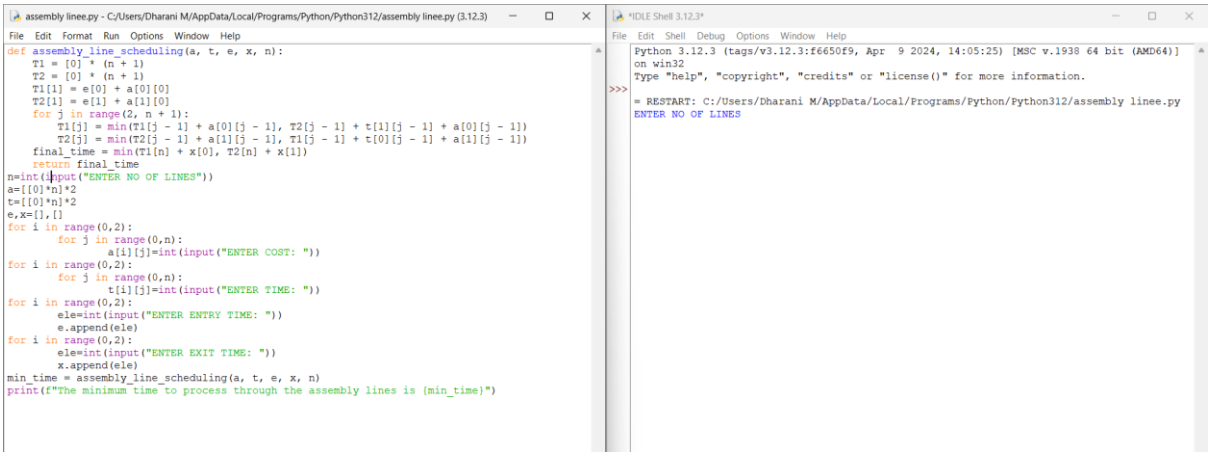
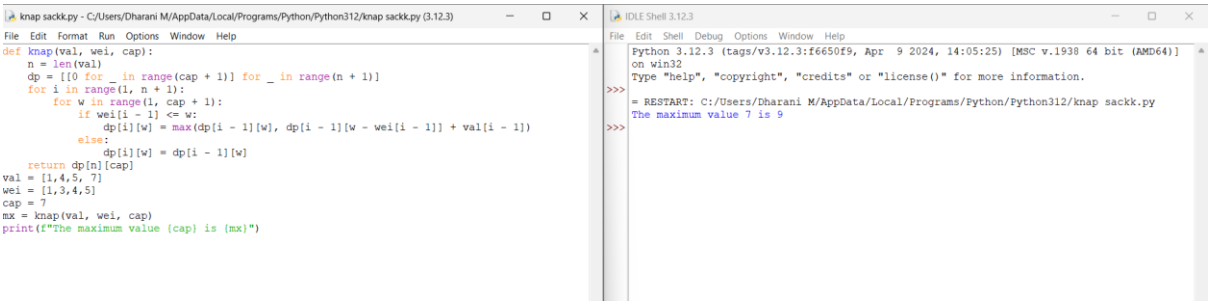
1. Assembly Line Scheduling



```
def assembly_line_scheduling(a, t, e, x, n):
    T1 = [0] * (n + 1)
    T2 = [0] * (n + 1)
    T1[1] = e[0] + a[0][0]
    T2[1] = e[1] + a[1][0]
    for j in range(2, n + 1):
        T1[j] = min(T1[j - 1] + a[0][j - 1], T2[j - 1] + t[1][j - 1] + a[0][j - 1])
        T2[j] = min(T2[j - 1] + a[1][j - 1], T1[j - 1] + t[0][j - 1] + a[1][j - 1])
    final_time = min(T1[n] + x[0], T2[n] + x[1])
    return final_time
n=int(input("ENTER NO OF LINES"))
a=[[0]*n]*2
t=[[0]*n]*2
e,x=[],[]
for i in range(0,2):
        for j in range(0,n):
                a[i][j]=int(input("ENTER COST: "))
for i in range(0,2):
        for j in range(0,n):
                t[i][j]=int(input("ENTER TIME: "))
for i in range(0,2):
        ele=int(input("ENTER ENTRY TIME: "))
        e.append(ele)
for i in range(0,2):
        ele=int(input("ENTER EXIT TIME: "))
        x.append(ele)
min_time = assembly_line_scheduling(a, t, e, x, n)
print(f"The minimum time to process through the assembly lines is {min_time}")
```

```
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr  9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Dharani M/AppData/Local/Programs/Python/Python312/assembly linee.py
ENTER NO OF LINES
```

2. Knapsack problem and Memory



```
def knap(val, wei, cap):
    n = len(val)
    dp = [[0 for _ in range(cap + 1)] for _ in range(n + 1)]
    for i in range(1, n + 1):
        for w in range(1, cap + 1):
            if wei[i - 1] <= w:
                dp[i][w] = max(dp[i - 1][w], dp[i - 1][w - wei[i - 1]] + val[i - 1])
            else:
                dp[i][w] = dp[i - 1][w]
    return dp[n][cap]
val = [1,4,5, 7]
wei = [1,3,4,5]
cap = 7
mx = knap(val, wei, cap)
print(f"The maximum value {cap} is {mx}")
```

```
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr  9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Dharani M/AppData/Local/Programs/Python/Python312/knap sackk.py
The maximum value 7 is 9
>>>
```

3. Bellman-Ford Algorithm

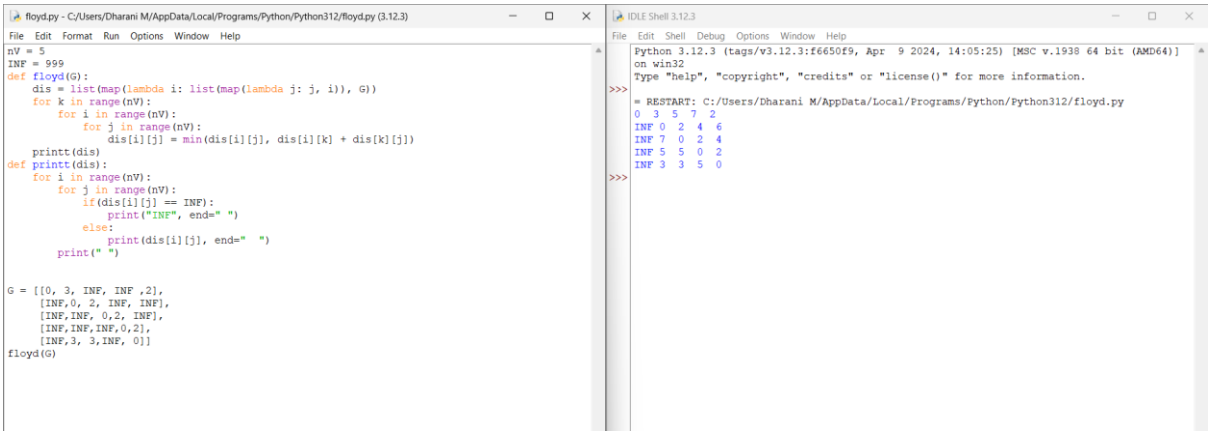
BELLMEN.py - C:\Users\Dharani M\AppData\Local\Programs\Python\Python312\BELLMEN.py (3.12.3)

File Edit Format Run Options Window Help

```
def BellmanFord(graph, vertices, source):
    distance = [float("Inf")] * vertices
    distance[source] = 0
    for _ in range(vertices - 1):
        for u, v, w in graph:
            if distance[u] != float("Inf") and distance[u] + w < distance[v]:
                distance[v] = distance[u] + w
    for u, v, w in graph:
        if distance[u] != float("Inf") and distance[u] + w < distance[v]:
            print("Graph contains a negative weight cycle")
            return
    return distance
vertices = int(input("Enter the number of vertices: "))
edges = int(input("Enter the number of edges: "))

print("Enter edges in the format: u v w (where u and v are vertices, w is the weight):")
graph = []
for _ in range(edges):
    u, v, w = map(int, input().split())
    graph.append((u, v, w))
source = int(input("Enter the source vertex: "))
distances = BellmanFord(graph, vertices, source)
if distances:
    print("Vertex Distance from Source")
    for i in range(vertices):
        print(f"{i}\t\t{distances[i]}")
```

4. Warshall’s & Floyd’s Algorithm



floyd.py - C:/Users/Dharani M/AppData/Local/Programs/Python/Python312/floyd.py (3.12.3)

File Edit Format Run Options Window Help

```
nV = 5
INF = 999
def floyd(G):
    dis = list(map(lambda i: list(map(lambda j: j, i)), G))
    for k in range(nV):
        for i in range(nV):
            for j in range(nV):
                dis[i][j] = min(dis[i][j], dis[i][k] + dis[k][j])
    printt(dis)
def printt(dis):
    for i in range(nV):
        for j in range(nV):
            if(dis[i][j] == INF):
                print("INF", end=" ")
            else:
                print(dis[i][j], end="  ")
        print(" ")


G = [[0, 3, INF, INF ,2],
     [INF,0, 2, INF, INF],
     [INF,INF, 0,2, INF],
     [INF,INF,INF,0,2],
     [INF,3, 3,INF, 0]]
floyd(G)
```

IDLE Shell 3.12.3

File Edit Shell Debug Options Window Help

```
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr  9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Dharani M/AppData/Local/Programs/Python/Python312/floyd.py
0  3  5  7  2
INF 0  2  4  6
INF 7  0  2  4
INF 5  5  0  2
INF 3  3  5  0
>>>
```