:
21         print(f"An I/O error occurred: {e}")
22         raise
23
24 #google style:
25 def read_file(filename):
26     """
27     Reads the content of a file and returns it as a string.
28
29     Args:
30     filename (str): The name of the file to be read.
31
32     Returns:
33     str: The content of the file.
34
35     Raises:
36     FileNotFoundError: If the specified file does not exist.
37     IOError: If an I/O error occurs while reading the file.
```

```
39     try:
40         with open(filename, 'r') as f:
41             return f.read()
42     except FileNotFoundError:
43         print(f"Error: The file '{filename}' was not found.")
44         raise
45     except IOError as e:
46         print(f"An I/O error occurred: {e}")
47         raise
48
49
50 #Inline comments
51 def read_file(filename):
52     # Open the file in read mode
53     # If the file does not exist, Python raises FileNotFoundError
54     with open(filename, 'r') as f:
55         # Read the entire contents of the file
56         return f.read()
```

```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> python -m pydoc -w course
{'name': 'Introduction to Computer Science', 'credits': 4}
Course with ID HIST101 not found in the catalog.

This function takes a course ID as input and simulates retrieving course information from a course catalog.
Args:course_id (str): The unique identifier for the course to be retrieved.
Returns:
dict: A dictionary containing the course information if found, or a message indicating that the course was not found.

None

This function takes a course ID as input and simulates retrieving course information from a course catalog.
Args:course_id (str): The unique identifier for the course to be retrieved.
Returns:
dict: A dictionary containing the course information if found, or a message indicating that the course was not found.

wrote course.html
```

```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> python -m pydoc -w course
{'name': 'Introduction to Computer Science', 'credits': 4}
Course with ID HIST101 not found in the catalog.

This function takes a course ID as input and simulates retrieving course information from a course catalog.
Args:course_id (str): The unique identifier for the course to be retrieved.
Returns:
dict: A dictionary containing the course information if found, or a message indicating that the course was not found.

None

This function takes a course ID as input and simulates retrieving course information from a course catalog.
Args:course_id (str): The unique identifier for the course to be retrieved.
Returns:
dict: A dictionary containing the course information if found, or a message indicating that the course was not found.

wrote course.html
```