4.2 ASSEMBLY LINE SCHEDULING USING DYNAMIC PROGRAMMING

Question:

In a factory, there are two assembly lines, each with *n* stations. Each station performs a specific task and takes a certain amount of time to complete. The task must go through each station in order, and there is also a transfer time for switching from one line to another. Given the time taken at each station on both lines and the transfer time between the lines, the goal is to find the minimum time required to process a product from start to end.

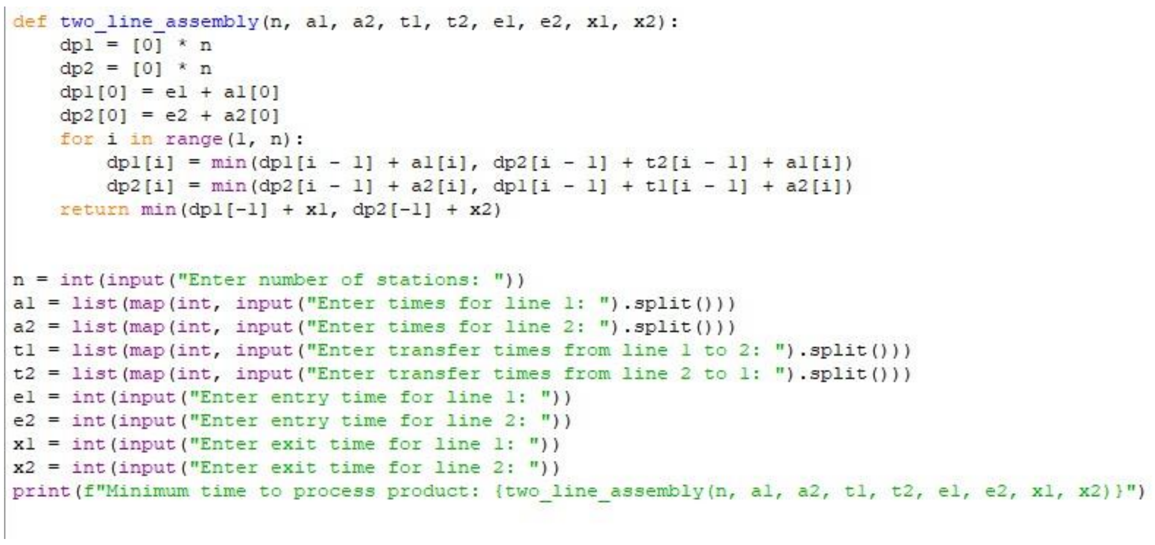
AIM

To implement the Assembly Line Scheduling algorithm in Python and compute the minimum time required to process a product through two assembly lines.

ALGORITHM

1. Let and be the processing times at station i on line 1 and line 2 respectively.
2. Let and be the transfer times from line 1 to line 2 and vice versa after station i.
3. Let and be the entry times for line 1 and line 2.
4. Let and be the exit times for line 1 and line 2.
5. Use dynamic programming to compute the minimum time to reach each station on both lines.
6. Return the minimum of the final station times plus exit times.

PROGRAM



```python
def two_line_assembly(n, a1, a2, t1, t2, e1, e2, x1, x2):
    dp1 = [0] * n
    dp2 = [0] * n
    dp1[0] = e1 + a1[0]
    dp2[0] = e2 + a2[0]
    for i in range(1, n):
        dp1[i] = min(dp1[i - 1] + a1[i], dp2[i - 1] + t2[i - 1] + a1[i])
        dp2[i] = min(dp2[i - 1] + a2[i], dp1[i - 1] + t1[i - 1] + a2[i])
    return min(dp1[-1] + x1, dp2[-1] + x2)


n = int(input("Enter number of stations: "))
a1 = list(map(int, input("Enter times for line 1: ").split()))
a2 = list(map(int, input("Enter times for line 2: ").split()))
t1 = list(map(int, input("Enter transfer times from line 1 to 2: ").split()))
t2 = list(map(int, input("Enter transfer times from line 2 to 1: ").split()))
e1 = int(input("Enter entry time for line 1: "))
e2 = int(input("Enter entry time for line 2: "))
x1 = int(input("Enter exit time for line 1: "))
x2 = int(input("Enter exit time for line 2: "))
print(f"Minimum time to process product: {two_line_assembly(n, a1, a2, t1, t2, e1, e2, x1, x2)}")
```

Input:

Enter number of stations: 3

Enter processing times for Line 1: 4 5 3

Enter processing times for Line 2: 2 10 1

Enter transfer times from Line 1 to Line 2: 7 4

Enter transfer times from Line 2 to Line 1: 9 2

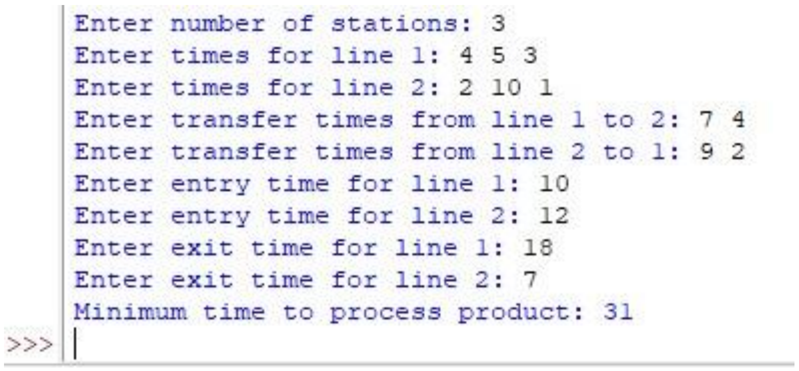
Enter entry time for Line 1: 10

Enter entry time for Line 2: 12

Enter exit time for Line 1: 18

Enter exit time for Line 2: 7

Output:



```
Enter number of stations: 3
Enter times for line 1: 4 5 3
Enter times for line 2: 2 10 1
Enter transfer times from line 1 to 2: 7 4
Enter transfer times from line 2 to 1: 9 2
Enter entry time for line 1: 10
Enter entry time for line 2: 12
Enter exit time for line 1: 18
Enter exit time for line 2: 7
Minimum time to process product: 31
>>>
```

RESULT:

Thus the program is successfully executed and the output is verified.

PERFORMANCE ANALYSIS:

- Time Complexity:
  - O(n)
- Space complexity:
  - O(n)