SCHOOLOFCOMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENTOFCOMPUTER SCIENCE ENGINEERING	
ProgramName:B. Tech		**AssignmentType: Lab**	**AcademicYear:**2025-2026
CourseCoordinatorName		Venkataramana Veeramsetty	
Instructor(s)Name		Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)	
CourseCode	24CS002PC215	**CourseTitle**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
DateandDay of Assignment	Week6 - WednesDay	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	
AssignmentNumber:12.3(Presentassignmentnumber)/**24**(Totalnumberofassignments)			

Q.No.	Question	*ExpectedTime 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.	Week5 - Monday

- To explore AI-assisted optimizations of existing algorithms.
- To compare naive vs. optimized approaches generated by AI.

Learning Outcomes

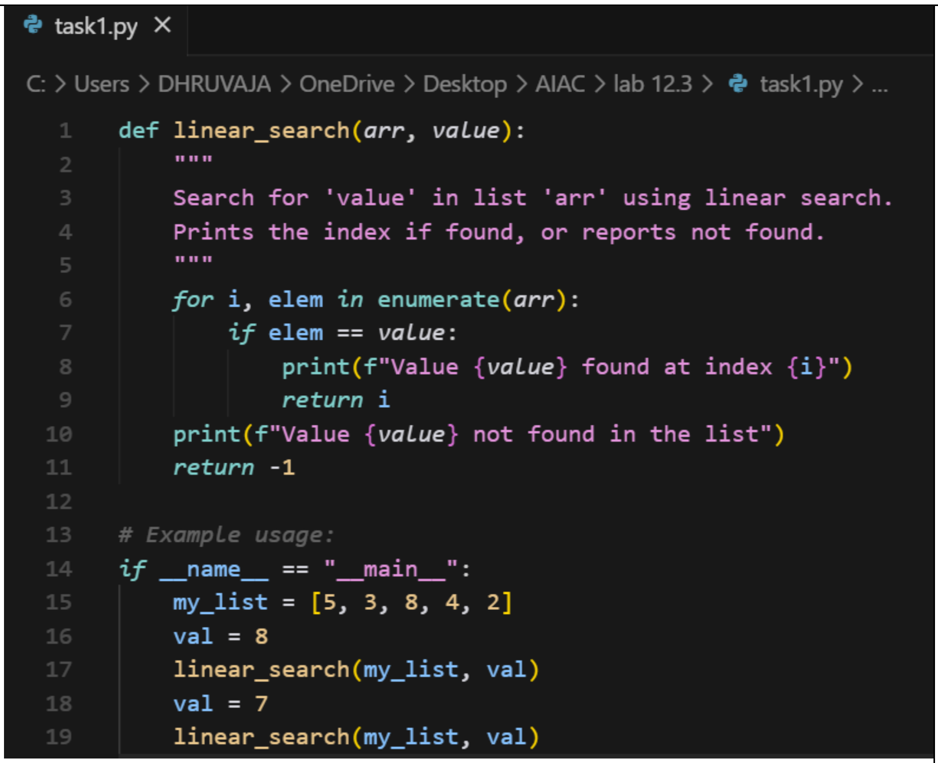
After completing this lab, students will be able to:

- 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 and extract it's index.

PROMPT : Write python code for linear_search() function to search a value in a list and extract it's index and display the output

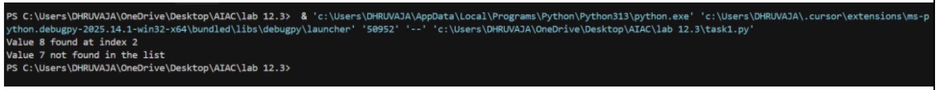


task1.py

C: > Users > DHRUVAJA > OneDrive > Desktop > AIAC > lab 12.3 > task1.py > ...

```python
def linear_search(arr, value):
    """
    Search for 'value' in list 'arr' using linear search.
    Prints the index if found, or reports not found.
    """
    for i, elem in enumerate(arr):
        if elem == value:
            print(f"Value {value} found at index {i}")
            return i
    print(f"Value {value} not found in the list")
    return -1

# Example usage:
if __name__ == "__main__":
    my_list = [5, 3, 8, 4, 2]
    val = 8
    linear_search(my_list, val)
    val = 7
    linear_search(my_list, val)
```

OUTPUT:

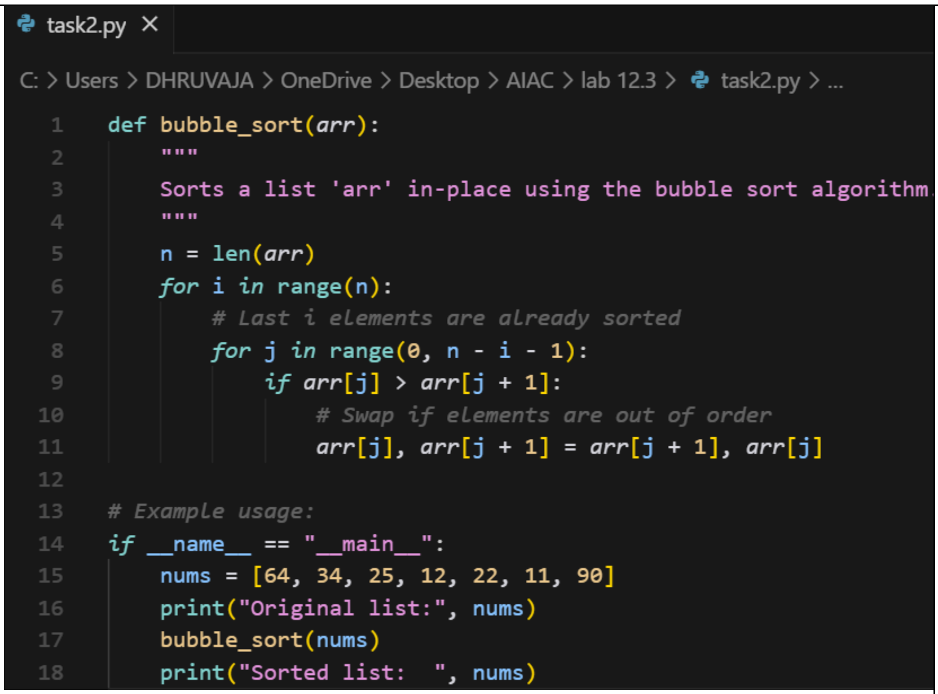


```
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>  & 'c:\Users\DHRUVAJA\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\DHRUVAJA\.cursor\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '50952' '--' 'c:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3\task1.py'
Value 8 found at index 2
Value 7 not found in the list
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>
```

Task Description #2 – Sorting Algorithms

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

PROMPT : generate a python code to implement Bubble Sort and check sorted output

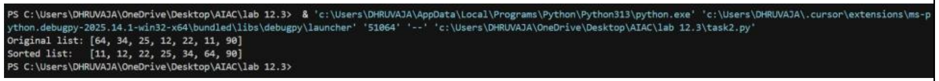


task2.py ×

C: > Users > DHRUVAJA > OneDrive > Desktop > AIAC > lab 12.3 > task2.py > ...

```python
def bubble_sort(arr):
    """
    Sorts a list 'arr' in-place using the bubble sort algorithm
    """
    n = len(arr)
    for i in range(n):
        # Last i elements are already sorted
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                # Swap if elements are out of order
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

# Example usage:
if __name__ == "__main__":
    nums = [64, 34, 25, 12, 22, 11, 90]
    print("Original list:", nums)
    bubble_sort(nums)
    print("Sorted list:  ", nums)
```

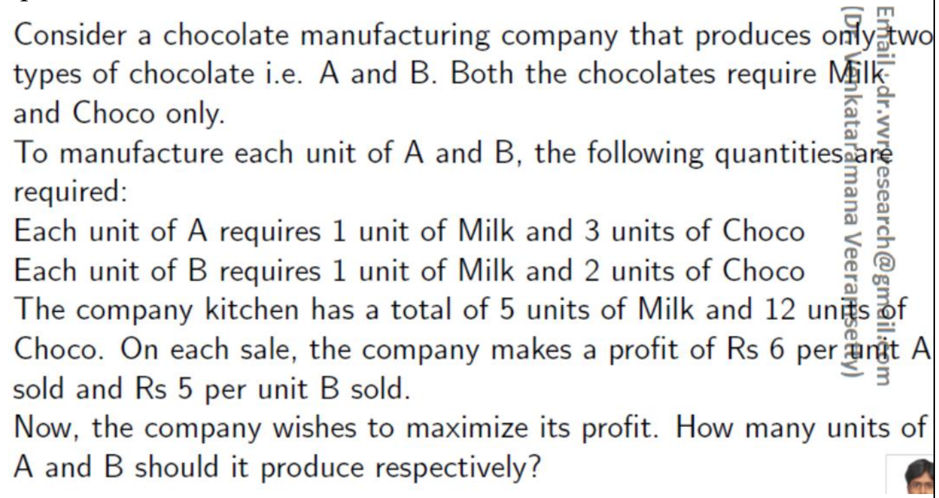
OUTPUT :



```
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>  & 'c:\Users\DHRUVAJA\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\DHRUVAJA\.cursor\extensions\ms-p
ython.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '51064' '--' 'c:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3\task2.py'
Original list: [64, 34, 25, 12, 22, 11, 90]
Sorted list:   [11, 12, 22, 25, 34, 64, 90]
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>
```

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?

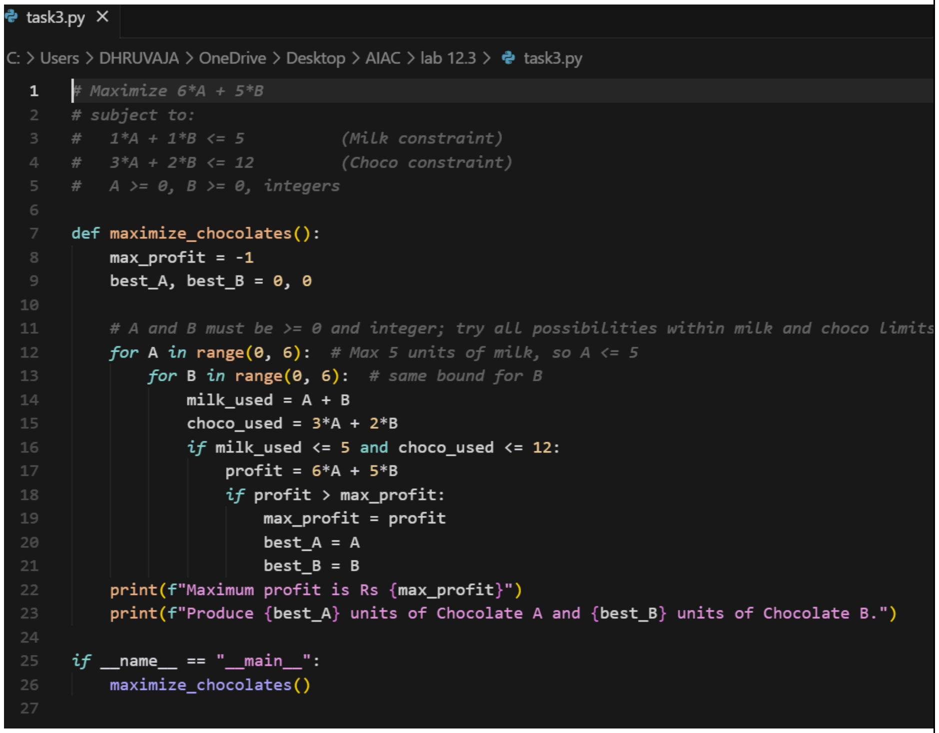
PROMPT : generate a python code for the given scenario

A chocolate company produces two types of chocolates, A and B. Both require two raw materials: Milk and Choco.

- **Each unit of chocolate A needs 1 unit of Milk and 3 units of Choco.**

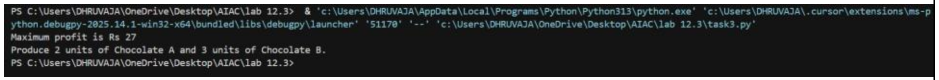
- **Each unit of chocolate B needs 1 unit of Milk and 2 units of Choco.**

The company has a total of 5 units of Milk and 12 units of Choco available.
Each unit of chocolate A generates a profit of Rs 6, and each unit of chocolate B generates a profit of Rs 5.
To maximize profit, how many units of chocolates A and B should the company produce, given the resource constraints?
Display the output



```
task3.py
C: > Users > DHRUVAJA > OneDrive > Desktop > AIAC > lab 12.3 > task3.py
# Maximize 6*A + 5*B
# subject to:
#   1*A + 1*B <= 5          (Milk constraint)
#   3*A + 2*B <= 12         (Choco constraint)
#   A >= 0, B >= 0, integers

def maximize_chocolates():
    max_profit = -1
    best_A, best_B = 0, 0

    # A and B must be >= 0 and integer; try all possibilities within milk and choco limits
    for A in range(0, 6):  # Max 5 units of milk, so A <= 5
        for B in range(0, 6):  # same bound for B
            milk_used = A + B
            choco_used = 3*A + 2*B
            if milk_used <= 5 and choco_used <= 12:
                profit = 6*A + 5*B
                if profit > max_profit:
                    max_profit = profit
                    best_A = A
                    best_B = B
    print(f"Maximum profit is Rs {max_profit}")
    print(f"Produce {best_A} units of Chocolate A and {best_B} units of Chocolate B.")

if __name__ == "__main__":
    maximize_chocolates()
```

OUTPUT :

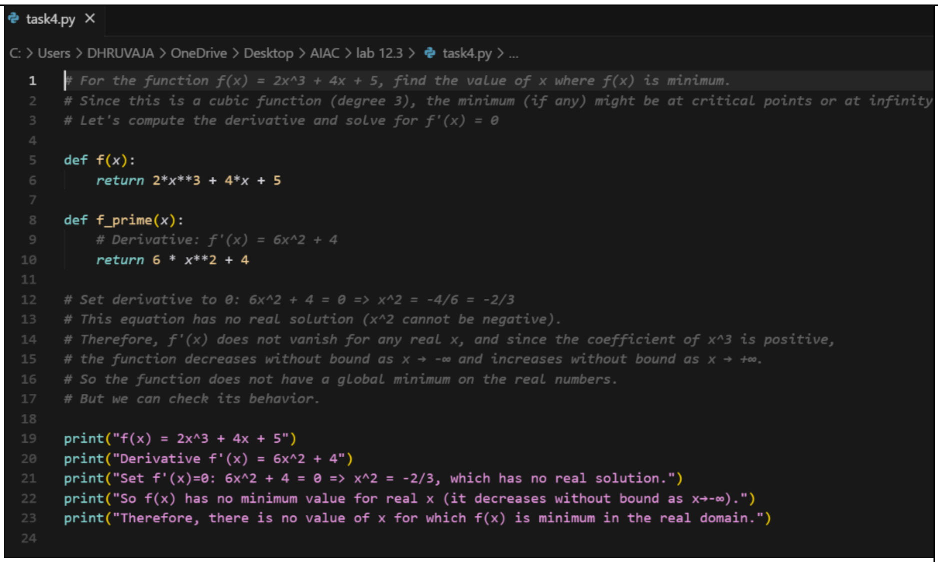


```
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>  & 'c:\Users\DHRUVAJA\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\DHRUVAJA\.cursor\extensions\ms-p
ython.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '51170' '--' 'c:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3\task3.py'
Maximum profit is Rs 27
Produce 2 units of Chocolate A and 3 units of Chocolate B.
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>
```

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

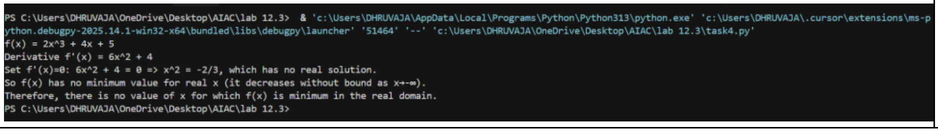
PROMPT : Write python code to find value of x at which the function $f(x)=2X^3+4x+5$ will be minimum and display the output


task4.py ×

C: > Users > DHRUVAJA > OneDrive > Desktop > AIAC > lab 12.3 > task4.py > ...

```python
# For the function f(x) = 2x^3 + 4x + 5, find the value of x where f(x) is minimum.
# Since this is a cubic function (degree 3), the minimum (if any) might be at critical points or at infinity
# Let's compute the derivative and solve for f'(x) = 0

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

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

# Set derivative to 0: 6x^2 + 4 = 0 => x^2 = -4/6 = -2/3
# This equation has no real solution (x^2 cannot be negative).
# Therefore, f'(x) does not vanish for any real x, and since the coefficient of x^3 is positive,
# the function decreases without bound as x → -∞ and increases without bound as x → +∞.
# So the function does not have a global minimum on the real numbers.
# But we can check its behavior.

print("f(x) = 2x^3 + 4x + 5")
print("Derivative f'(x) = 6x^2 + 4")
print("Set f'(x)=0: 6x^2 + 4 = 0 => x^2 = -2/3, which has no real solution.")
print("So f(x) has no minimum value for real x (it decreases without bound as x→-∞).")
print("Therefore, there is no value of x for which f(x) is minimum in the real domain.")
```

OUTPUT :



```
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>  & 'c:\Users\DHRUVAJA\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\DHRUVAJA\.cursor\extensions\ms-p
ython.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '51464' '--' 'c:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3\task4.py'
f(x) = 2x^3 + 4x + 5
Derivative f'(x) = 6x^2 + 4
Set f'(x)=0: 6x^2 + 4 = 0 => x^2 = -2/3, which has no real solution.
So f(x) has no minimum value for real x (it decreases without bound as x→-∞).
Therefore, there is no value of x for which f(x) is minimum in the real domain.
PS C:\Users\DHRUVAJA\OneDrive\Desktop\AIAC\lab 12.3>
```