Practical no:03

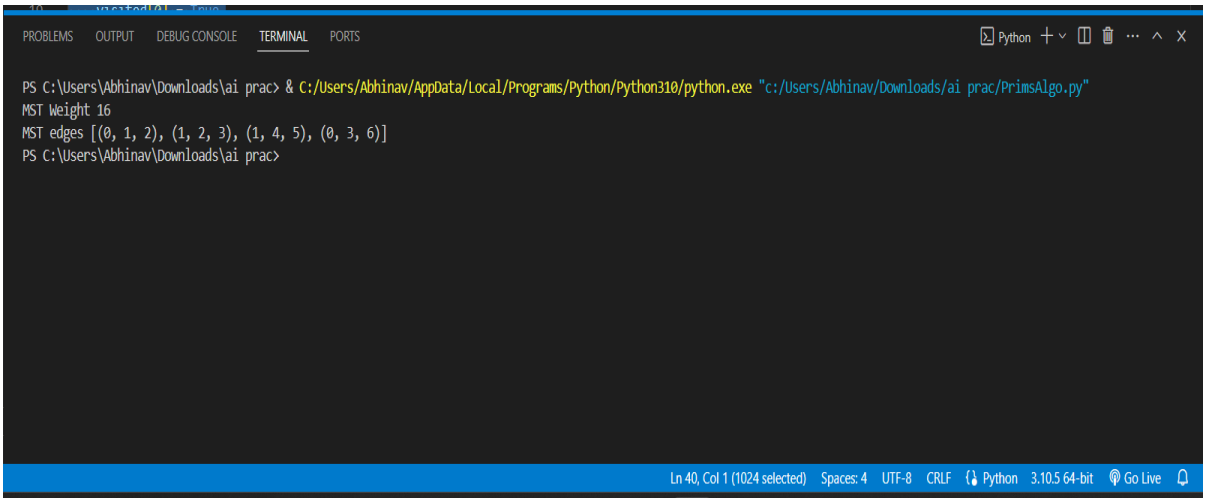
Greedy search algorithm: Prim's Minimal Spanning Tree Algorithm

Program:

```
def prim(adjacency_list):
    n = len(adjacency_list)
    visited = [False] * n
    mst_weight = 0
    mst_edges = []
    def find_min_edge():
        min_weight = 9999
        min_edge = None
        for i in range(n):
            if visited[i]:
                for j, weight in adjacency_list[i].items():
                    if not visited[j] and weight < min_weight:
                        min_weight = weight
                        min_edge = (i, j, weight)
        return min_edge
    visited[0] = True
    for i in range(n - 1):
        min_edge = find_min_edge()
        if min_edge is None:
            break
        mst_edges.append(min_edge)
        mst_weight += min_edge[2]
        visited[min_edge[1]] = True
```

```
    print("MST Weight",mst_weight)
    print("MST edges", mst_edges)
adjacency_list = [
    {1: 2, 3: 6},
    {0: 2, 2: 3, 3: 8, 4: 5},
    {1: 3, 4: 7},
    {0: 6, 1: 8, 4: 9},
    {1: 5, 2: 7, 3: 9}
]
if __name__ == '__main__' :
    prim(adjacency_list)
```

Output:



;