Task 2

Implementation of **Hill climbing algorithm for Heuristic search** approach using following constraints in python.

Aim: To Implement Hill climbing algorithm for Heuristic search approach for travelling salesman problem using python

Algorithm:

Step 1: start

Step 2: define TSP with (graph, s) and assign value for vertex.

Step 3: store all vertex apart from source vertex.

Step 4: store minimum weight hamiltonian cycle and assign permutation (vertex).

Step 5: store current path weight (cost) and compute current path weight.

Step 6: Update minimum and matrix representation of the graph values and print it. **Step 7:** stop

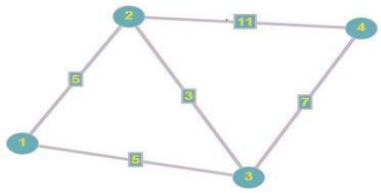
Program:

```
from sys import maxsize from
itertools import permutations
V = 4  # Number of vertices
def travellingSalesmanProblem(graph, s):
    # Create a list of vertices excluding the starting point
vertex = [i for i in range(V) if i != s]
    min_path = maxsize  # Initialize minimum path to max integer
    next_permutation = permutations(vertex)  # Generate all permutations of vertices
    for perm in next_permutation:
        current_pathweight = 0
        k = s  # Start from the source vertex
        # Calculate the path weight for the current permutation
for j in perm:
            current_pathweight += graph[k][j]
            k = j
        current_pathweight += graph[k][s]  # Add cost to return to start
    # Update minimum path if current path is better          min_path = min(min_path,
current_pathweight)
```

```
    return min_path
if __name__ == "__main__":
    graph = [
        [0, 10, 15, 20],
        [10, 0, 35, 25],
        [15, 35, 0, 30],
        [20, 25, 30, 0]
    ]
 s = 0
# Starting vertex
    print(travellingSalesmanProblem(graph, s))
```

Output:

```
========== RESTART: C:/Users/mahes/VTU26520.py ==
Minimum cost of TSP path: 80
>>>
```



Result:

Thus the Implementation of Hill climbing algorithm for Heuristic search approach for travelling salesman problem using python was successfully executed and output was verified.