

AI ASSISTANT CODING

2303A51944

Ch .Tejaswi

ASSIGNMENT 9.1

QUESTION 1

PROMPT:

Write documentation in:

1. Docstring format
2. Inline comments
3. Google-style documentation

Also compare the three approaches and recommend the best style for a mathematical utilities library.

CODE:

```
python

def find_max(numbers):
    """
    Returns the maximum value from a list of numbers.

    Parameters:
    numbers (list): List of numerical values.

    Returns:
    int or float: Maximum value from the list.
    """
    return max(numbers)
```

INLINE COMMENTS:

```
python

def find_max(numbers):
    # Returns the largest element in the list
    return max(numbers)
```

GOOGLE STYLE DOCUMENTATION:

```
def find_max(numbers):  
    """  
    Finds the maximum value in a list.  
  
    Args:  
        numbers (list): List of numbers.  
  
    Returns:  
        int or float: Maximum value.  
    """  
    return max(numbers)
```

Output

less

For input: [2, 5, 1, 9]

Output: 9


QUESTION 2:

PROMPT:

Write documentation in three formats and recommend the best style for new developers.


CODE:

python

 Copy code

```
def login(user, password, credentials):  
    """  
    Validates user login credentials.  
  
    Parameters:  
    user (str): Username  
    password (str): Password  
    credentials (dict): Stored credentials  
  
    Returns:  
    bool: True if valid, else False  
    """  
    return credentials.get(user) == password
```

python


 Copy code

```
def login(user, password, credentials):  
    # Check if password matches stored credenti  
    return credentials.get(user) == password
```

```
def login(user, password, credentials):  
    """  
    Authenticates a user.  
  
    Args:  
        user (str): Username  
        password (str): Password  
        credentials (dict): User credential store  
  
    Returns:  
        bool: Authentication result  
    """  
    return credentials.get(user) == password
```

Output

pgsql

 Copy code

Input: user="admin", password="1234"

Output: True / False

EXPLANATION:

Google-style documentation is most helpful for new developers as it clearly explains parameters and return values in a structured manner.

QUESTION 3:

PROMPT:

Design a Python module calculator.py and demonstrate automatic documentation generation.

CODE:

```
"""
```

Conversion Utilities Module

```
"""
```

```
def decimal_to_binary(n):
```

```
    """Converts decimal to binary."""
```

```
    return bin(n)[2:]
```

```
def binary_to_decimal(b):
```

```
    """Converts binary to decimal."""
```

```
    return int(b, 2)
```

```
def decimal_to_hexadecimal(n):
```

```
    """Converts decimal to hexadecimal."""
```

```
    return hex(n)[2:]
```

Output

Terminal Documentation:

```
bash
```

```
python -m pydoc calculator
```

HTML Documentation Generated:

```
bash
```

```
python -m pydoc -w calculator
```

EXPLANATION:

Explanation

- Docstrings allow automatic documentation generation.
- pydoc extracts function and module details.
- HTML output provides readable documentation.

PROBLEM 4:

PROMPT:

Create conversion.py and generate documentation automatically.

CODE:

python

```
"""  
  
Conversion Utilities Module  
  
"""  
  
def decimal_to_binary(n):  
    """Converts decimal to binary."""  
    return bin(n)[2:]  
  
def binary_to_decimal(b):  
    """Converts binary to decimal."""  
    return int(b, 2)  
  
def decimal_to_hexadecimal(n):  
    """Converts decimal to hexadecimal."""  
    return hex(n)[2:]
```



OUTPUT:

```
bash

python -m pydoc conversion
python -m pydoc -w conversion
```

EXPLANATION:

Automatic documentation improves understanding of conversion utilities and supports reuse.

PROBLEM 5:

PROMPT:

Create course.py and generate documentation.

CODE:

```
"""
Course Management Module
"""

courses = {}

def add_course(course_id, name, credits):
    """Adds a course."""
    courses[course_id] = {"name": name, "credits": credits}

def remove_course(course_id):
    """Removes a course."""
    return courses.pop(course_id, None)

def get_course(course_id):
    """Gets course details."""
    return courses.get(course_id)
```


OUTPUT:

Output

```
bash
```

```
python -m pydoc course
```

```
python -m pydoc -w course
```

Explanation

Course functions are documented using docstrings which allow automatic documentation generation.