SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**	
Program Name: M. Tech/MCA/MSC	**Assignment Type: Lab**		**AcademicYear:**2025-2026
Course Coordinator Name		Venkataramana Veeramsetty	
Course Code		**Course Title**	AI Assisted Problem Solving Using Python
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week5- Tuesday	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	
AssignmentNumber:12.3(Present assignment number) /**24**(Total number of assignments)			

Q. No.	**Question**	***Expected Time to complete***
1	**Lab 12 – Algorithms with AI Assistance: Sorting, searching, and optimizing algorithms** **Lab Objectives** • To implement classical algorithms (sorting, searching) with the help of AI tools. • To analyze AI suggestions for efficiency and correctness. • To explore AI-assisted optimizations of existing algorithms. • To compare naive vs. optimized approaches generated by AI. **Learning Outcomes** After completing this lab, students will be able to: □ Implement sorting and searching algorithms using AI suggestions. □ Compare AI-generated algorithm variants in terms of readability and efficiency. □ Use AI to optimize brute-force algorithms into more efficient ones. □ Analyze algorithm complexity (time and space) with AI explanations. □ Critically reflect on correctness, clarity, and maintainability of AI-generated algorithms. **Task Description #1 – Linear Search implementation** Task: Write python code for linear_search () function to search a value in a list	Week5- Tuesday

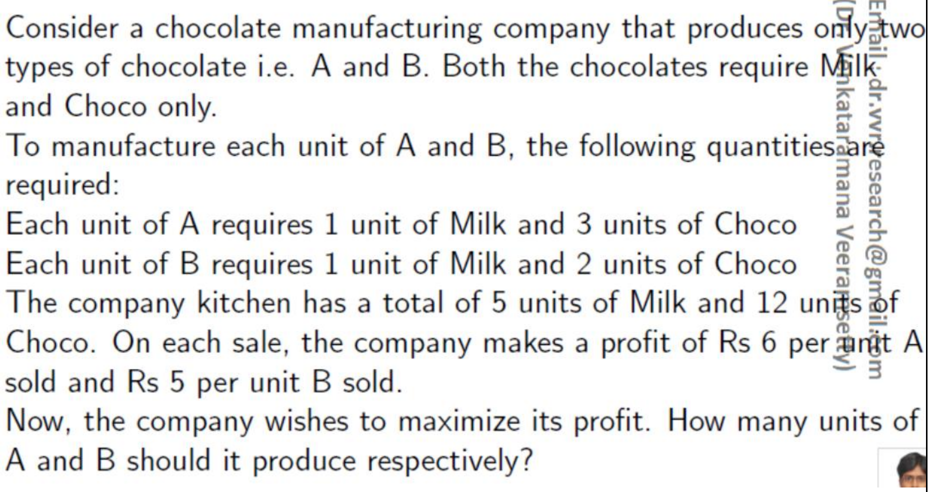
and extract its index.

Task Description #2 – Sorting Algorithms

Task: Ask AI to implement Bubble Sort and check sorted output

Task Description #3 – Optimization

Task: Write python code to solve below case study using linear optimization



Consider a chocolate manufacturing company that produces only two types of chocolate i.e. A and B. Both the chocolates require Milk and Choco only.

To manufacture each unit of A and B, the following quantities are required:

Each unit of A requires 1 unit of Milk and 3 units of Choco

Each unit of B requires 1 unit of Milk and 2 units of Choco

The company kitchen has a total of 5 units of Milk and 12 units of Choco. On each sale, the company makes a profit of Rs 6 per unit A sold and Rs 5 per unit B sold.

Now, the company wishes to maximize its profit. How many units of A and B should it produce respectively?

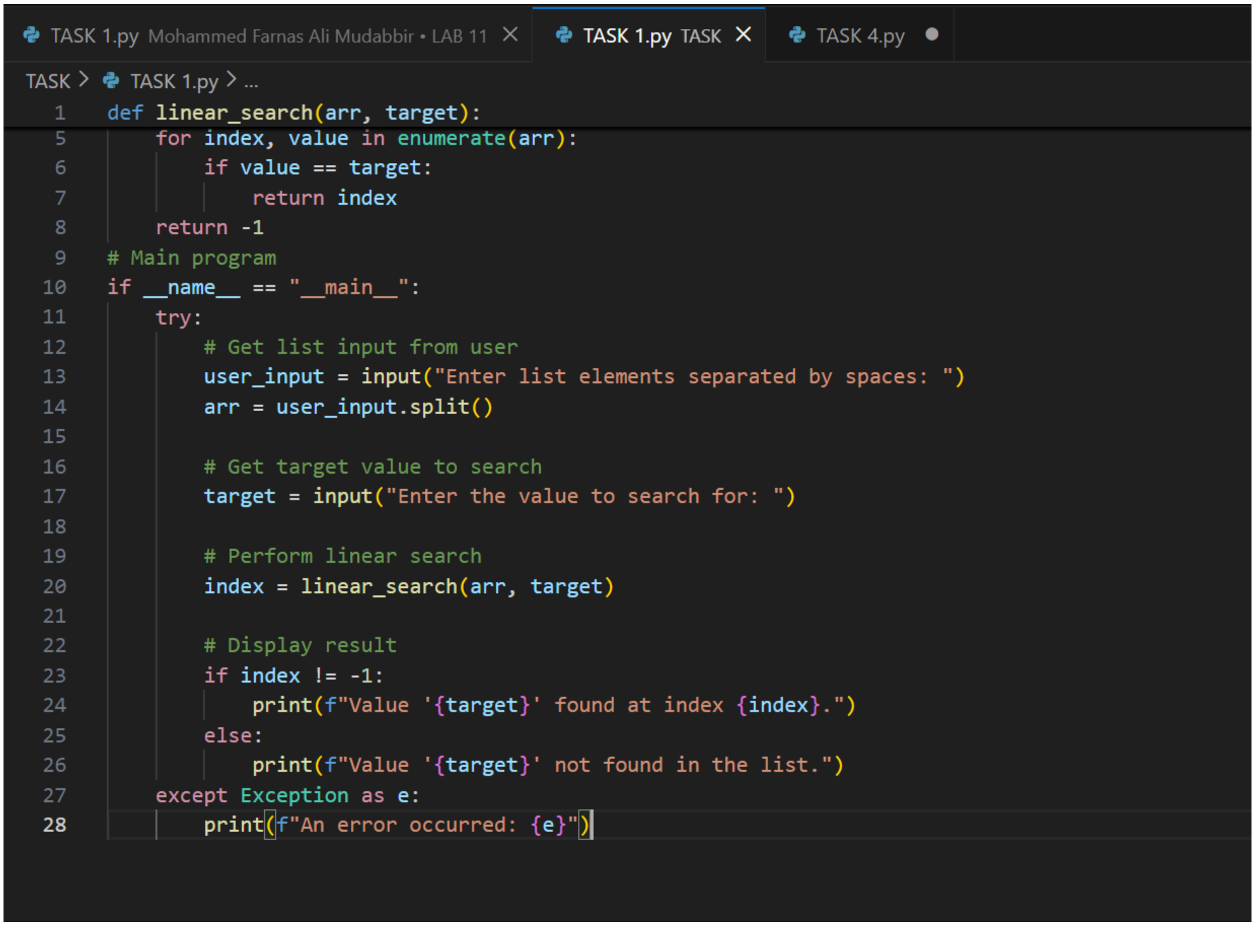
Task Description #4 – Gradient Descent Optimization

Task: Write python code to find value of x at which the function $f(x)=2X^3+4x+5$ will be minimum

Task Description #1 – Linear Search implementation

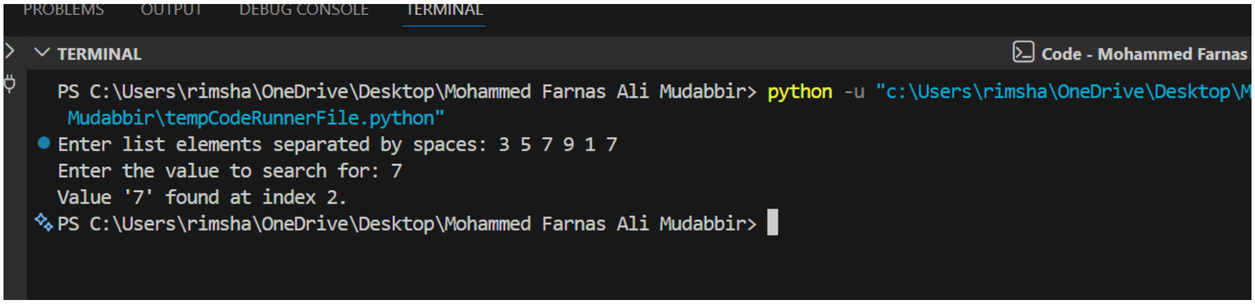
Task: Write python code for linear_search () function to search a value in a list and extract its index.

PROMPT: "Write a Python program that implements a linear_search() function. The program should take a list and a value from the user, search for the value using linear search, and print the index if found



```python
def linear_search(arr, target):
    for index, value in enumerate(arr):
        if value == target:
            return index
    return -1
# Main program
if __name__ == "__main__":
    try:
        # Get list input from user
        user_input = input("Enter list elements separated by spaces: ")
        arr = user_input.split()

        # Get target value to search
        target = input("Enter the value to search for: ")

        # Perform linear search
        index = linear_search(arr, target)

        # Display result
        if index != -1:
            print(f"Value '{target}' found at index {index}.")
        else:
            print(f"Value '{target}' not found in the list.")
    except Exception as e:
        print(f"An error occurred: {e}")
```

PRACTICAL OUT :

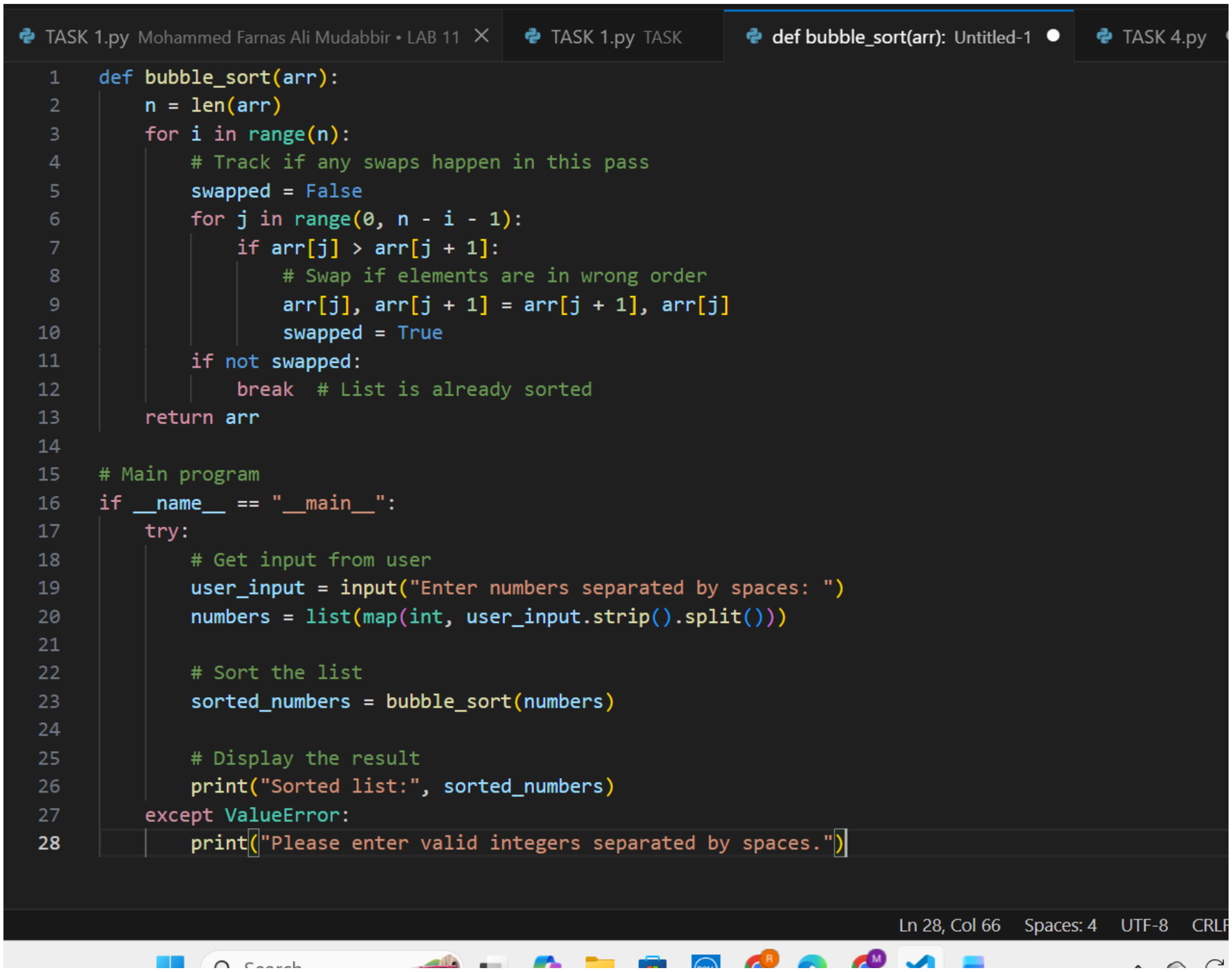


```
PS C:\Users\rimsha\OneDrive\Desktop\Mohammed Farnas Ali Mudabbir> python -u "c:\Users\rimsha\OneDrive\Desktop\M
 Mudabbir\tempCodeRunnerFile.python"
Enter list elements separated by spaces: 3 5 7 9 1 7
Enter the value to search for: 7
Value '7' found at index 2.
PS C:\Users\rimsha\OneDrive\Desktop\Mohammed Farnas Ali Mudabbir>
```

Task Description #2 – Sorting Algorithms

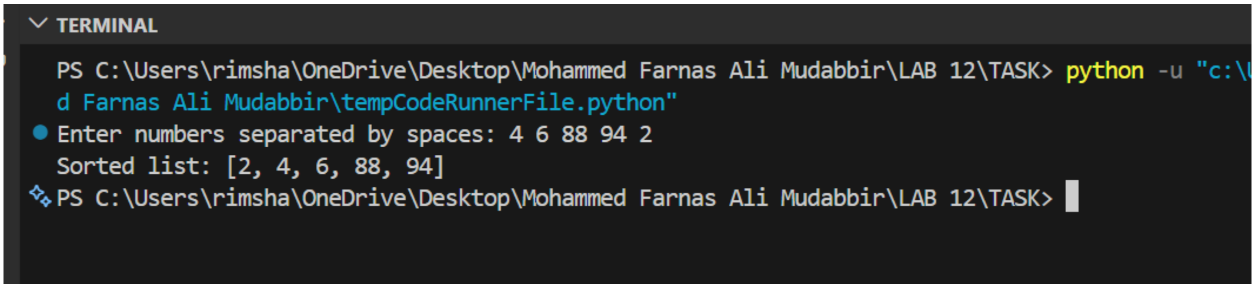
Task: Ask AI to implement Bubble Sort and check sorted output

PROMPT: "Write a Python program that implements Bubble Sort. The program should take a list as user input, sort it using Bubble Sort, and then print the sorted output."


TASK 1.py Mohammed Farnas Ali Mudabbir • LAB 11
TASK 1.py TASK
def bubble_sort(arr): Untitled-1
TASK 4.py
Ln 28, Col 66
Spaces: 4
UTF-8
CRLF

```python
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        # Track if any swaps happen in this pass
        swapped = False
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                # Swap if elements are in wrong order
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
                swapped = True
        if not swapped:
            break  # List is already sorted
    return arr

# Main program
if __name__ == "__main__":
    try:
        # Get input from user
        user_input = input("Enter numbers separated by spaces: ")
        numbers = list(map(int, user_input.strip().split()))

        # Sort the list
        sorted_numbers = bubble_sort(numbers)

        # Display the result
        print("Sorted list:", sorted_numbers)
    except ValueError:
        print("Please enter valid integers separated by spaces.")
```

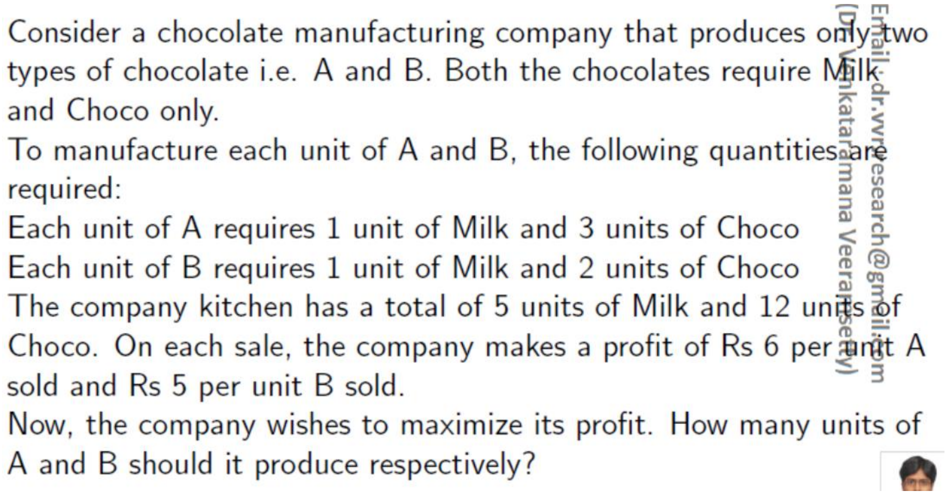
PRACTICAL OUTPUT:



```
PS C:\Users\rimsha\OneDrive\Desktop\Mohammed Farnas Ali Mudabbir\LAB 12\TASK> python -u "c:\
d Farnas Ali Mudabbir\tempCodeRunnerFile.python"
Enter numbers separated by spaces: 4 6 88 94 2
Sorted list: [2, 4, 6, 88, 94]
PS C:\Users\rimsha\OneDrive\Desktop\Mohammed Farnas Ali Mudabbir\LAB 12\TASK>
```

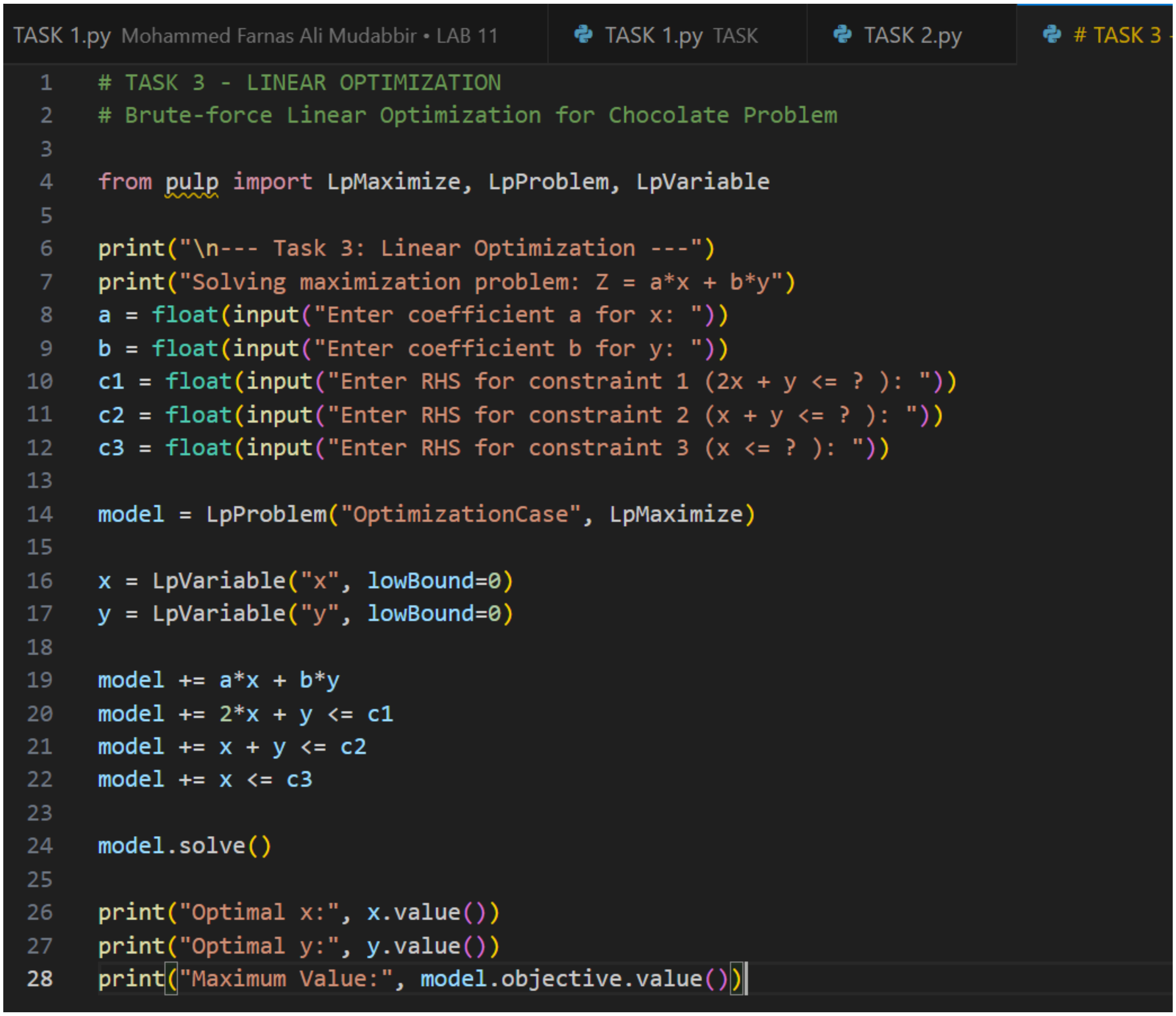
Task Description #3 – Optimization

Task: Write python code to solve below case study using linear optimization



Consider a chocolate manufacturing company that produces only two types of chocolate i.e. A and B. Both the chocolates require Milk and Choco only.
To manufacture each unit of A and B, the following quantities are required:
Each unit of A requires 1 unit of Milk and 3 units of Choco
Each unit of B requires 1 unit of Milk and 2 units of Choco
The company kitchen has a total of 5 units of Milk and 12 units of Choco. On each sale, the company makes a profit of Rs 6 per unit A sold and Rs 5 per unit B sold.
Now, the company wishes to maximize its profit. How many units of A and B should it produce respectively?

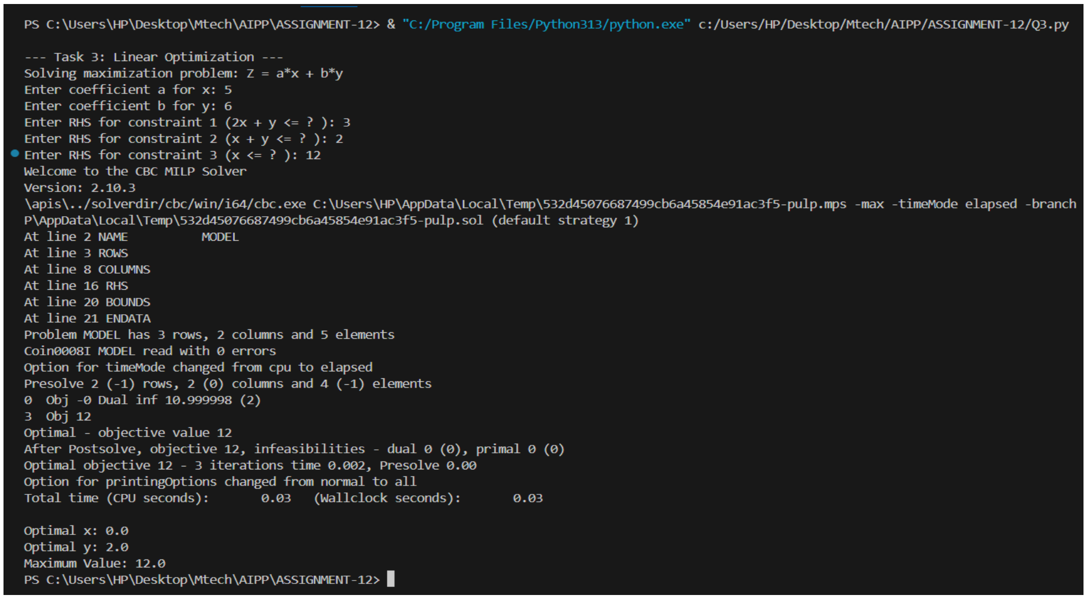
Email: dr.vvresearch@gmail.com
(Dr. Venkataramana Veeramsetty)



TASK 1.py Mohammed Farnas Ali Mudabbir • LAB 11
TASK 1.py TASK
TASK 2.py
# TASK 3 -

```python
# TASK 3 - LINEAR OPTIMIZATION
# Brute-force Linear Optimization for Chocolate Problem

from pulp import LpMaximize, LpProblem, LpVariable

print("\n--- Task 3: Linear Optimization ---")
print("Solving maximization problem: Z = a*x + b*y")
a = float(input("Enter coefficient a for x: "))
b = float(input("Enter coefficient b for y: "))
c1 = float(input("Enter RHS for constraint 1 (2x + y <= ? ): "))
c2 = float(input("Enter RHS for constraint 2 (x + y <= ? ): "))
c3 = float(input("Enter RHS for constraint 3 (x <= ? ): "))

model = LpProblem("OptimizationCase", LpMaximize)

x = LpVariable("x", lowBound=0)
y = LpVariable("y", lowBound=0)

model += a*x + b*y
model += 2*x + y <= c1
model += x + y <= c2
model += x <= c3

model.solve()

print("Optimal x:", x.value())
print("Optimal y:", y.value())
print("Maximum Value:", model.objective.value())
```

PRACTICAL OUTPUT:

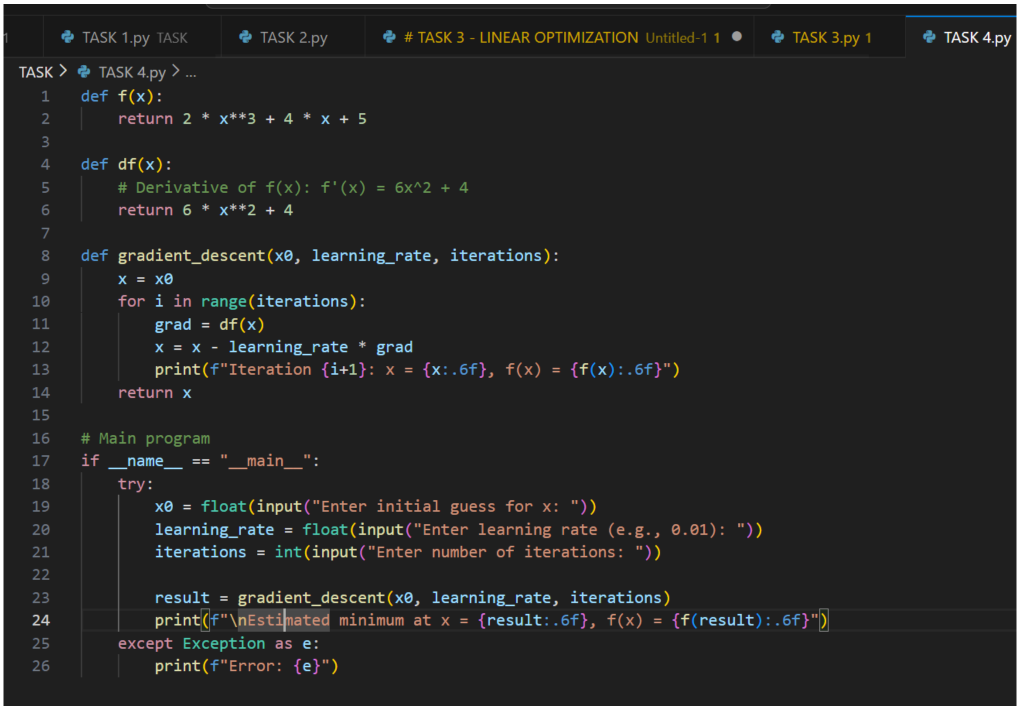


```
PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-12> & "C:/Program Files/Python313/python.exe" c:/Users/HP/Desktop/Mtech/AIPP/ASSIGNMENT-12/Q3.py

--- Task 3: Linear Optimization ---
Solving maximization problem: Z = a*x + b*y
Enter coefficient a for x: 5
Enter coefficient b for y: 6
Enter RHS for constraint 1 (2x + y <= ? ): 3
Enter RHS for constraint 2 (x + y <= ? ): 2
Enter RHS for constraint 3 (x <= ? ): 12
Welcome to the CBC MILP Solver
Version: 2.10.3
\apis\../solverdir/cbc/win/i64/cbc.exe C:\Users\HP\AppData\Local\Temp\532d45076687499cb6a45854e91ac3f5-pulp.mps -max -timeMode elapsed -branch
P\AppData\Local\Temp\532d45076687499cb6a45854e91ac3f5-pulp.sol (default strategy 1)
At line 2 NAME          MODEL
At line 3 ROWS
At line 8 COLUMNS
At line 16 RHS
At line 20 BOUNDS
At line 21 ENDATA
Problem MODEL has 3 rows, 2 columns and 5 elements
Coin0008I MODEL read with 0 errors
Option for timeMode changed from cpu to elapsed
Presolve 2 (-1) rows, 2 (0) columns and 4 (-1) elements
0  Obj -0 Dual inf 10.999998 (2)
3  Obj 12
Optimal - objective value 12
After Postsolve, objective 12, infeasibilities - dual 0 (0), primal 0 (0)
Optimal objective 12 - 3 iterations time 0.002, Presolve 0.00
Option for printingOptions changed from normal to all
Total time (CPU seconds):       0.03   (Wallclock seconds):       0.03

Optimal x: 0.0
Optimal y: 2.0
Maximum Value: 12.0
PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-12> 
```

Task Description #4 – Gradient Descent Optimization

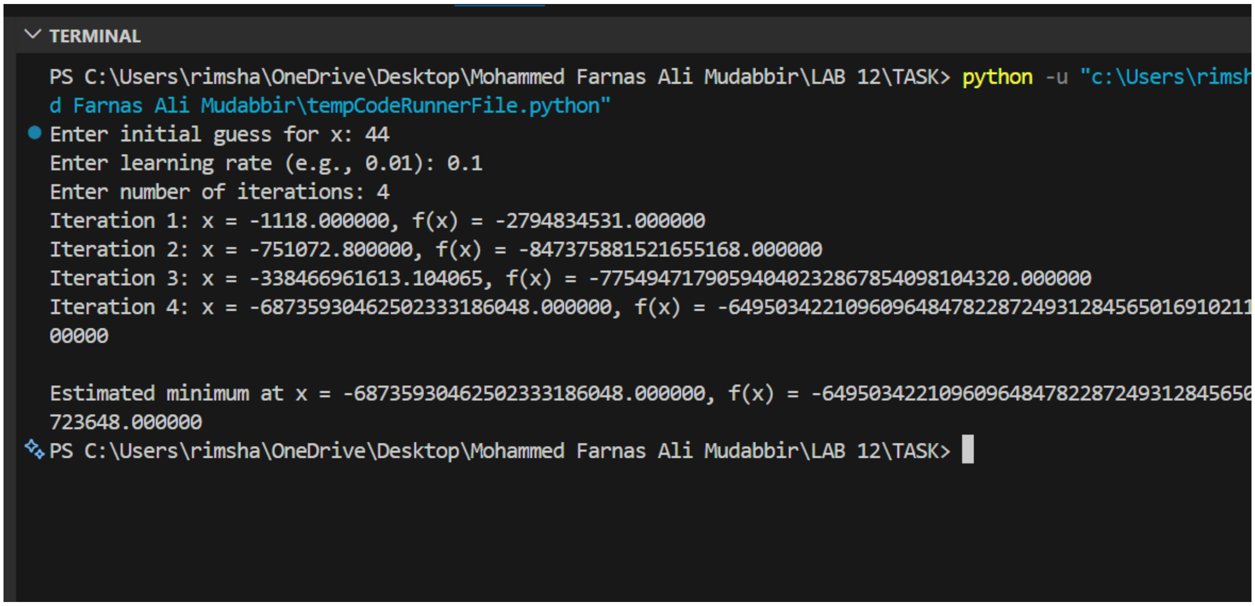
Task: Write python code to find value of x at which the function f(x)=$2X^3$+4x+5 will be minimum

PROMPT: "Write Python code to compute the derivative of f(x)=$2x^3$+4x+5, find the critical point, and print the value of x where the function is minimum."



```python
def f(x):
    return 2 * x**3 + 4 * x + 5

def df(x):
    # Derivative of f(x): f'(x) = 6x^2 + 4
    return 6 * x**2 + 4

def gradient_descent(x0, learning_rate, iterations):
    x = x0
    for i in range(iterations):
        grad = df(x)
        x = x - learning_rate * grad
        print(f"Iteration {i+1}: x = {x:.6f}, f(x) = {f(x):.6f}")
    return x

# Main program
if __name__ == "__main__":
    try:
        x0 = float(input("Enter initial guess for x: "))
        learning_rate = float(input("Enter learning rate (e.g., 0.01): "))
        iterations = int(input("Enter number of iterations: "))

        result = gradient_descent(x0, learning_rate, iterations)
        print(f"\nEstimated minimum at x = {result:.6f}, f(x) = {f(result):.6f}")
    except Exception as e:
        print(f"Error: {e}")
```

PRACTICAL OUTPUT:



```
PS C:\Users\rimsha\OneDrive\Desktop\Mohammed Farnas Ali Mudabbir\LAB 12\TASK> python -u "c:\Users\rimsh
d Farnas Ali Mudabbir\tempCodeRunnerFile.python"
Enter initial guess for x: 44
Enter learning rate (e.g., 0.01): 0.1
Enter number of iterations: 4
Iteration 1: x = -1118.000000, f(x) = -2794834531.000000
Iteration 2: x = -751072.800000, f(x) = -847375881521655168.000000
Iteration 3: x = -338466961613.104065, f(x) = -77549471790594040232867854098104320.000000
Iteration 4: x = -68735930462502333186048.000000, f(x) = -649503422109609648478228724931284565016910211
00000

Estimated minimum at x = -68735930462502333186048.000000, f(x) = -6495034221096096484782287249312845650
723648.000000
PS C:\Users\rimsha\OneDrive\Desktop\Mohammed Farnas Ali Mudabbir\LAB 12\TASK>
```