DESIGN AND ANALYSIS OF ALGORITHM PROJECT

GROUP MEMBERS:

Ali Ammar Khan (17k-3713)

Taha Khan (17k-3753)

Section:B

ABSTRACT:

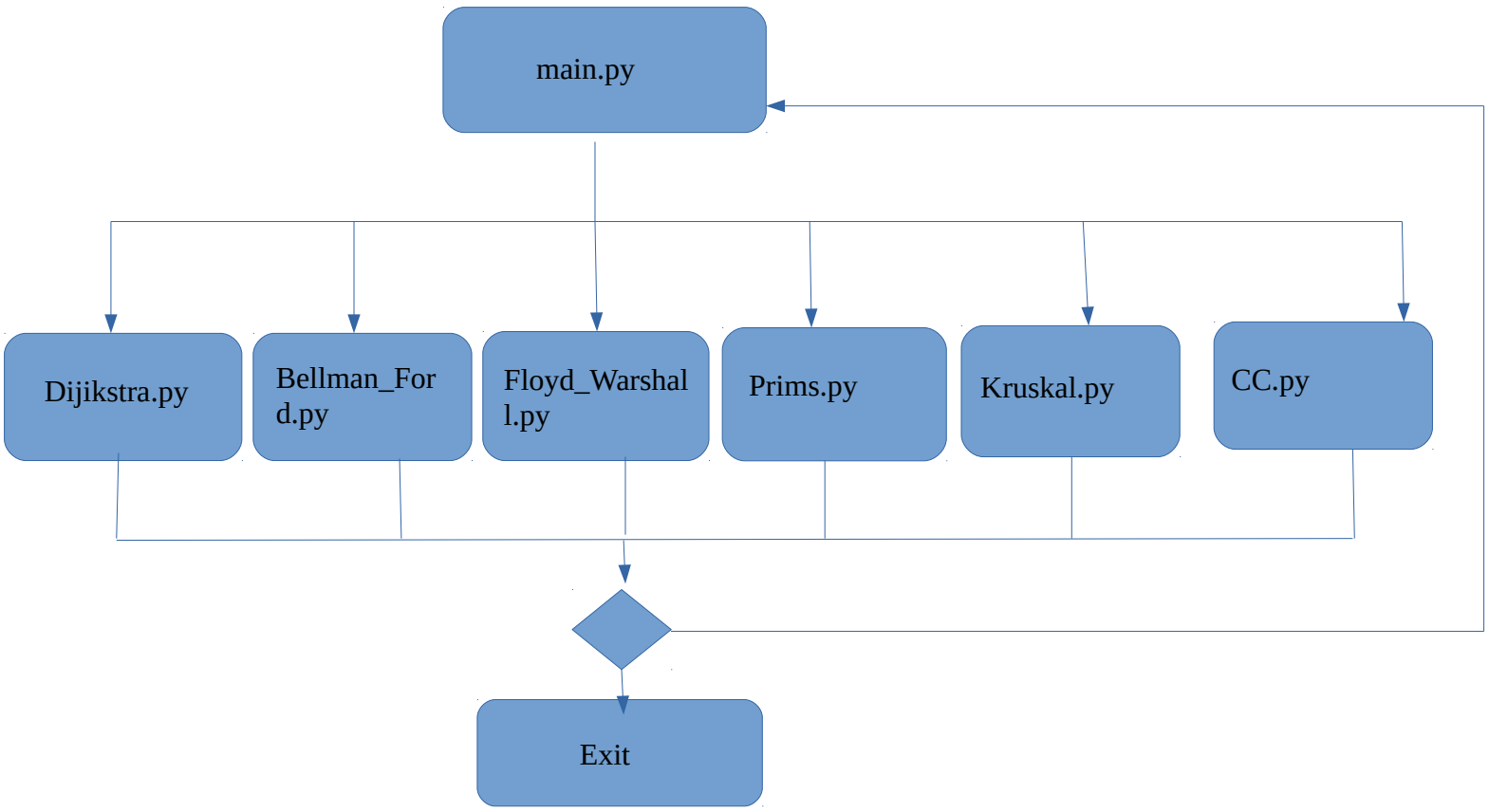
The given problem is to implement graph algorithms in a program and on the basis of the given set of inputs, display and then generate the results. Python has been used for the implementation of the given algorithms

INTRODUCTION:

The algorithms implemented are as follows:

1. Prim's Algorithm for minimum spanning tree
2. Kruskal's Algorithm for minimum spanning tree
3. Dijiskstra Algorithm for single source shortest path
4. Bellman Ford Algorithm for single source shortest path
5. Floyed Warshall for all pair shortest path
6. Clustering Coefficient in Graph theory

PROPOSED SYSTEM:



The program will initiate with the main.py file,which will prompt a very basic GUI and ask the user to select the input file , on selection it will prompt the graph according to the co-ordinates provided in the input file. After , it will ask for the algorithm that the user wants to implement and on selection will show the final graph and the result after running the algorithm on the inputs. The user can then select another input file or exit the program.

EXPERIMENTAL SETUP

The following considerations have been made while making this program:

1. The program will run on the input files provided with the project and any other files based on the same format
2. The algorithms used for implementing Prim's , Kruskal , Dijikstra, Bellman Ford are all based on the greedy approach while Floyd Warshall is implemented using the principles of Dynamic Programming. No particular algorithmic technique is considered while implementing Clustering Coefficient.
3. The implementation of the single source shortest path algorithms is done for the directed graphs but it can also work for the undirected graphs by making minor changes in the code . All those changes have been commented out.
4. The data structures used in this project are graphs and lists. Although dictionaries has also been used but their usage is very minor and just for the sake of showing output.
5. Python Libraries used in the project are
    - networkx(for strong graphs and processing)
    - matplotlib(for displaying graphs)
    - tkinter (for GUI)
    - statistics

RESULTS AND DISCUSSION

The following are the results obtained by running the algorithms on the given set of inpits. Prim's , Kruskal and cluster coefficient results are based on the undirected graph while the rest are for the directed. All weights are divided by 10000000

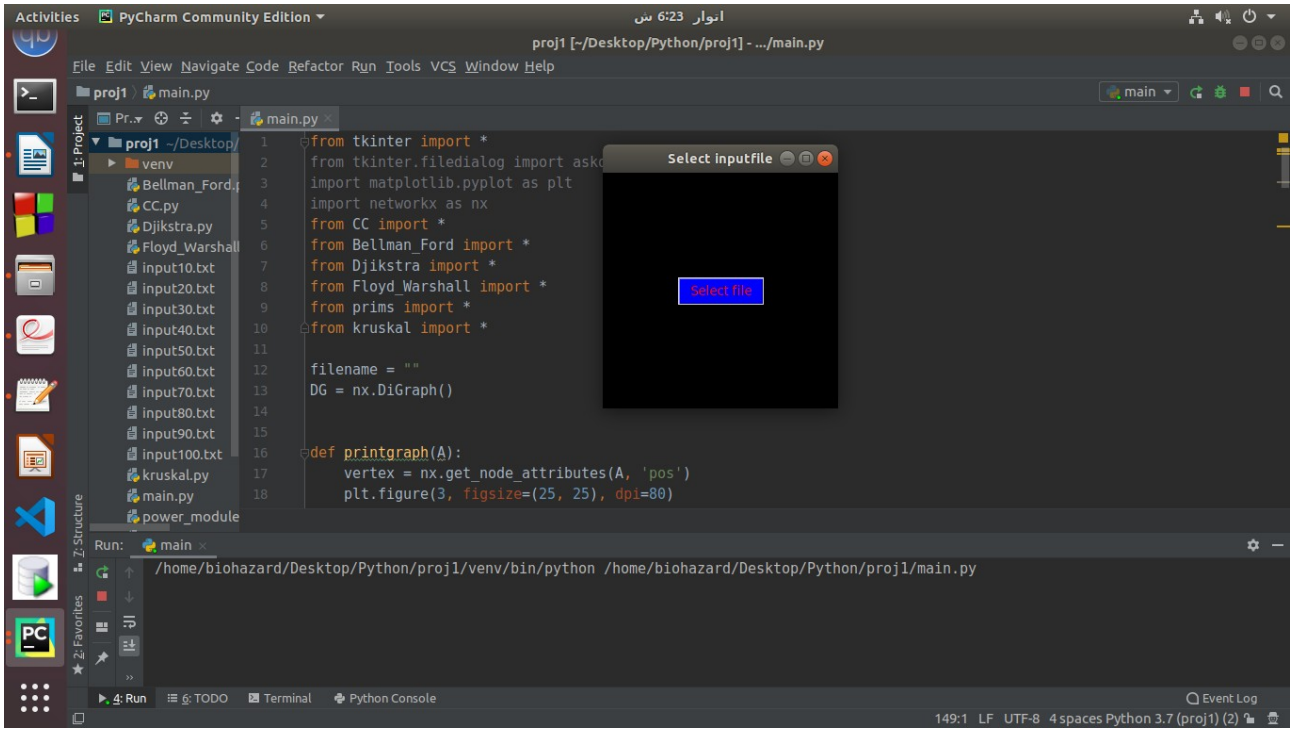
Benchmark	Prim's total cost	Kruskal total cost	Dijikstra source to all nodes	Bellman ford source to all nodes	Floyd Warshall sum of cost of each pair + inf(the cost of pairs not reached)	Cluster coefficient
Input 10	24.45	24.45	inf	inf	181.950+inf	0.6583
Input 20	51.45	51.45	inf	inf	1120.49+inf	0.4828
Input 30	87.6	87.6	inf	inf	1587.30+inf	0.6980
Input 40	137.099	137.099	inf	inf	2376.90+inf	0.7709
Input 50	133.05	133.05	inf	inf	1667.39+inf	0.6114
Input 60	201.15	201.15	inf	inf	4015.80+inf	0.7103
Input 70	195.7499	195.7499	inf	inf	3007.35+inf	0.6854
Input 80	249.30	249.30	inf	inf	4002.45+inf	0.7008
Input 90	287.100	287.100	inf	inf	4173.90+inf	0.7858
Input 100	304.50	304.50	inf	inf	5497.35+inf	0.6994

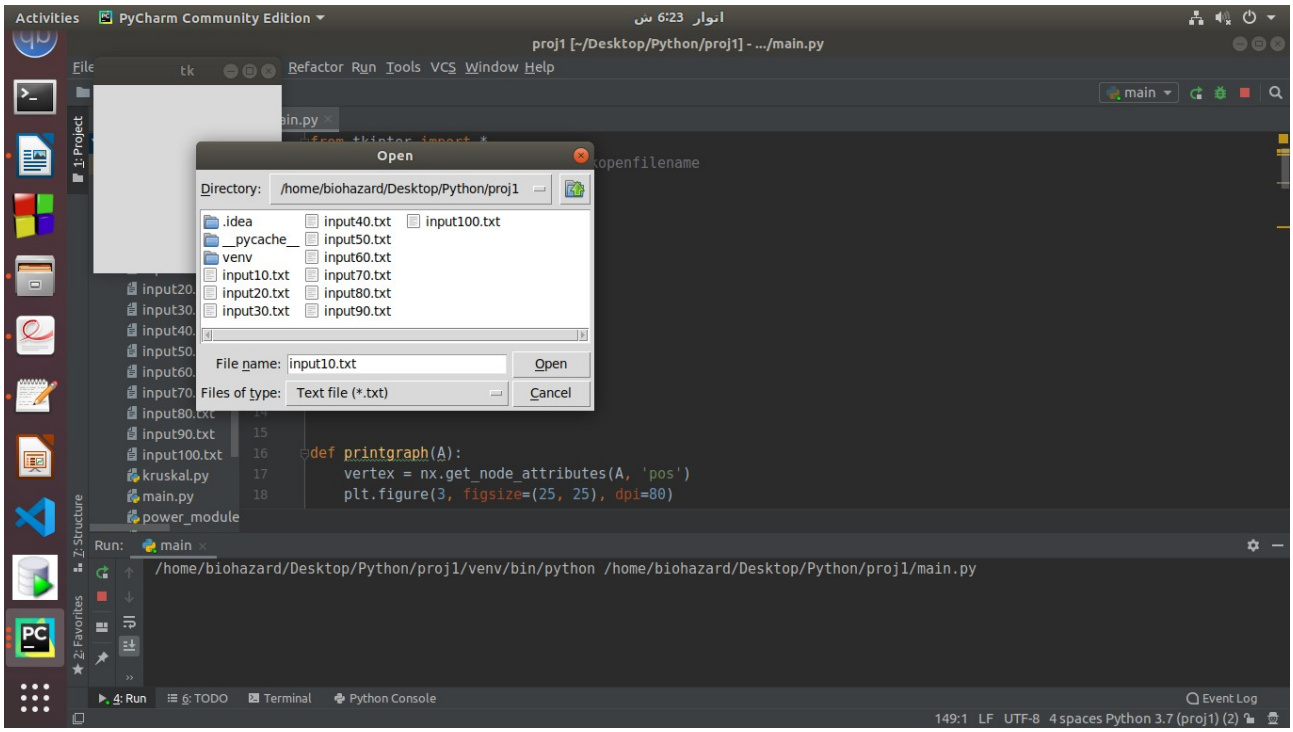
Since there are no outgoing edges from the source vertex (1) in Dijikstra and Bellmanford so the distance from source to all other vertices is inf . The Floyd Warshall cost is shown as the total distance of all the pairs plus inf (the distance of the unreachable pairs).

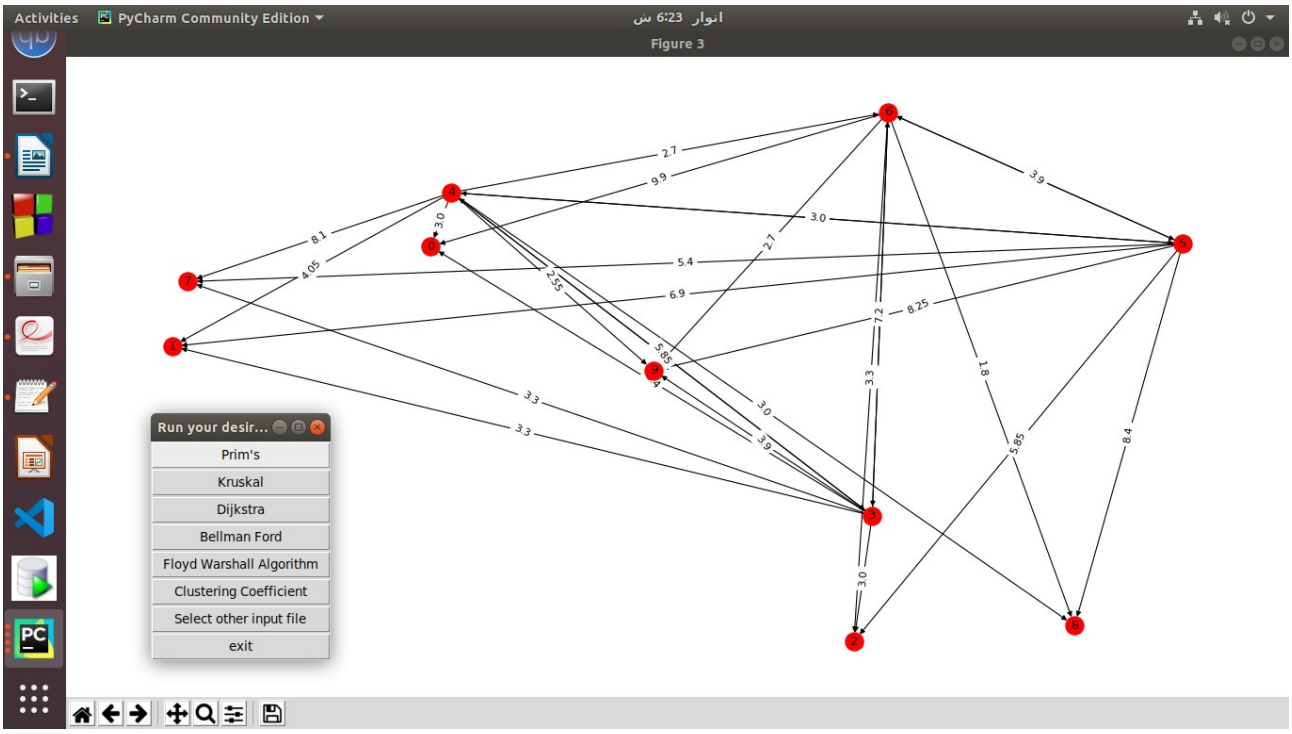
Dijkstra and Bellman ford for undirected graph

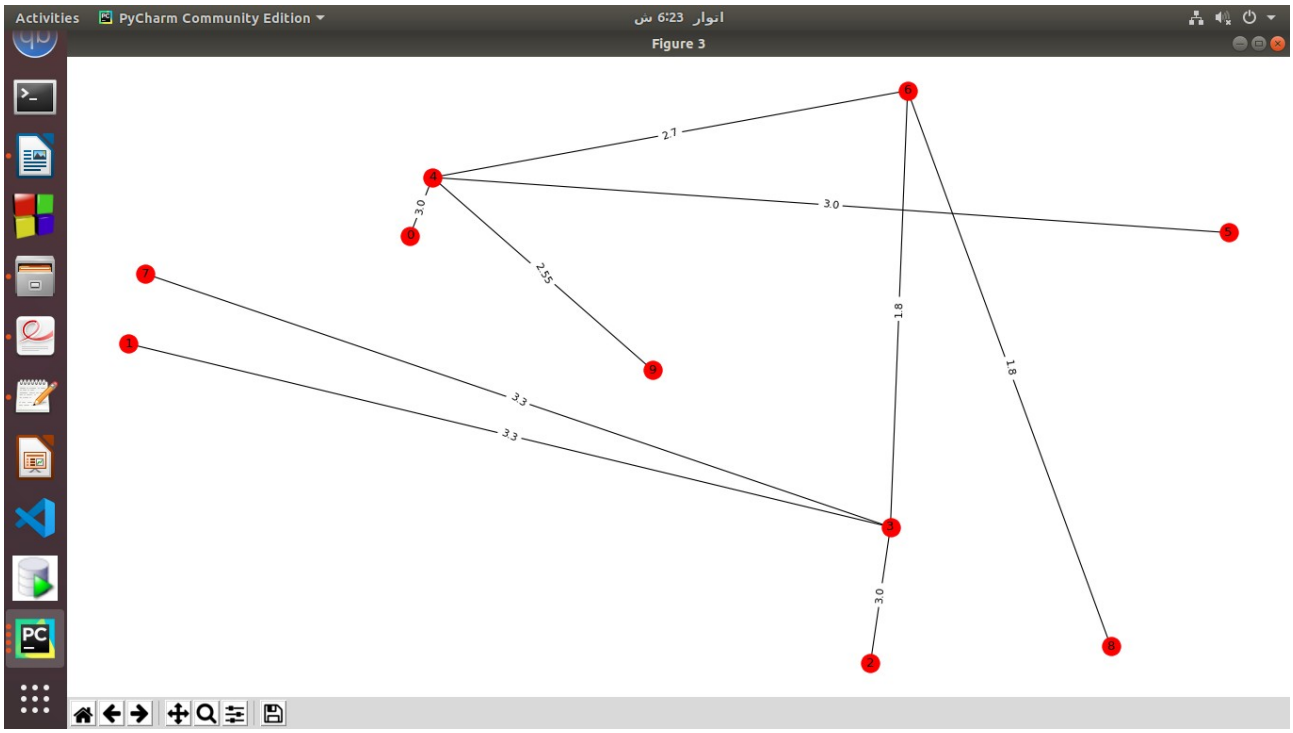
Benchmark	Dijikstra source to node 8	Bellman Ford source to node 8
Input 10	6.899	6.899
Input 20	7.19	7.19
Input 30	4.8	4.8
Input 40	11.4	11.4
Input 50	5.1	5.1
Input 60	6.15	6.15
Input 70	8.1	8.1
Input 80	3.75	3.75
Input 90	7.5	7.5
Input 100	7.199	7.199

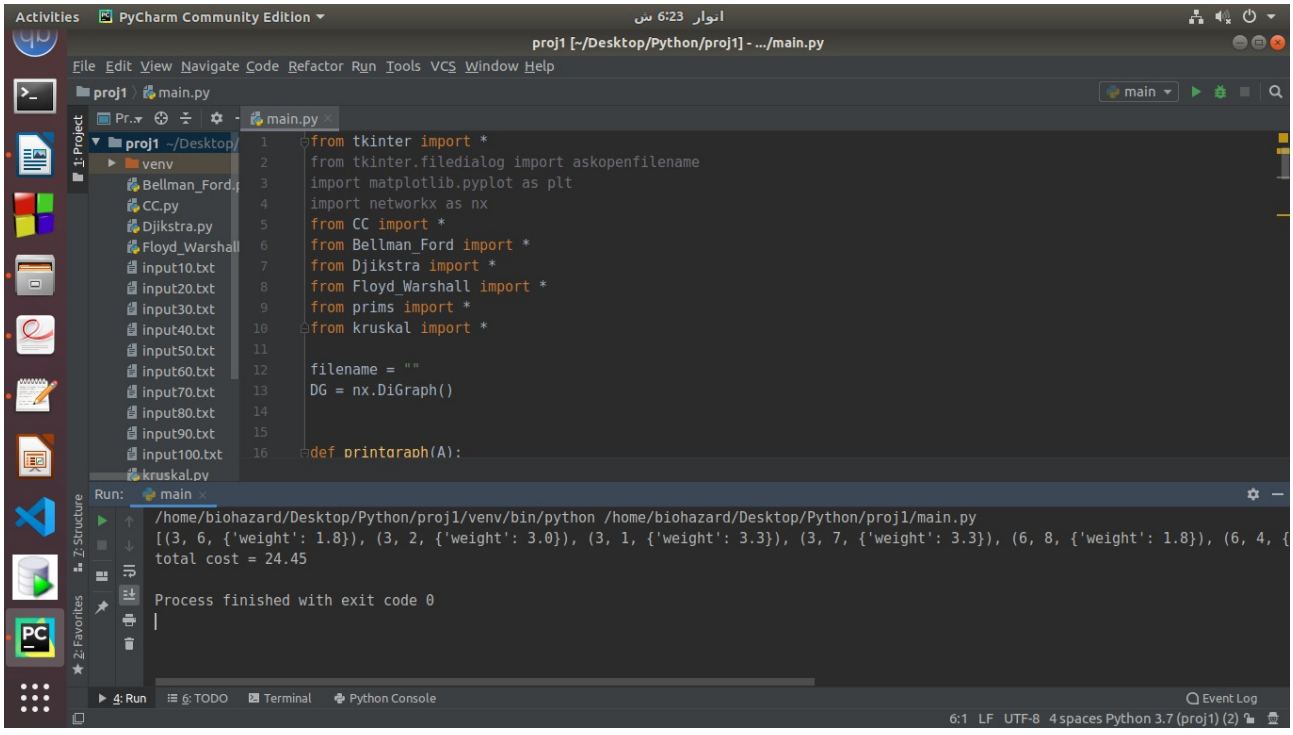
The following are the screenshots of a test case for Prim's MST




Activities
PyCharm Community Edition
proj1 [~/Desktop/Python/proj1] - .../main.py
File Refactor Run Tools VCS Window Help
Open
Directory: /home/biohazard/Desktop/Python/proj1
.idea
__pycache__
venv
input10.txt
input20.txt
input30.txt
input40.txt
input50.txt
input60.txt
input70.txt
input80.txt
input90.txt
input100.txt
File name: input10.txt
Open
Files of type: Text file (*.txt)
Cancel
def printgraph(A):
vertex = nx.get_node_attributes(A, 'pos')
plt.figure(3, figsize=(25, 25), dpi=80)
Run: main
/home/biohazard/Desktop/Python/proj1/venv/bin/python /home/biohazard/Desktop/Python/proj1/main.py
4: Run
6: TODO
Terminal
Python Console
Event Log
149:1 LF UTF-8 4 spaces Python 3.7 (proj1) (2)


Activities
PyCharm Community Edition
Figure 3
Run your desir...
Prim's
Kruskal
Dijkstra
Bellman Ford
Floyd Warshall Algorithm
Clustering Coefficient
Select other input file
exit


Activities
PyCharm Community Edition
Figure 3
2.1
3.0
3.0
2.55
1.8
1.8
3.3
3.3
3.0


Activities
PyCharm Community Edition
proj1 [~/Desktop/Python/proj1] - .../main.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
proj1
main.py
venv
Bellman_Ford.
CC.py
Djikstra.py
Floyd_Warshall
input10.txt
input20.txt
input30.txt
input40.txt
input50.txt
input60.txt
input70.txt
input80.txt
input90.txt
input100.txt
kruskal.py
from tkinter import *
from tkinter.filedialog import askopenfilename
import matplotlib.pyplot as plt
import networkx as nx
from CC import *
from Bellman_Ford import *
from Djikstra import *
from Floyd_Warshall import *
from prims import *
from kruskal import *
filename = ""
DG = nx.DiGraph()
def printgraph(A):
Run:
main
/home/biohazard/Desktop/Python/proj1/venv/bin/python /home/biohazard/Desktop/Python/proj1/main.py
[(3, 6, {'weight': 1.8}), (3, 2, {'weight': 3.0}), (3, 1, {'weight': 3.3}), (3, 7, {'weight': 3.3}), (6, 8, {'weight': 1.8}), (6, 4, {
total cost = 24.45
Process finished with exit code 0
4: Run
6: TODO
Terminal
Python Console
Event Log
6:1 LF UTF-8 4 spaces Python 3.7 (proj1) (2)

CONCLUSIONS:

REFERENCES:

1. Introduction to the Design & Analysis of Algorithms by Anany Levitin
2. https://www.geeksforgeeks.org/fundamentals-of-algorithms/
3. https://stackoverflow.com/