Central university of Haryana

Department of computer science & engineering under SOET

Algorithms lab

(BT CS 505a)

Lab-6.

Submitted by :-

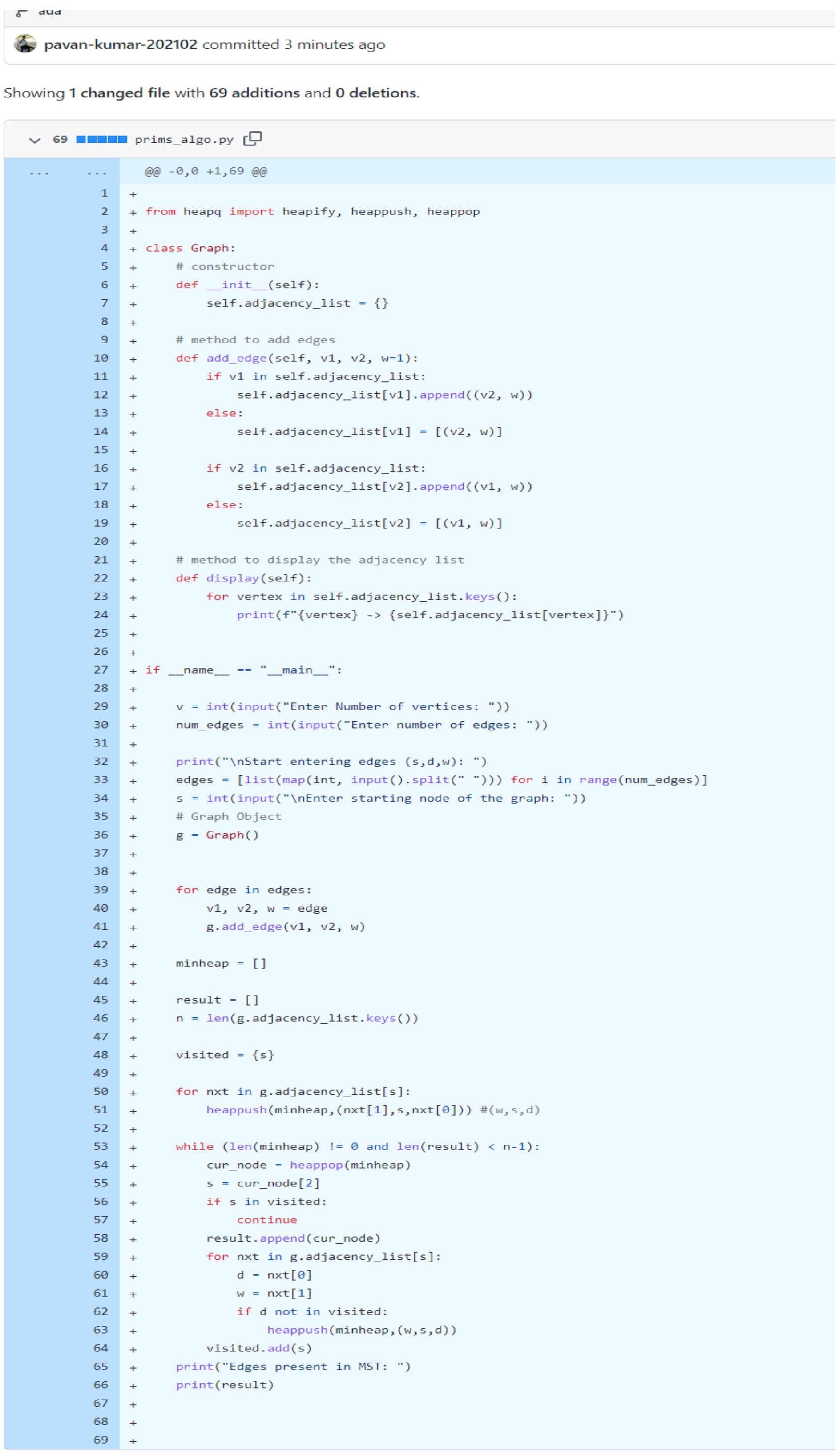
Ponnaganti pavan kumar

ROLL NO: 202102

submitted to :-

anant rajee bara

Problem statement:
WAP to represent Graph using Adjaceny list



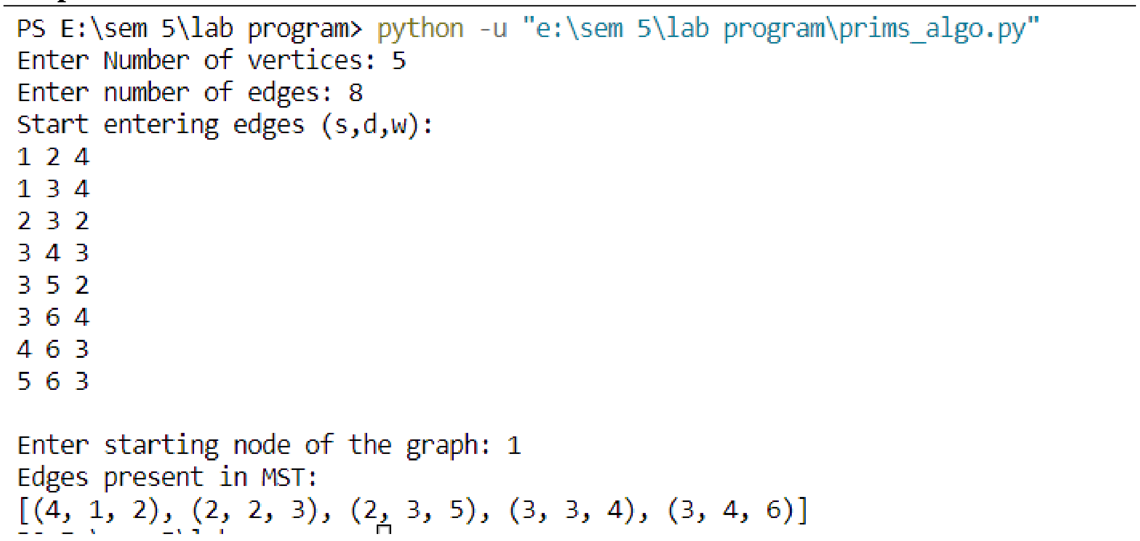
pavan-kumar-202102 committed 3 minutes ago

Showing **1 changed file** with **69 additions** and **0 deletions**.

69 prims_algo.py

```diff
@@ -0,0 +1,69 @@
+
+ from heapq import heapify, heappush, heappop
+
+ class Graph:
+     # constructor
+     def __init__(self):
+         self.adjacency_list = {}
+
+     # method to add edges
+     def add_edge(self, v1, v2, w=1):
+         if v1 in self.adjacency_list:
+             self.adjacency_list[v1].append((v2, w))
+         else:
+             self.adjacency_list[v1] = [(v2, w)]
+
+         if v2 in self.adjacency_list:
+             self.adjacency_list[v2].append((v1, w))
+         else:
+             self.adjacency_list[v2] = [(v1, w)]
+
+     # method to display the adjacency list
+     def display(self):
+         for vertex in self.adjacency_list.keys():
+             print(f"{vertex} -> {self.adjacency_list[vertex]}")
+
+
+ if __name__ == "__main__":
+
+     v = int(input("Enter Number of vertices: "))
+     num_edges = int(input("Enter number of edges: "))
+
+     print("\nStart entering edges (s,d,w): ")
+     edges = [list(map(int, input().split(" "))) for i in range(num_edges)]
+     s = int(input("\nEnter starting node of the graph: "))
+     # Graph Object
+     g = Graph()
+
+
+     for edge in edges:
+         v1, v2, w = edge
+         g.add_edge(v1, v2, w)
+
+     minheap = []
+
+     result = []
+     n = len(g.adjacency_list.keys())
+
+     visited = {s}
+
+     for nxt in g.adjacency_list[s]:
+         heappush(minheap,(nxt[1],s,nxt[0])) #(w,s,d)
+
+     while (len(minheap) != 0 and len(result) < n-1):
+         cur_node = heappop(minheap)
+         s = cur_node[2]
+         if s in visited:
+             continue
+         result.append(cur_node)
+         for nxt in g.adjacency_list[s]:
+             d = nxt[0]
+             w = nxt[1]
+             if d not in visited:
+                 heappush(minheap,(w,s,d))
+         visited.add(s)
+     print("Edges present in MST: ")
+     print(result)
+
+
+
```

Output:



```
PS E:\sem 5\lab program> python -u "e:\sem 5\lab program\prims_algo.py"
Enter Number of vertices: 5
Enter number of edges: 8
Start entering edges (s,d,w):
1 2 4
1 3 4
2 3 2
3 4 3
3 5 2
3 6 4
4 6 3
5 6 3

Enter starting node of the graph: 1
Edges present in MST:
[(4, 1, 2), (2, 2, 3), (2, 3, 5), (3, 3, 4), (3, 4, 6)]
```

Github link:

https://github.com/pavan-kumar-202102/python_lab/commit/def5ca77413e84f69f835990707c950aeeedf056