Assignment – 9.5

Name : A.Srivani

Ht.no : 2303A51272

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

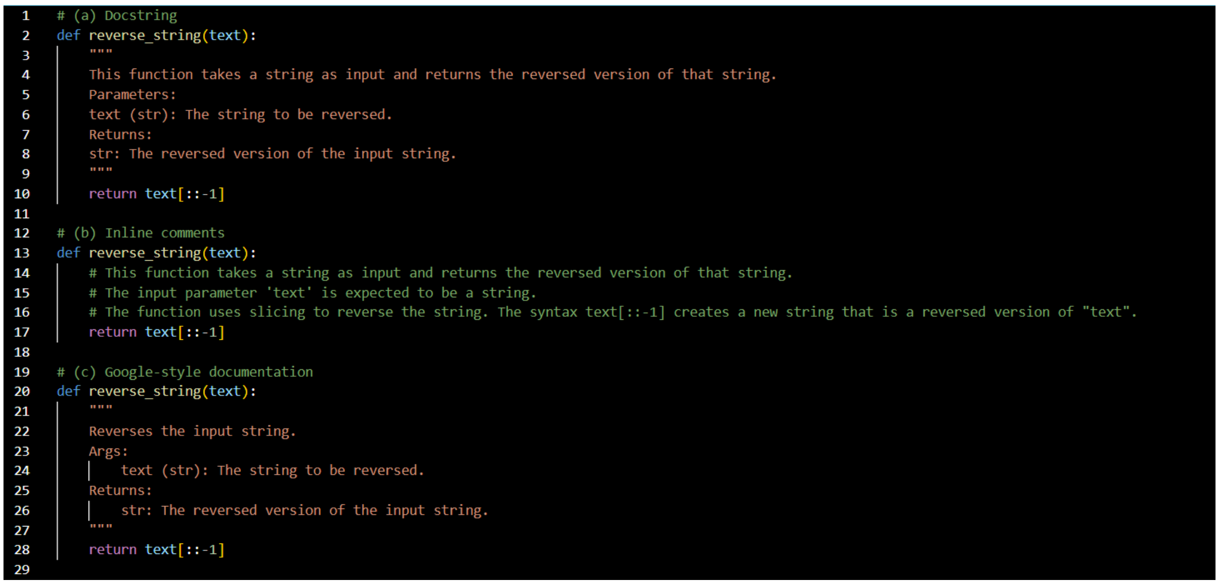
comments o (c) Google-style

documentation

2. Compare the three documentation styles.

3. Recommend the most suitable style for a utility-based string

library.

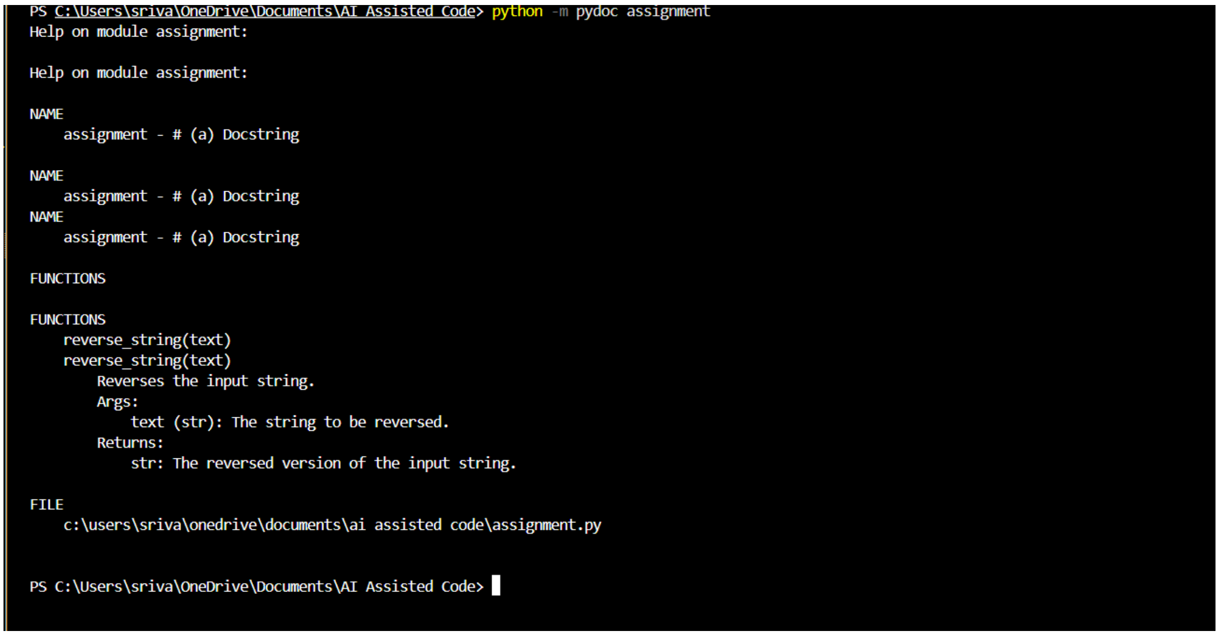


```python
# (a) Docstring
def reverse_string(text):
    """
    This function takes a string as input and returns the reversed version of that string.
    Parameters:
    text (str): The string to be reversed.
    Returns:
    str: The reversed version of the input string.
    """
    return text[::-1]

# (b) Inline comments
def reverse_string(text):
    # This function takes a string as input and returns the reversed version of that string.
    # The input parameter 'text' is expected to be a string.
    # The function uses slicing to reverse the string. The syntax text[::-1] creates a new string that is a reversed version of "text".
    return text[::-1]

# (c) Google-style documentation
def reverse_string(text):
    """
    Reverses the input string.
    Args:
        text (str): The string to be reversed.
    Returns:
        str: The reversed version of the input string.
    """
    return text[::-1]

```



```
PS C:\Users\sriva\OneDrive\Documents\AI Assisted Code> python -m pydoc assignment
Help on module assignment:

Help on module assignment:

NAME
    assignment - # (a) Docstring

NAME
    assignment - # (a) Docstring
NAME
    assignment - # (a) Docstring

FUNCTIONS

FUNCTIONS
    reverse_string(text)
    reverse_string(text)
        Reverses the input string.
        Args:
            text (str): The string to be reversed.
        Returns:
            str: The reversed version of the input string.

FILE
    c:\users\sriva\onedrive\documents\ai assisted code\assignment.py


PS C:\Users\sriva\OneDrive\Documents\AI Assisted Code> 
```

Problem 2: Password Strength Checker

Consider the function: def

check_strength(password):

return len(password) >= 8

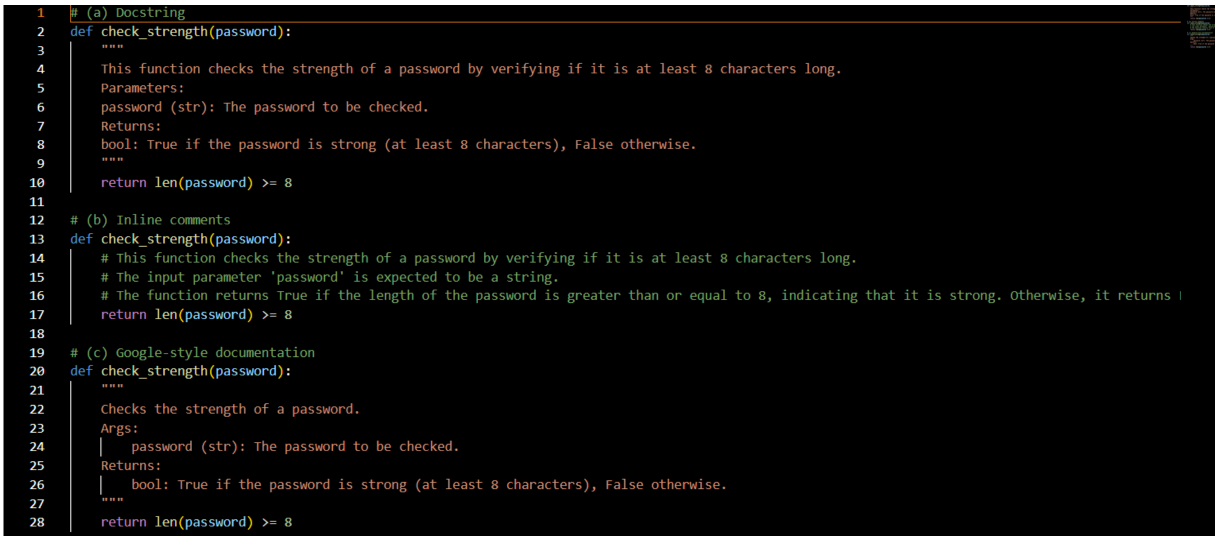
Task:

1. Document the function using docstring, inline comments, and

Google style.

2. Compare documentation styles for security-related code.

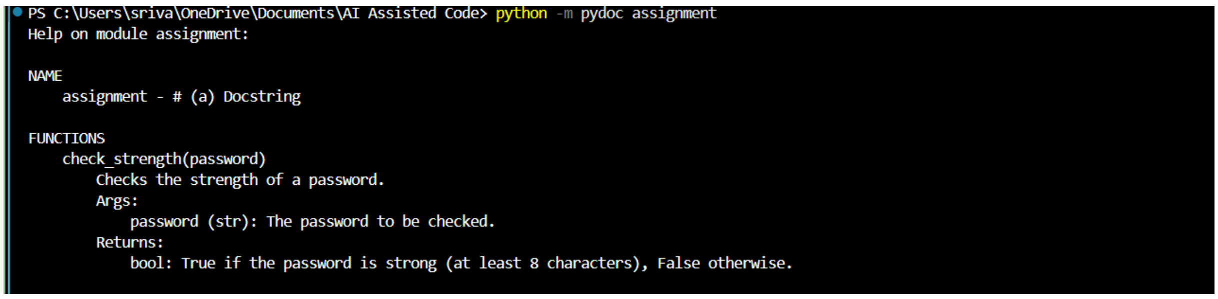
3. Recommend the most appropriate style.



```python
# (a) Docstring
def check_strength(password):
    """
    This function checks the strength of a password by verifying if it is at least 8 characters long.
    Parameters:
    password (str): The password to be checked.
    Returns:
    bool: True if the password is strong (at least 8 characters), False otherwise.
    """
    return len(password) >= 8

# (b) Inline comments
def check_strength(password):
    # This function checks the strength of a password by verifying if it is at least 8 characters long.
    # The input parameter 'password' is expected to be a string.
    # The function returns True if the length of the password is greater than or equal to 8, indicating that it is strong. Otherwise, it returns
    return len(password) >= 8

# (c) Google-style documentation
def check_strength(password):
    """
    Checks the strength of a password.
    Args:
        password (str): The password to be checked.
    Returns:
        bool: True if the password is strong (at least 8 characters), False otherwise.
    """
    return len(password) >= 8
```



```
PS C:\Users\sriva\OneDrive\Documents\AI Assisted Code> python -m pydoc assignment
Help on module assignment:

NAME
    assignment - # (a) Docstring

FUNCTIONS
    check_strength(password)
        Checks the strength of a password.
        Args:
            password (str): The password to be checked.
        Returns:
            bool: True if the password is strong (at least 8 characters), False otherwise.
```

Problem 3: Math Utilities Module

Task:

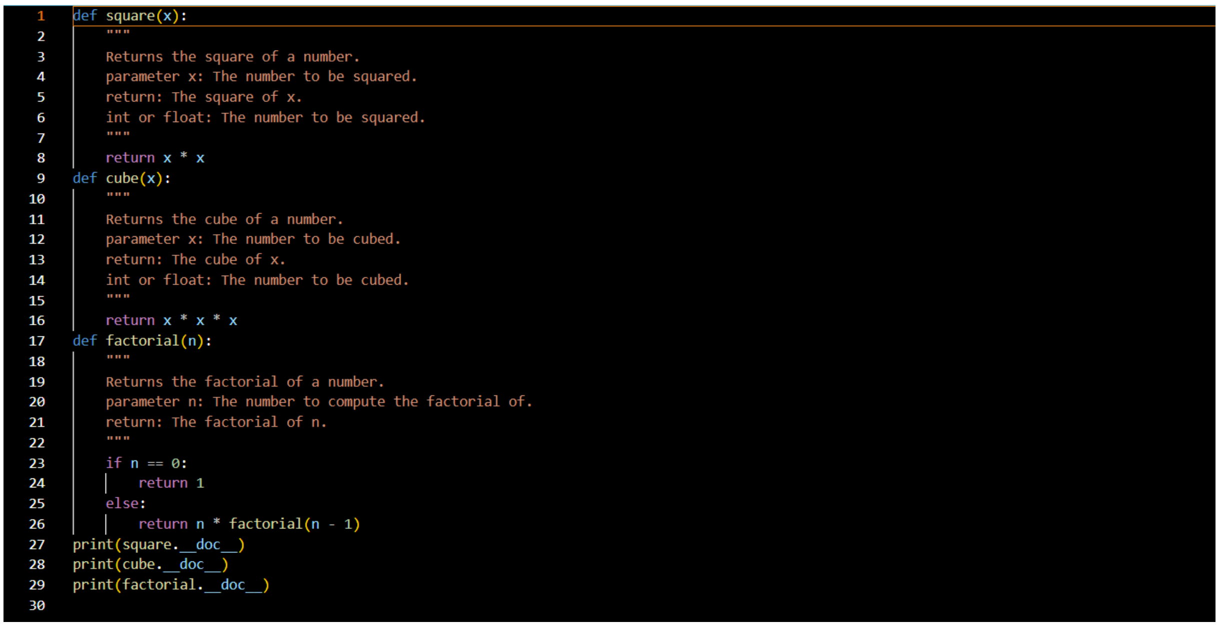
1. Create a module math_utils.py with functions:

o square(n) o cube(n)

o factorial(n)

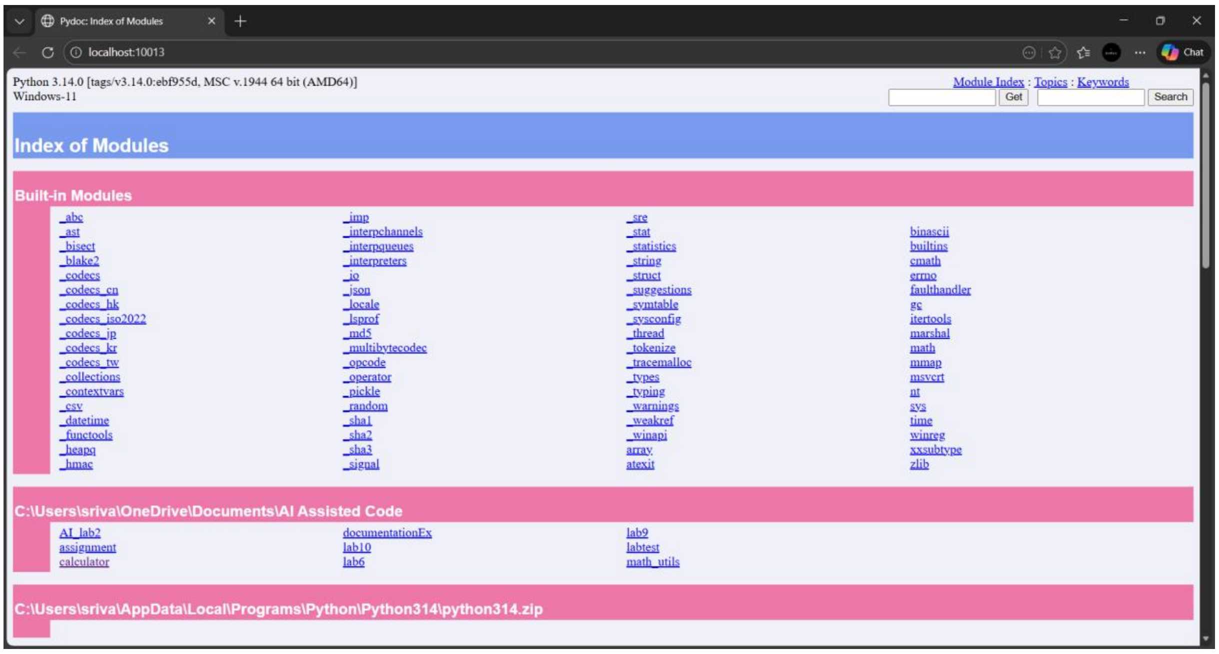
2. Generate docstrings automatically using AI tools.

3. Export documentation as an HTML file.



```python
def square(x):
    """
    Returns the square of a number.
    parameter x: The number to be squared.
    return: The square of x.
    int or float: The number to be squared.
    """
    return x * x
def cube(x):
    """
    Returns the cube of a number.
    parameter x: The number to be cubed.
    return: The cube of x.
    int or float: The number to be cubed.
    """
    return x * x * x
def factorial(n):
    """
    Returns the factorial of a number.
    parameter n: The number to compute the factorial of.
    return: The factorial of n.
    """
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
print(square.__doc__)
print(cube.__doc__)
print(factorial.__doc__)

```


Python 3.14.0 [tags/v3.14.0:ebf955d, MSC v.1944 64 bit (AMD64)]
Windows-11

Module Index : Topics : Keywords

Index of Modules

Built-in Modules

_abc	_imp	_sre	
_ast	_interpchannels	_stat	binascii
_bisect	_interpqueues	_statistics	builtins
_blake2	_interpreters	_string	cmath
_codecs	_io	_struct	errno
_codecs_cn	_json	_suggestions	faulthandler
_codecs_hk	_locale	_symtable	gc
_codecs_iso2022	_lsprof	_sysconfig	itertools
_codecs_jp	_md5	_thread	marshal
_codecs_kr	_multibytecodec	_tokenize	math
_codecs_tw	_opcode	_tracemalloc	mmap
_collections	_operator	_types	msvcrt
_contextvars	_pickle	_typing	nt
_csv	_random	_warnings	sys
_datetime	_sha1	_weakref	time
_functools	_sha2	_winapi	winreg
_heapq	_sha3	array	xxsubtype
_hmac	_signal	atexit	zlib

C:\Users\sriva\OneDrive\Documents\AI Assisted Code

AI_lab2	documentationEx	lab9
assignment	lab10	labtest
calculator	lab6	math_utils

C:\Users\sriva\AppData\Local\Programs\Python\Python314\python314.zip

Problem 4: Attendance Management Module

Task:

1. Create a module attendance.py with functions:

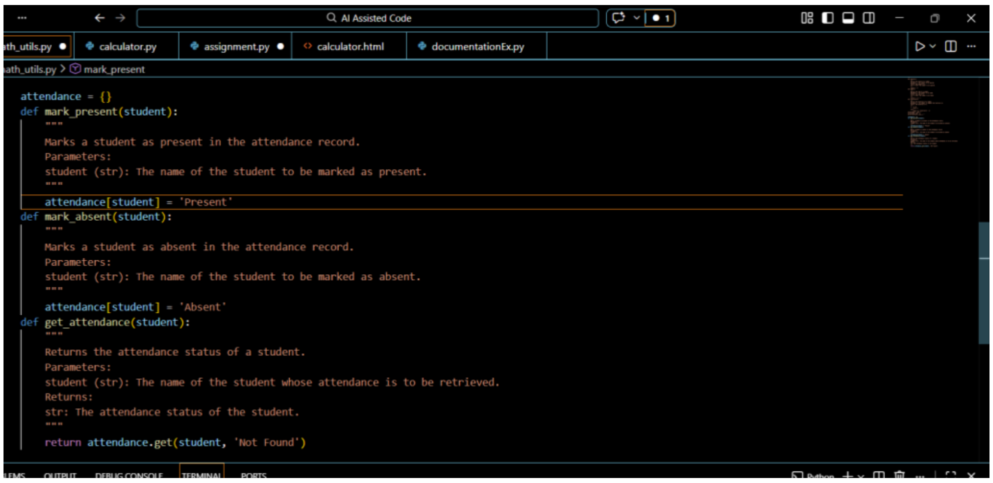
o mark_present(student) o

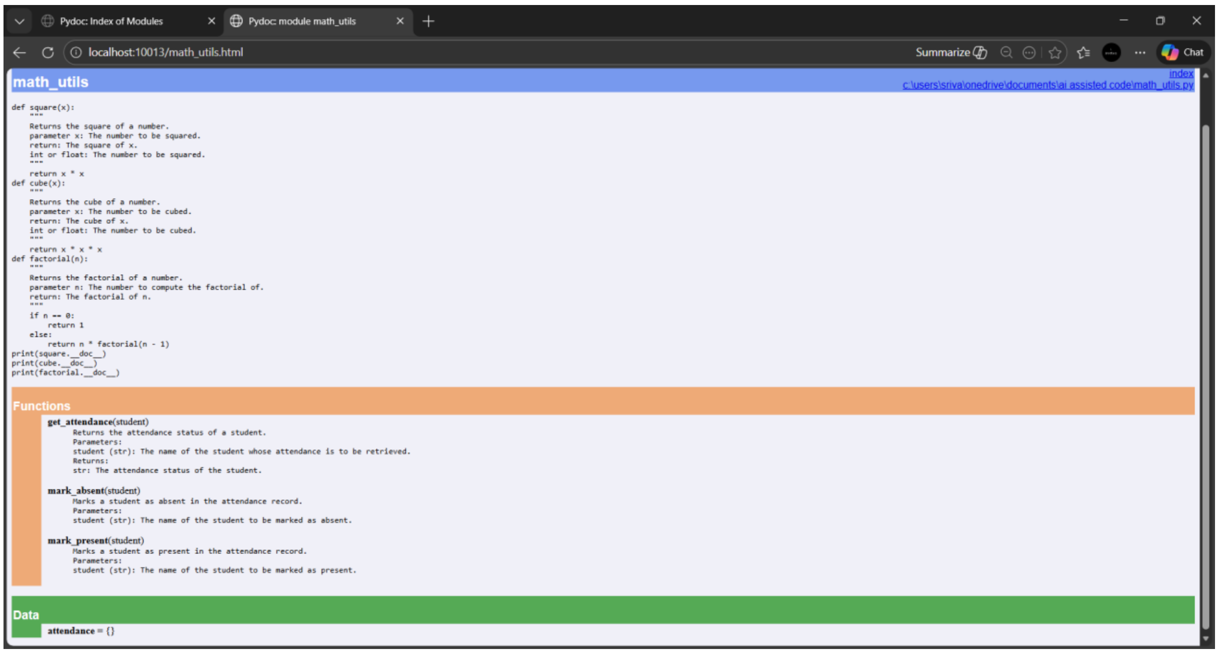
mark_absent(student) o

get_attendance(student)

2. Add proper docstrings.

3. Generate and view documentation in terminal and browse


```
attendance = {}
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.
    """
    attendance[student] = 'Present'
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.
    """
    attendance[student] = 'Absent'
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.
    """
    return attendance.get(student, 'Not Found')
```



math_utils

index
c:\users\sriva\onedrive\documents\ai assisted code\math_utils.py

```
def square(x):
    """
    Returns the square of a number.
    parameter x: The number to be squared.
    return: The square of x.
    int or float: The number to be squared.
    """
    return x * x
def cube(x):
    """
    Returns the cube of a number.
    parameter x: The number to be cubed.
    return: The cube of x.
    int or float: The number to be cubed.
    """
    return x * x * x
def factorial(n):
    """
    Returns the factorial of a number.
    parameter n: The number to compute the factorial of.
    return: The factorial of n.
    """
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
print(square.__doc__)
print(cube.__doc__)
print(factorial.__doc__)
```

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

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.
```

Data

attendance = {}

Problem 5: File Handling Function

Consider the function:

def read_file(filename):

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

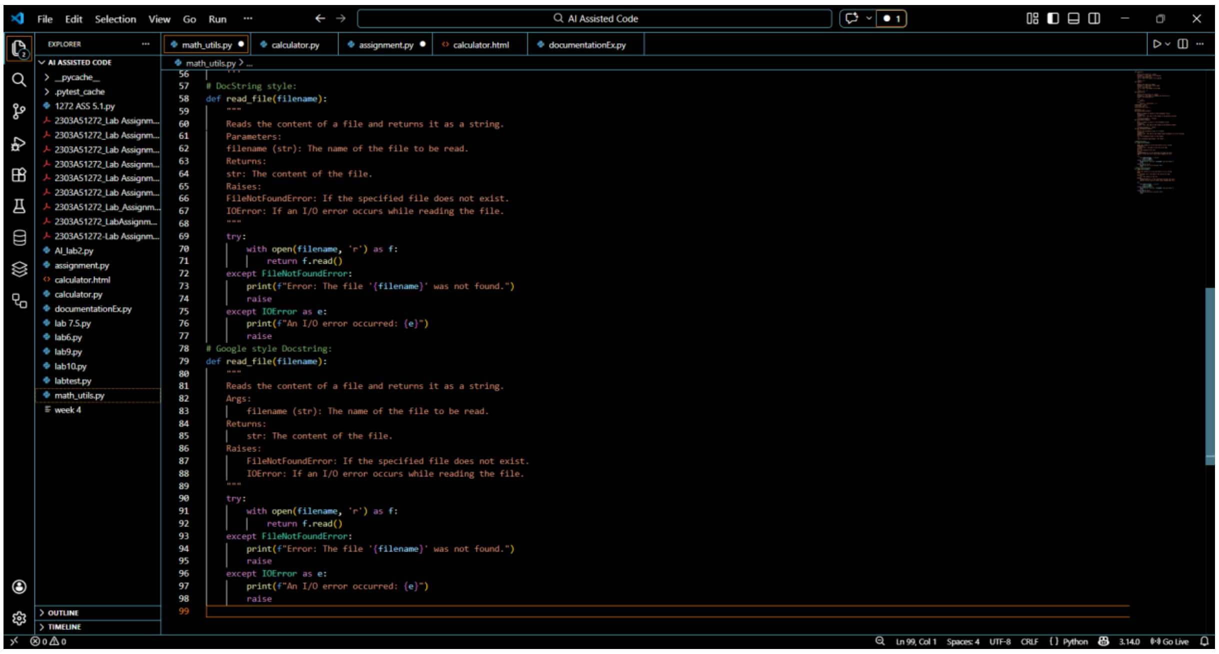
return f.read()

Task:

1. Write documentation using all three formats.

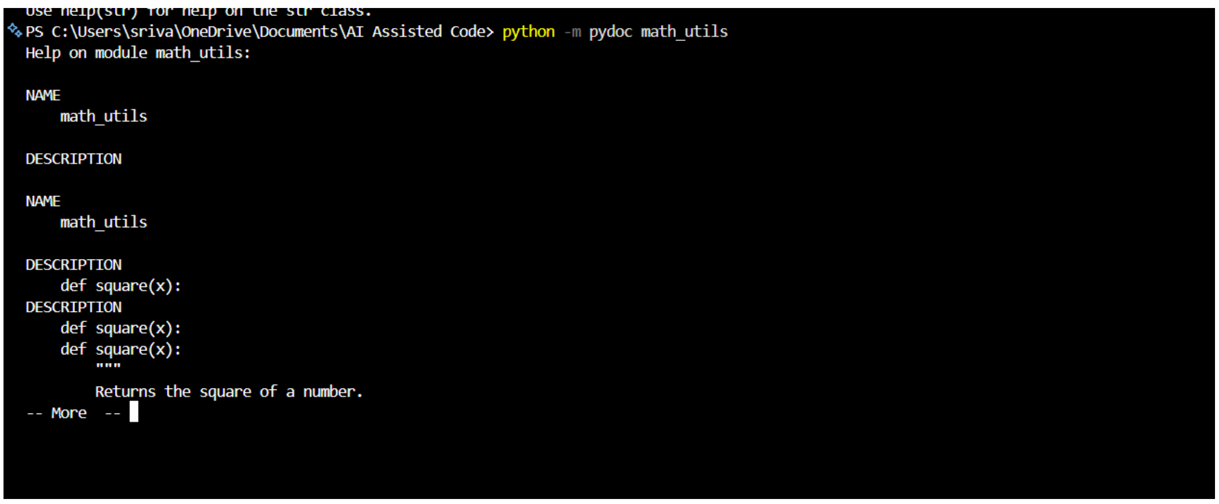
2. Identify which style best explains exception handling.

3. Justify your recommendation.



```python
    '''
# DocString style:
def read_file(filename):
    """
    Reads the content of a file and returns it as a string.
    Parameters:
    filename (str): The name of the file to be read.
    Returns:
    str: The content of the file.
    Raises:
    FileNotFoundError: If the specified file does not exist.
    IOError: If an I/O error occurs while reading the file.
    """
    try:
        with open(filename, 'r') as f:
            return f.read()
    except FileNotFoundError:
        print(f"Error: The file '{filename}' was not found.")
        raise
    except IOError as e:
        print(f"An I/O error occurred: {e}")
        raise
# Google style Docstring:
def read_file(filename):
    """
    Reads the content of a file and returns it as a string.
    Args:
        filename (str): The name of the file to be read.
    Returns:
        str: The content of the file.
    Raises:
        FileNotFoundError: If the specified file does not exist.
        IOError: If an I/O error occurs while reading the file.
    """
    try:
        with open(filename, 'r') as f:
            return f.read()
    except FileNotFoundError:
        print(f"Error: The file '{filename}' was not found.")
        raise
    except IOError as e:
        print(f"An I/O error occurred: {e}")
        raise

```



```
  Use help(str) for help on the str class.
PS C:\Users\sriva\OneDrive\Documents\AI Assisted Code> python -m pydoc math_utils
  Help on module math_utils:

  NAME
      math_utils

  DESCRIPTION

  NAME
      math_utils

  DESCRIPTION
      def square(x):
  DESCRIPTION
      def square(x):
      def square(x):
          """
          Returns the square of a number.
  -- More  --
```