AI Assisted Coding

Assignment-9.5
Ht.No : 2303A52350
Batch : 45

Codes:

```python
#Generate a python function to reverse a string and include doc
string documentation
def reverse_string(text):
    """
    Reverses the given string.

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

    Returns:
        str: The reversed string.
    """
    return text[::-1]
# Assert test cases
assert reverse_string("hello") == "olleh", "Test case 1 failed"
assert reverse_string("Python") == "nohtyP", "Test case 2 failed"
assert reverse_string("") == "", "Test case 3 failed"
assert reverse_string("A9_5") == "5_9A", "Test case 4 failed"
print("All test cases passed!")


#Generate a python function check_strength to the check the strength
of a password and include doc string documentation
def check_strength(password):
    """
    Checks the strength of a password.

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

    Returns:
        str: The strength of the password ("Weak", "Medium",
"Strong").
    """
    if len(password) < 6:
        return "Weak"
    elif len(password) < 12:
        return "Medium"
    else:
        return "Strong"
# Assert test cases
assert check_strength("12345")
assert check_strength("password")
assert check_strength("P@ssw0rd123")
assert check_strength("abc")
assert check_strength("longpassword")
```

```
print("All test cases passed!")
```

```
#Create a module math_utils.py  and include a function to calcuate
the square(n), q(n) and factorial(n) and doc string documentation.
def square(n):
    """
    Calculates the square of a number.

    Args:
        n (int): The number to be squared.

    Returns:
        int: The square of the number.
    """
    return n * n
def q(n):
    """Calculates the cube of a number.
    Args:
        n (int): The number to be cubed.

    Returns:
        int: The cube of the number.
    """
    return n * n * n
def factorial(n):
    """
    Calculates the factorial of a number.
    Args:
        n (int): The number to calculate factorial for.

    Returns:
        int: The factorial of the number.
    """
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)
# Assert test cases
assert square(5)
assert q(3)
assert factorial(5)
assert factorial(0)
print("All test cases passed!")
```

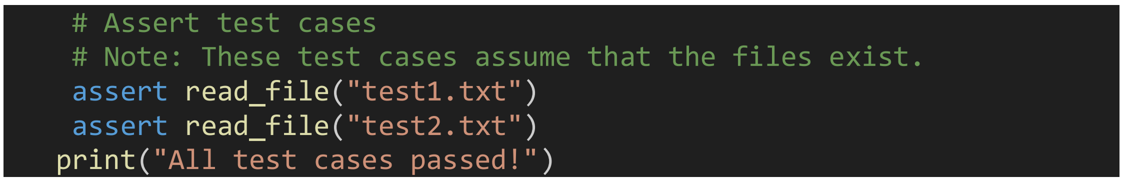
```
#create a module attendance.py and include functions
mark_present(student),
# mark_absent(student),get_attendace(student). Include docstring
documentation in the function.
class Attendance:
    def __init__(self):
        self.attendance_record = {}
    def mark_present(self, student):
        """
        Marks a student as present.
        Args:
            student (str): The name of the student to be marked
present.
        """
        self.attendance_record[student] = "Present"
    def mark_absent(self, student):
        """
        Marks a student as absent.
        Args:
            student (str): The name of the student to be marked
absent.
        """
        self.attendance_record[student] = "Absent"
    def get_attendance(self, student):
        """
        Gets the attendance status of a student.
```

```
        Args:
            student (str): The name of the student to check
attendance for.

        Returns:
            str: The attendance status of the student ("Present",
"Absent", or "Not Recorded").
        """
        return self.attendance_record.get(student, "Not Recorded")
# Assert test cases
attendance = Attendance()
attendance.mark_present("Alice")
attendance.mark_absent("Bob")
assert attendance.get_attendance("Alice")
assert attendance.get_attendance("Bob")
assert attendance.get_attendance("Charlie")
print("All test cases passed!")
```

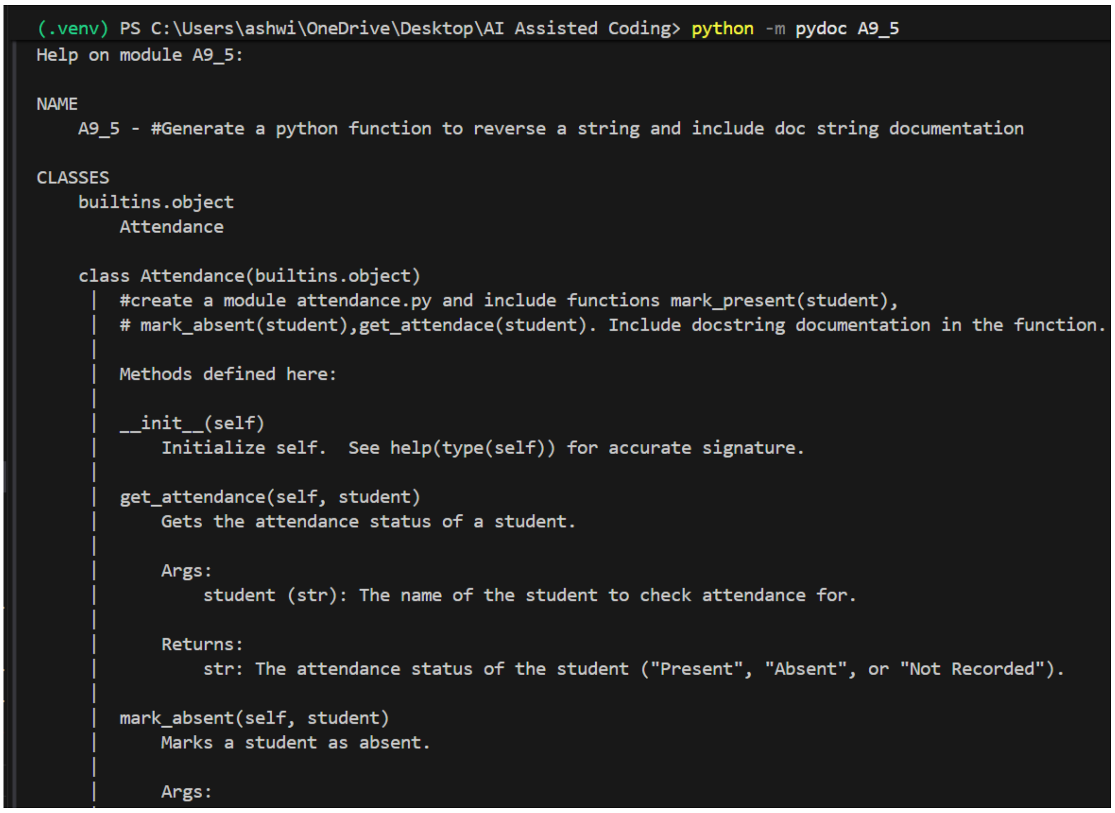
```
#Consider the function: def read_file(filename): with open(filename,
'r') as f: return f.read() and include docstring documentation in
the function.
def read_file(filename):
    """
    Reads the content of a file.
    Args:
        filename (str): The name of the file to be read.

    Returns:
        str: The content of the file.
    """
    with open(filename, 'r') as f:
        return f.read()
if __name__ == "__main__":
```



```python
    # Assert test cases
    # Note: These test cases assume that the files exist.
    assert read_file("test1.txt")
    assert read_file("test2.txt")
   print("All test cases passed!")
```

Output:

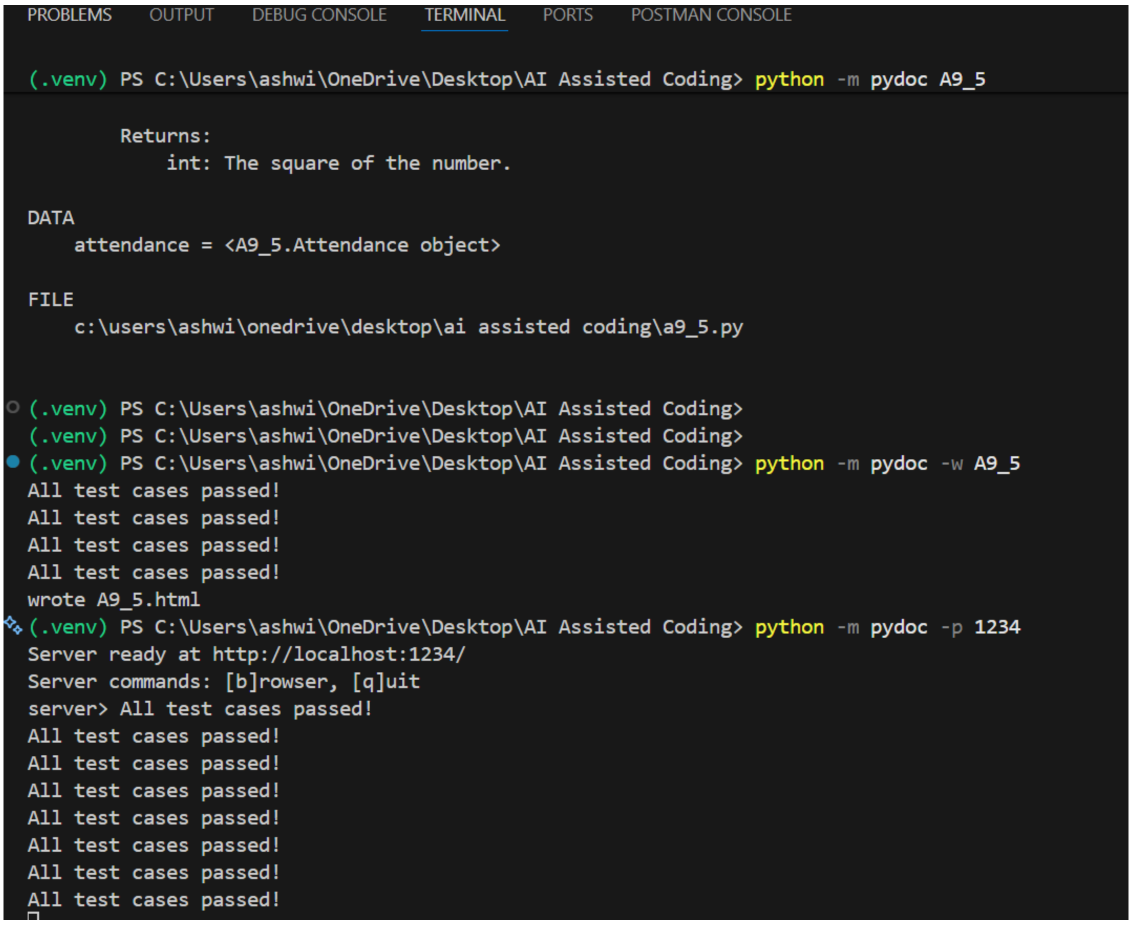


```
(.venv) PS C:\Users\ashwi\OneDrive\Desktop\AI Assisted Coding> python -m pydoc A9_5
Help on module A9_5:

NAME
    A9_5 - #Generate a python function to reverse a string and include doc string documentation

CLASSES
    builtins.object
        Attendance

    class Attendance(builtins.object)
     |  #create a module attendance.py and include functions mark_present(student),
     |  # mark_absent(student),get_attendace(student). Include docstring documentation in the function.
     |
     |  Methods defined here:
     |
     |  __init__(self)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  get_attendance(self, student)
     |      Gets the attendance status of a student.
     |
     |      Args:
     |          student (str): The name of the student to check attendance for.
     |
     |      Returns:
     |          str: The attendance status of the student ("Present", "Absent", or "Not Recorded").
     |
     |  mark_absent(self, student)
     |      Marks a student as absent.
     |
     |      Args:
```



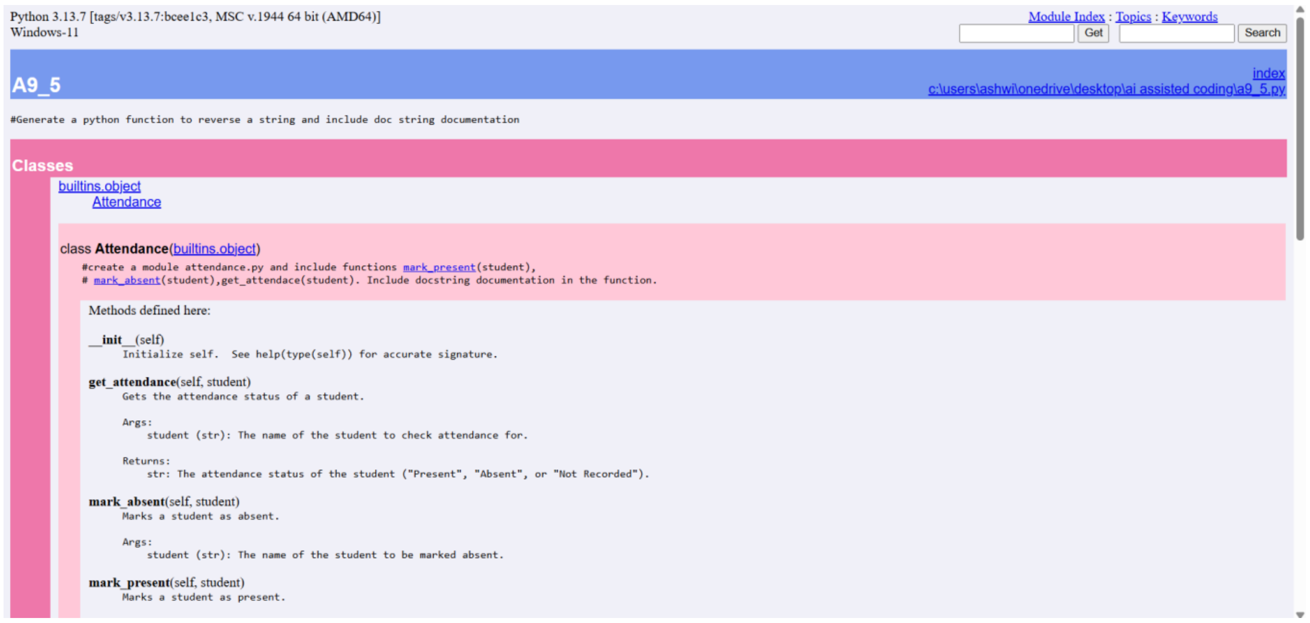
```
(.venv) PS C:\Users\ashwi\OneDrive\Desktop\AI Assisted Coding> python -m pydoc A9_5

        Returns:
            int: The square of the number.

DATA
    attendance = <A9_5.Attendance object>

FILE
    c:\users\ashwi\onedrive\desktop\ai assisted coding\a9_5.py


(.venv) PS C:\Users\ashwi\OneDrive\Desktop\AI Assisted Coding>
(.venv) PS C:\Users\ashwi\OneDrive\Desktop\AI Assisted Coding>
(.venv) PS C:\Users\ashwi\OneDrive\Desktop\AI Assisted Coding> python -m pydoc -w A9_5
All test cases passed!
All test cases passed!
All test cases passed!
All test cases passed!
wrote A9_5.html
(.venv) PS C:\Users\ashwi\OneDrive\Desktop\AI Assisted Coding> python -m pydoc -p 1234
Server ready at http://localhost:1234/
Server commands: [b]rowser, [q]uit
server> All test cases passed!
All test cases passed!
All test cases passed!
All test cases passed!
All test cases passed!
All test cases passed!
All test cases passed!
All test cases passed!
```



Python 3.13.7 [tags/v3.13.7:bceelc3, MSC v.1944 64 bit (AMD64)]
Windows-11

c:\users\ashwi\onedrive\desktop\ai assisted coding\a9_5.py

A9_5

```
#Generate a python function to reverse a string and include doc string documentation
```

Classes

builtins.object
    Attendance

class **Attendance**(builtins.object)

```
#create a module attendance.py and include functions mark_present(student),
# mark_absent(student),get_attendace(student). Include docstring documentation in the function.
```

Methods defined here:

__init__(self)

```
    Initialize self.  See help(type(self)) for accurate signature.
```

get_attendance(self, student)

```
    Gets the attendance status of a student.

    Args:
        student (str): The name of the student to check attendance for.

    Returns:
        str: The attendance status of the student ("Present", "Absent", or "Not Recorded").
```

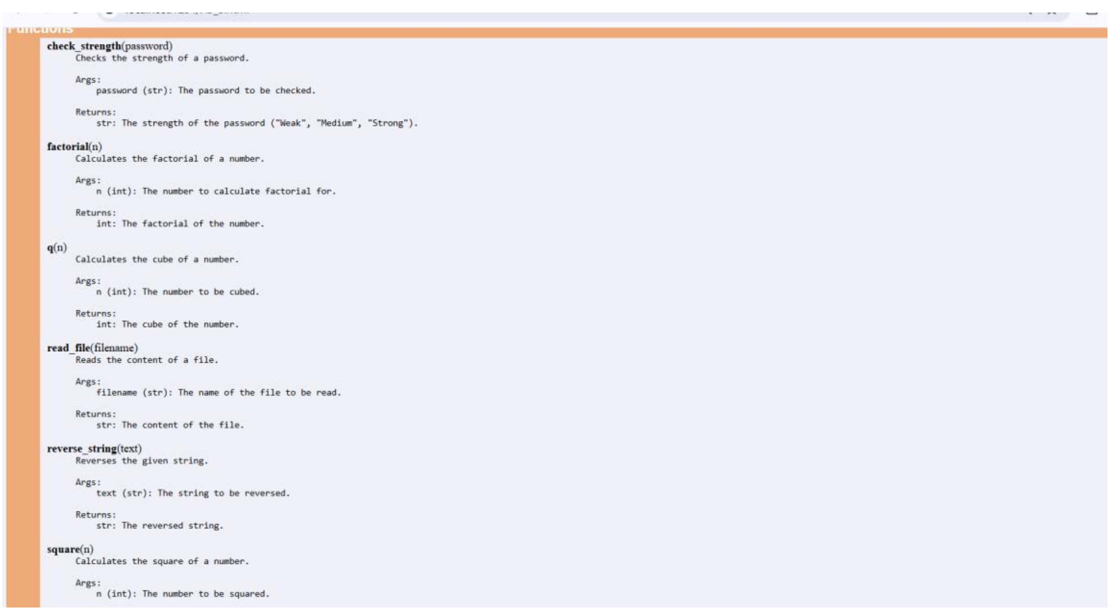
mark_absent(self, student)

```
    Marks a student as absent.

    Args:
        student (str): The name of the student to be marked absent.
```

mark_present(self, student)

```
    Marks a student as present.
```



Functions

check_strength(password)
```
    Checks the strength of a password.

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

    Returns:
        str: The strength of the password ("Weak", "Medium", "Strong").
```

factorial(n)
```
    Calculates the factorial of a number.

    Args:
        n (int): The number to calculate factorial for.

    Returns:
        int: The factorial of the number.
```

q(n)
```
    Calculates the cube of a number.

    Args:
        n (int): The number to be cubed.

    Returns:
        int: The cube of the number.
```

read_file(filename)
```
    Reads the content of a file.

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

    Returns:
        str: The content of the file.
```

reverse_string(text)
```
    Reverses the given string.

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

    Returns:
        str: The reversed string.
```

square(n)
```
    Calculates the square of a number.

    Args:
        n (int): The number to be squared.
```