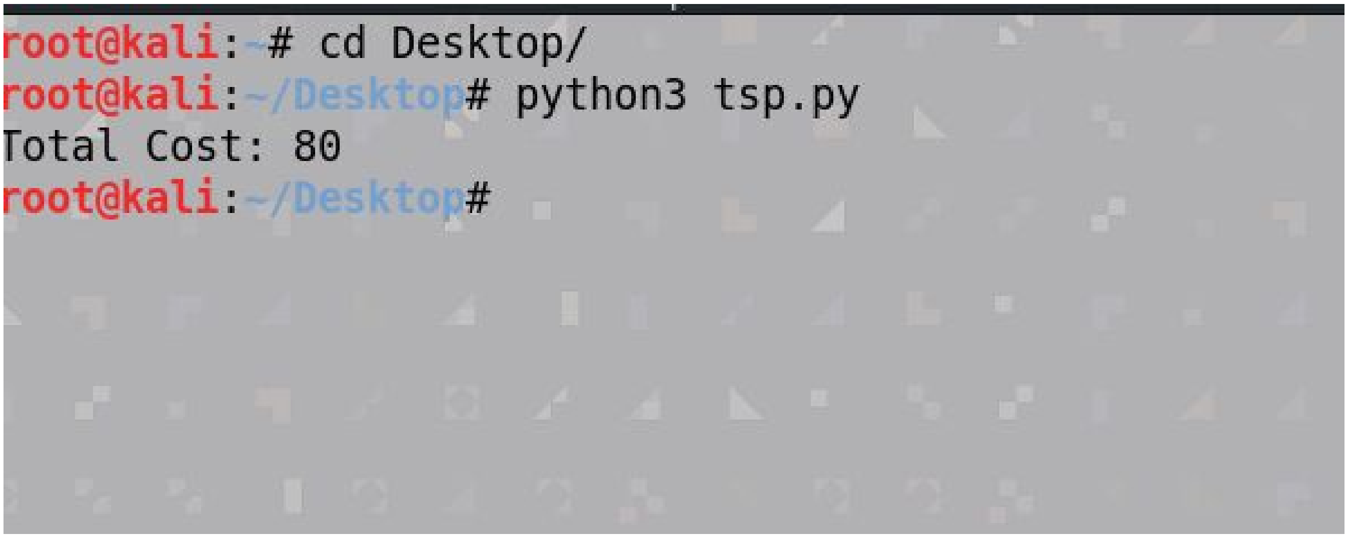
Travelling Salesman Problem

```
from sys import maxsize
V = 4

def travellingSalesmanProblem(graph, s):

    vertex = []
    for i in range(V):
        if i != s:
            vertex.append(i)

    min_path = maxsize

    while True:
        current_pathweight = 0

        k = s
        for i in range(len(vertex)):
            current_pathweight += graph[k][vertex[i]]
            k = vertex[i]
        current_pathweight += graph[k][s]

        min_path = min(min_path, current_pathweight)

        if not next_permutation(vertex):
            break

    return min_path

def next_permutation(L):

    n = len(L)

    i = n - 2
    while i >= 0 and L[i] >= L[i + 1]:
        i -= 1

    if i == -1:
        return False

    j = i + 1
```

```
        while j < n and L[j] > L[i]:
            j += 1
        j -= 1

        L[i], L[j] = L[j], L[i]

        left = i + 1
        right = n - 1

        while left < right:
            L[left], L[right] = L[right], L[left]
            left += 1
            right -= 1

        return True


    if __name__ == "__main__":

        graph = [[0, 10, 15, 20], [10, 0, 35, 25],
                 [15, 35, 0, 30], [20, 25, 30, 0]]
        s = 0
        print("Total Cost:",travellingSalesmanProblem(graph, s))
```



```
root@kali:~# cd Desktop/
root@kali:~/Desktop# python3 tsp.py
Total Cost: 80
root@kali:~/Desktop#
```