Nirbhay Sharma (B19CSE114)

Optmization for machine learning

Que-1

Code

```
import sys

choice = int(sys.argv[1])

if choice == 1:
    graph = [
        [0, 2, 2, 0, 3, 0, 0],
        [2, 0, 2, 0, 0, 4, 0],
        [2, 2, 0, 3, 0, 0, 0],
        [0, 0, 3, 0, 3, 6, 0],
        [3, 0, 0, 3, 0, 5, 3],
        [0, 4, 0, 6, 5, 0, 3],
        [0, 0, 0, 0, 3, 3, 0]
    ]
else:

    graph = [
        [0, 9, 8, 0, 0, 0],
        [9, 0, 2, 4, 4, 0],
        [8, 2, 0, 0, 5, 3],
        [0, 4, 0, 0, 0, 5],
        [0, 4, 5, 0, 0, 6],
        [0, 0, 3, 5, 6, 0],
    ]

degree_list = []
for idx, nodes in enumerate(graph):
    degr = 0
    for node in nodes:
        degr += node
    degree_list.append(degr)

print(degree_list)

def out_edges(s):
    global graph, degree_list;
    final_out_edges = 0
    for degree in s:
        contained = 0;
        for idx, nodes in enumerate(graph[degree]):
            if nodes != 0 and idx in s:
                contained += nodes
```

```
            final_out_edges += (degree_list[degree] - contained)
        return final_out_edges

    n_v = len(graph)
    s = []
    visited = [0 for _ in range(n_v)]
    global_max = -1
    while (True):
        max_vertex = -1;
        max_value = -1
        for i in range(n_v):
            if not visited[i]:
                new_s = s + [i]
                oe = out_edges(new_s)
                if oe > max_value:
                    max_value = oe;
                    max_vertex = i
        if max_value < global_max:
            break
        global_max = max_value
        visited[max_vertex] = 1
        s.append(max_vertex)
        # print(max_vertex, max_value, global_max)
        # print(s)
        if all([v == 1 for v in visited]):
            break;
    print(s)
    print(global_max)

    """
    for que-1
    s = [5, 0, 2, 6]
    max_cut = 28

    for que-2
    s = [1, 2, 5]
    max_cut = 41
    """
```