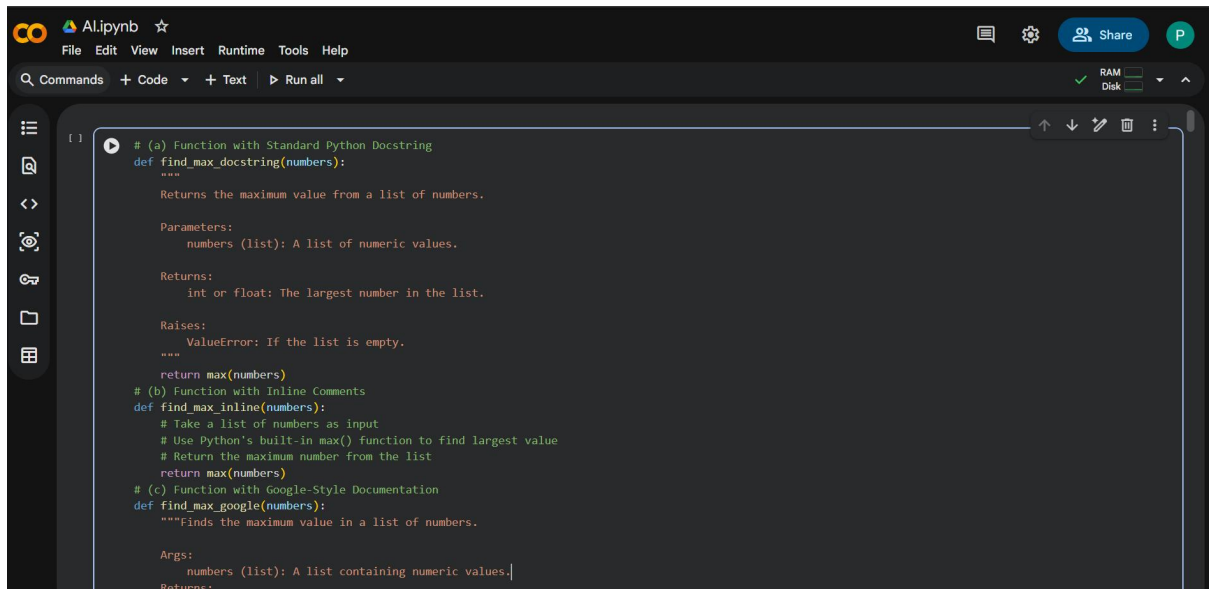


AI

NAME:-P.LIKITHA

HT NO:-2303A52393

BATCH-43



```
[1] # (a) Function with Standard Python Docstring
def find_max_docstring(numbers):
    """
    Returns the maximum value from a list of numbers.

    Parameters:
        numbers (list): A list of numeric values.

    Returns:
        int or float: The largest number in the list.

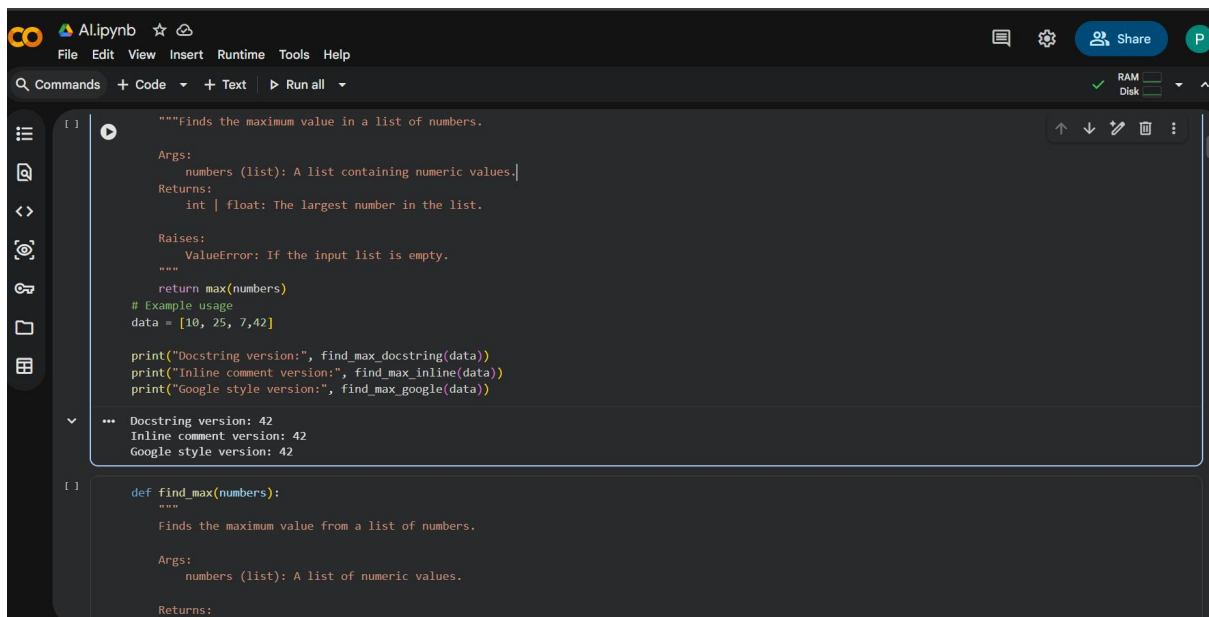
    Raises:
        ValueError: If the list is empty.
    """
    return max(numbers)

# (b) Function with Inline Comments
def find_max_inline(numbers):
    # Take a list of numbers as input
    # Use Python's built-in max() function to find largest value
    # Return the maximum number from the list
    return max(numbers)

# (c) Function with Google-Style Documentation
def find_max_google(numbers):
    """Finds the maximum value in a list of numbers.

    Args:
        numbers (list): A list containing numeric values.

    Returns:
```



```
[1] """Finds the maximum value in a list of numbers.

    Args:
        numbers (list): A list containing numeric values.

    Returns:
        int | float: The largest number in the list.

    Raises:
        ValueError: If the input list is empty.
    """
    return max(numbers)

# Example usage
data = [10, 25, 7, 42]

print("Docstring version:", find_max_docstring(data))
print("Inline comment version:", find_max_inline(data))
print("Google style version:", find_max_google(data))

... Docstring version: 42
    Inline comment version: 42
    Google style version: 42

[1] def find_max(numbers):
    """
    Finds the maximum value from a list of numbers.

    Args:
        numbers (list): A list of numeric values.

    Returns:
```



Alipyb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

Share

RAM Disk

```
[ ]

'''
Calculator Module
Provides basic arithmetic operations.
'''

def add(a, b):
    '''
    Returns the sum of two numbers.
    '''
    return a + b

def subtract(a, b):
    '''
    Returns the difference of two numbers.
    '''
    return a - b

def multiply(a, b):
    '''
    Returns the product of two numbers.
    '''
    return a * b

def divide(a, b):
    '''
    Returns the quotient of two numbers.
    '''
```

Alipyb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

Share

RAM Disk

```
[ ]

    Returns the product of two numbers.
    '''
    return a * b

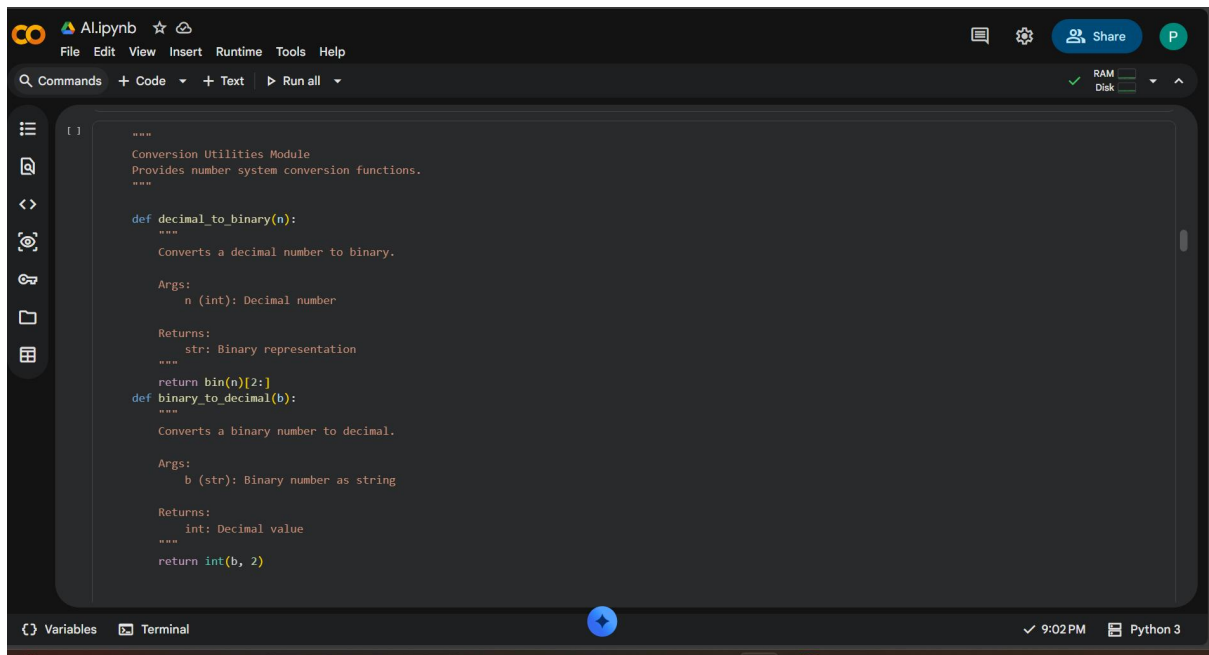
def divide(a, b):
    '''
    Returns the quotient of two numbers.

    Raises:
        ValueError: If division by zero is attempted
    '''
    if b == 0:
        raise ValueError("Cannot divide by zero")
    return a / b

# Example usage
print("Add:", add(5, 3))
print("Subtract:", subtract(5, 3))
print("Multiply:", multiply(5, 3))
print("Divide:", divide(10, 2))
```

▼

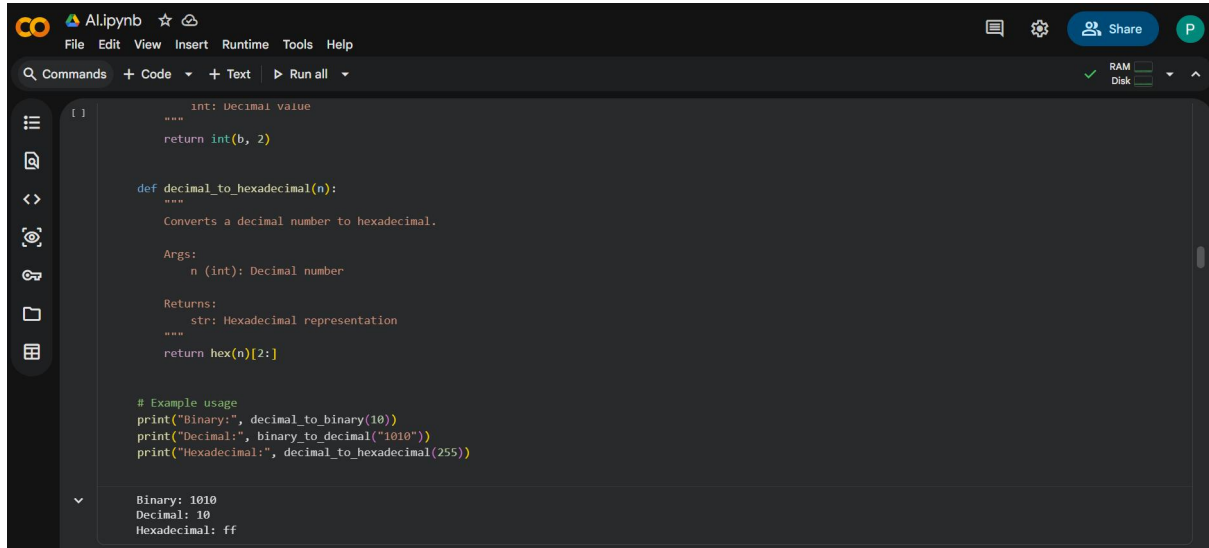
Add: 8  
Subtract: 2  
Multiply: 15  
Divide: 5.0



The image shows the Alipython IDE interface. The top bar includes the Alipython logo, a star icon, and a share icon. Below this is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. A search bar labeled 'Commands' is on the left, and a 'Run all' button is on the right. The main editor area contains the following Python code:

```
[ ]  
  
"""  
Conversion Utilities Module  
Provides number system conversion functions.  
"""  
  
def decimal_to_binary(n):  
    """  
    Converts a decimal number to binary.  
  
    Args:  
        n (int): Decimal number  
  
    Returns:  
        str: Binary representation  
    """  
    return bin(n)[2:]  
def binary_to_decimal(b):  
    """  
    Converts a binary number to decimal.  
  
    Args:  
        b (str): Binary number as string  
  
    Returns:  
        int: Decimal value  
    """  
    return int(b, 2)
```

The bottom status bar shows 'Variables', 'Terminal', a blue plus icon, '9:02 PM', and 'Python 3'.



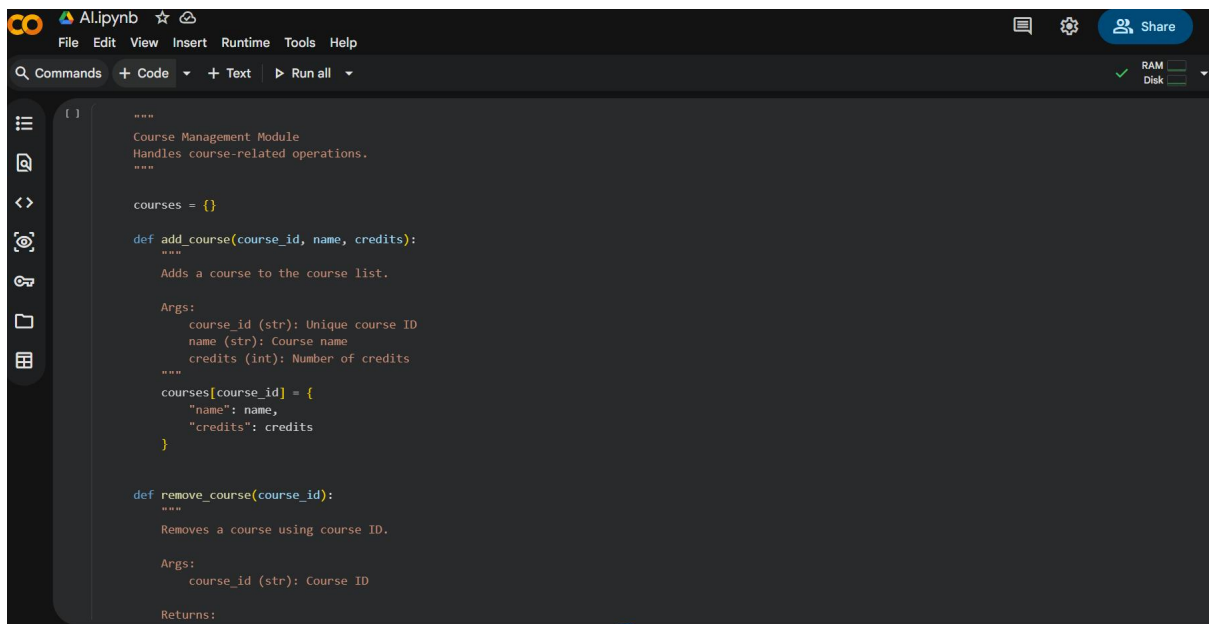
The image shows the Alipython IDE interface with the code from the previous image completed. The code now includes a third function, `decimal_to_hexadecimal`, and example usage. The output of the code is displayed at the bottom.

```
[ ]  
  
    int: Decimal value  
    """  
    return int(b, 2)  
  
def decimal_to_hexadecimal(n):  
    """  
    Converts a decimal number to hexadecimal.  
  
    Args:  
        n (int): Decimal number  
  
    Returns:  
        str: Hexadecimal representation  
    """  
    return hex(n)[2:]  
  
# Example usage  
print("Binary:", decimal_to_binary(10))  
print("Decimal:", binary_to_decimal("1010"))  
print("Hexadecimal:", decimal_to_hexadecimal(255))
```

The output at the bottom is:

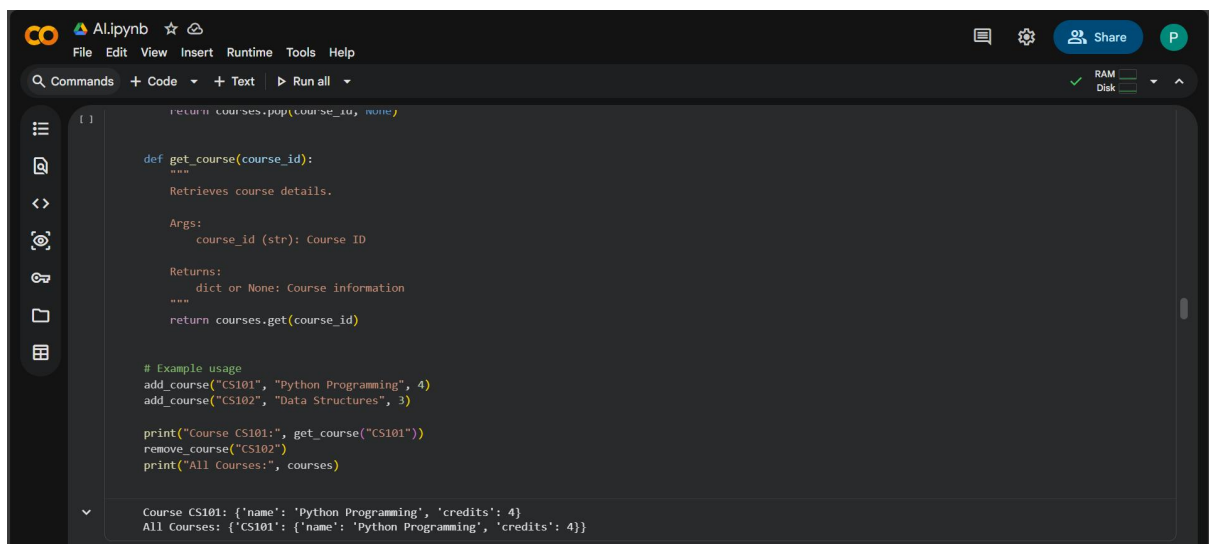
```
Binary: 1010  
Decimal: 10  
Hexadecimal: ff
```

The bottom status bar shows 'Variables', 'Terminal', a blue plus icon, '9:02 PM', and 'Python 3'.



The image shows the Alipython IDE interface. The top bar includes the Alipython logo, a star icon, and a share button. Below the bar is a menu with File, Edit, View, Insert, Runtime, Tools, and Help. A search bar with 'Commands' is on the left, and a 'Run all' button is on the right. The main editor area contains a Python script for a 'Course Management Module'. The script defines a dictionary 'courses' and two functions: 'add\_course' and 'remove\_course'. The 'add\_course' function takes 'course\_id', 'name', and 'credits' as arguments and adds a new course to the dictionary. The 'remove\_course' function takes 'course\_id' as an argument and removes the course from the dictionary. The script is as follows:

```
[ ]  
  
"""  
Course Management Module  
Handles course-related operations.  
"""  
  
courses = {}  
  
def add_course(course_id, name, credits):  
    """  
    Adds a course to the course list.  
  
    Args:  
        course_id (str): Unique course ID  
        name (str): Course name  
        credits (int): Number of credits  
    """  
    courses[course_id] = {  
        "name": name,  
        "credits": credits  
    }  
  
def remove_course(course_id):  
    """  
    Removes a course using course ID.  
  
    Args:  
        course_id (str): Course ID  
  
    Returns:
```



The image shows the Alipython IDE interface with the continuation of the Python script. The script defines a 'get\_course' function that retrieves course details from the 'courses' dictionary. It also includes example usage code that adds two courses, removes one, and prints the results. The output of the script is displayed at the bottom of the editor area.

```
[ ]  
  
    return courses.pop(course_id, None)  
  
def get_course(course_id):  
    """  
    Retrieves course details.  
  
    Args:  
        course_id (str): Course ID  
  
    Returns:  
        dict or None: Course information  
    """  
    return courses.get(course_id)  
  
# Example usage  
add_course("CS101", "Python Programming", 4)  
add_course("CS102", "Data Structures", 3)  
  
print("Course CS101:", get_course("CS101"))  
remove_course("CS102")  
print("All Courses:", courses)  
  
Course CS101: {'name': 'Python Programming', 'credits': 4}  
All Courses: {'CS101': {'name': 'Python Programming', 'credits': 4}}
```