**Name:** Priya Lakhotiya

**Enrollment No.:** 180089

**PROGRAM-13**

**Aim:** Write an algorithm and program to implement Assembly-Line Scheduling.

**Algorithm:**

1. f1[1] = e1 + a1,1

2. f2[1] = e2 + a2,1

3. for j = 2 to n

4. if ((f1[j − 1] + a1,j ) ≤ (f2[j − 1] + t2,j−1 + a1,j )) then

5. f1[j] = f1[j − 1] + a1,j and l1[j] = 1 /\* lp denotes the line p \*/

6. else

7. f1[j] = f2[j − 1] + t2,j−1 + a1,j and l1[j] = 2

8. if ((f2[j − 1] + a2,j ) ≤ (f1[j − 1] + t1,j−1 + a2,j )) then

9. f2[j] = f2[j − 1] + a2,j and l2[j] = 2

10. else

11. f2[j] = f1[j − 1] + t1,j−1 + a2,j and l2[j] = 1

12. end for

13. if (f1[n] + x1 ≤ f2[n] + x2) then

14. f OP T = f1[n] + x1 and l OP T = 1

15. else

16. f OP T = f2[n] + x2 and l OP T = 2

**Source Code:**

#include <stdio.h>

#include<conio.h>

#define NUM\_LINE 2

#define NUM\_STATION 4

// Utility function to find minimum of two numbers

int min(int a, int b) { return a < b ? a : b; }

int carAssembly(int a[][NUM\_STATION], int t[][NUM\_STATION], int \*e, int \*x)

{

int T1[NUM\_STATION], T2[NUM\_STATION], i;

T1[0] = e[0] + a[0][0]; // time taken to leave first station in line 1

T2[0] = e[1] + a[1][0]; // time taken to leave first station in line 2

// Fill tables T1[] and T2[] using the above given recursive relations

for (i = 1; i < NUM\_STATION; ++i)

{

T1[i] = min(T1[i-1] + a[0][i], T2[i-1] + t[1][i] + a[0][i]);

T2[i] = min(T2[i-1] + a[1][i], T1[i-1] + t[0][i] + a[1][i]);

}

// Consider exit times and retutn minimum

return min(T1[NUM\_STATION-1] + x[0], T2[NUM\_STATION-1] + x[1]);

}

void main()

{

clrscr();

int a[][NUM\_STATION] = {{4, 5, 3, 2},

{2, 10, 1, 4}};

int t[][NUM\_STATION] = {{0, 7, 4, 5},

{0, 9, 2, 8}};

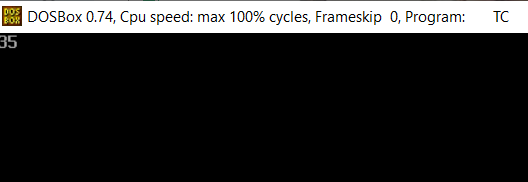
int e[] = {10, 12}, x[] = {18, 7};

printf("%d", carAssembly(a, t, e, x));

getch();

}

**Output:**



**Complexity:**

O(n)