GRAPH ALGORITHMS

ASSIGNMENT 03 – **Centrality Measures**

MUHAMMAD HARRIS
BCS203193

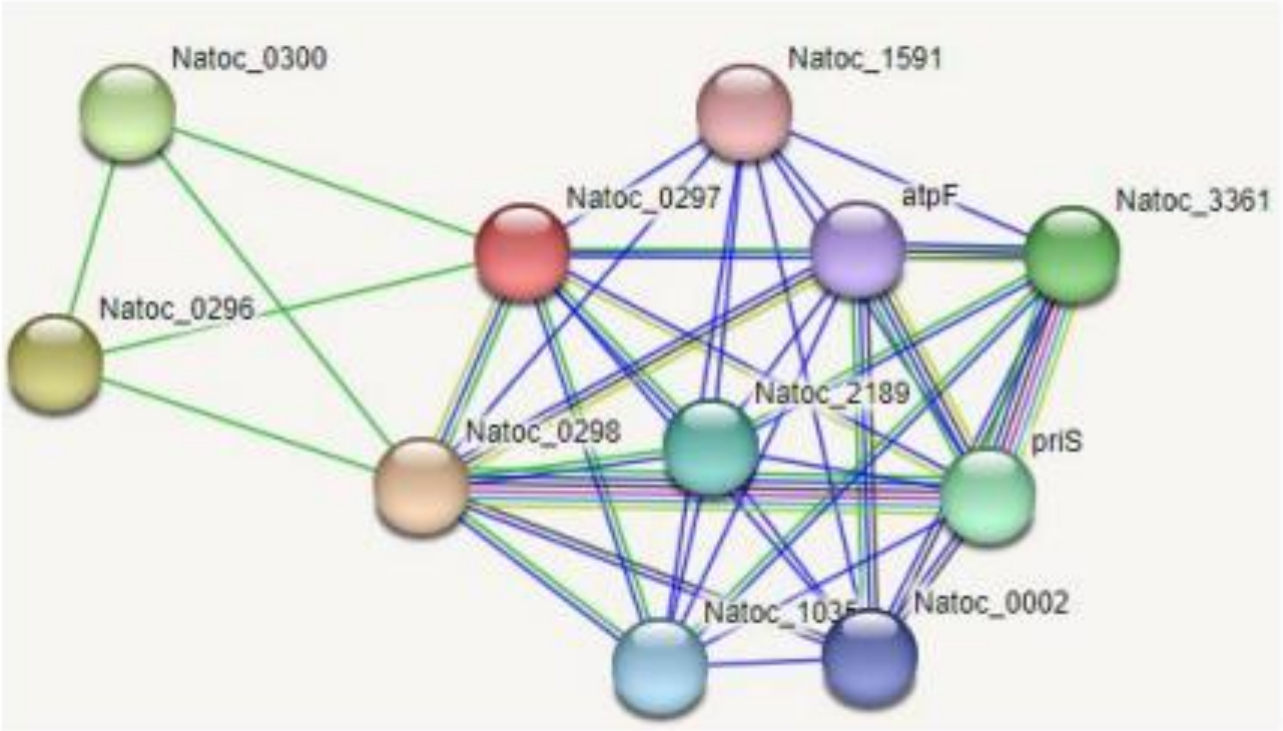
Table of Contents

Table of Figures

GRAPH ALGORITHMS

ASSIGNMENT 03 – **Centrality Measures**

Consider the following biological network of NATOC protein with its interactions with other proteins as shown in the figure below. (Excel file of dataset containing edges is provided)



Perform the following tasks:

1. ***Read the dataset file in Python***
2. ***Identify the unique proteins and display them which represents the nodes***
3. ***Calculate the degrees of all the nodes***
4. ***Save the edges in a list after reading from excel file.***
5. ***Identify the important proteins using:***
    a. **Degree Centrality**
    b. **Betweenness Centrality**
    c. **Closeness Centrality**
6. ***Show the results of all centrality measures along with values.***

Python Program

Code

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithms - Assignment 3

import networkx
import pandas
import os
import matplotlib.pyplot
import openpyxl

menu = """*GRAPH ALGORITHM ASSIGNMENT*
1. Visualize Graph
2. Degrees of all Nodes
3. Degree Centrality
4. Betweenness Centrality
5. Closeness Centrality
6. Exit"""

# function to visualize the graph
def visualizeGraph(g):
    networkx.draw(g, pos=networkx.spring_layout(g), with_labels=True,
node_size=1000, node_color='#49be25')
    matplotlib.pyplot.show()

# function to display degree of all nodes
def printDegrees(g, nodes):
    print('\nDegree of all nodes:')
    for n in nodes:
        print(n, ' = ', graph.degree(n))

# function to display degree centrality
def printDegreeCentrality(g, nodes):
    print('\nDegree Centrality:')
    degreeCentrality = networkx.degree_centrality(g)
    for n in nodes:
        print(n, ' = ', degreeCentrality[n])

# function to display betweenness centrality
def printBetweennessCentrality(g, nodes):
```

```
    print('\nBetweenness Centrality:')
    betweennessCentrality = networkx.betweenness_centrality(g)
    for n in nodes:
        print(n, ' = ', betweennessCentrality[n])

# function to display closeness centrality
def printClosenessCentrality(g, nodes):
    print('\nCloseness Centrality:')
    closenessCentrality = networkx.closeness_centrality(g)
    for n in nodes:
        print(n, ' = ', closenessCentrality[n])

# main

# importing data from .xlsx file
filePath = r'C:\Users\Harri\OneDrive\Desktop\Assignment3_Dataset.xlsx'
data = pandas.read_excel(filePath)

# creating graph using data from .xlsx file
graph = networkx.Graph()
for i in range(0, len(data)):
    graph.add_edge(data.iloc[i, 0], data.iloc[i, 1]) # creating edge
between columns of each row of Excel sheet

# creating a node list of graph
nodeList = graph.nodes()

# menu system
while True:
    os.system('cls')
    print(menu)
    option = input('option: ')[0]
    if option == '1':
        visualizeGraph(graph)
    elif option == '2':
        printDegrees(graph, nodeList)
        input('\npress enter key to return to menu...')
    elif option == '3':
        printDegreeCentrality(graph, nodeList)
        input('\npress enter key to return to menu...')
    elif option == '4':
        printBetweennessCentrality(graph, nodeList)
```

```
        input('\npress enter key to return to menu...')
    elif option == '5':
        printClosenessCentrality(graph, nodeList)
        input('\npress enter key to return to menu...')
    elif option == '6':
        break
    else:
        continue
```

Screenshots

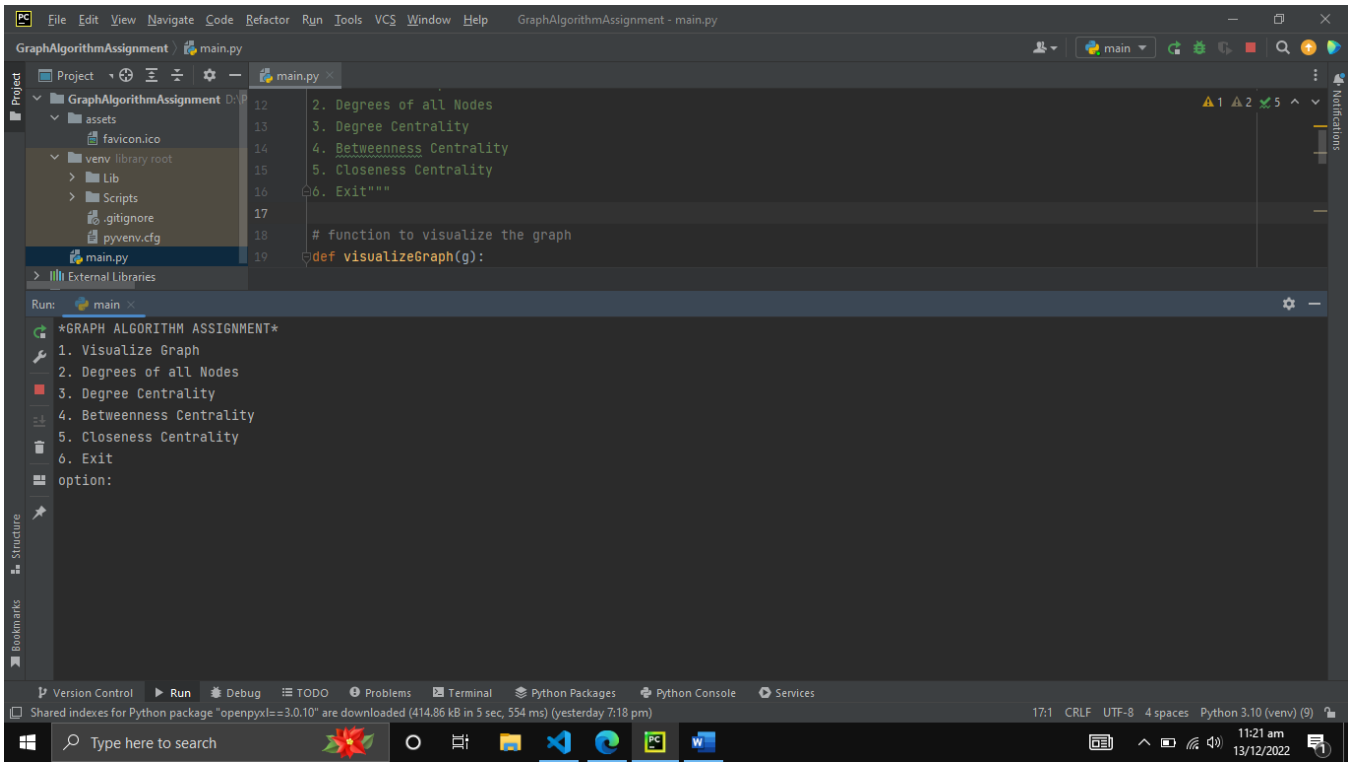


Figure 1: Program Menu

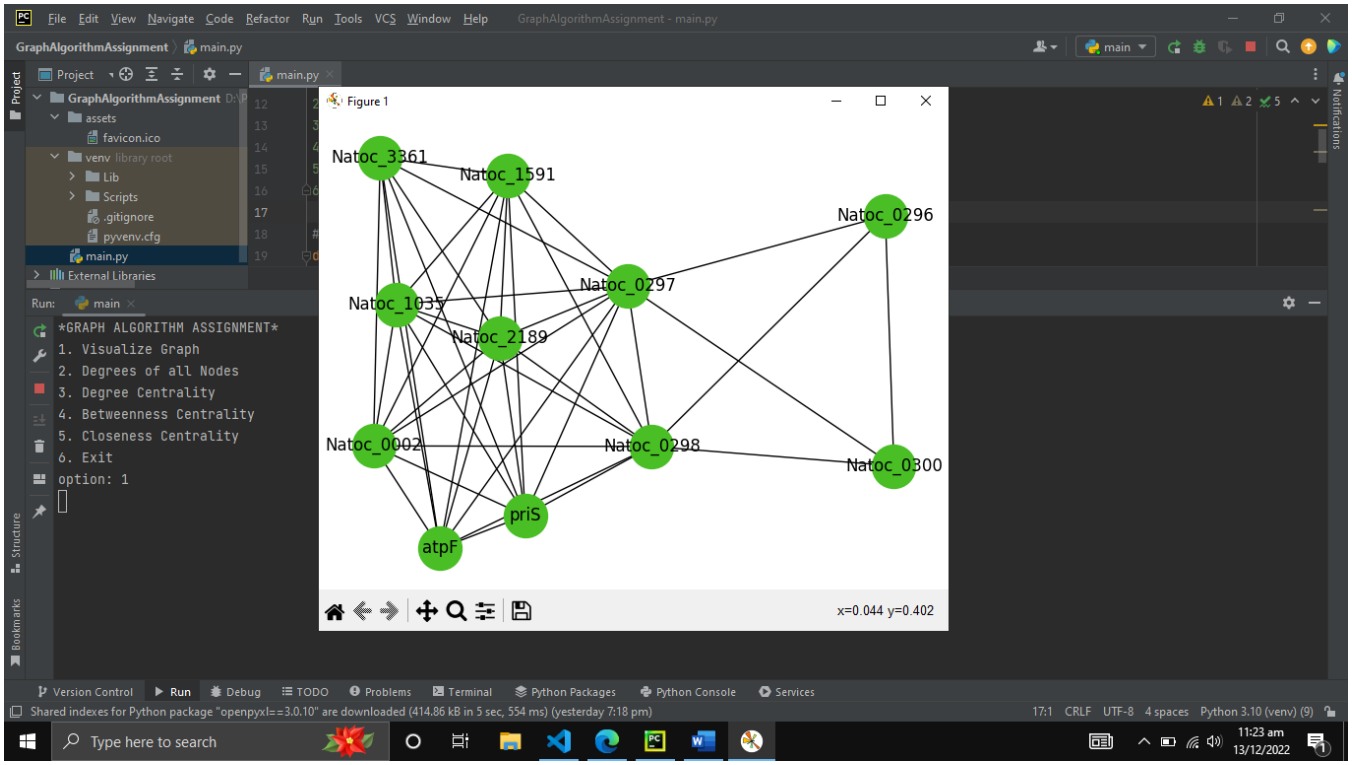


Figure 2: Visualize Graph

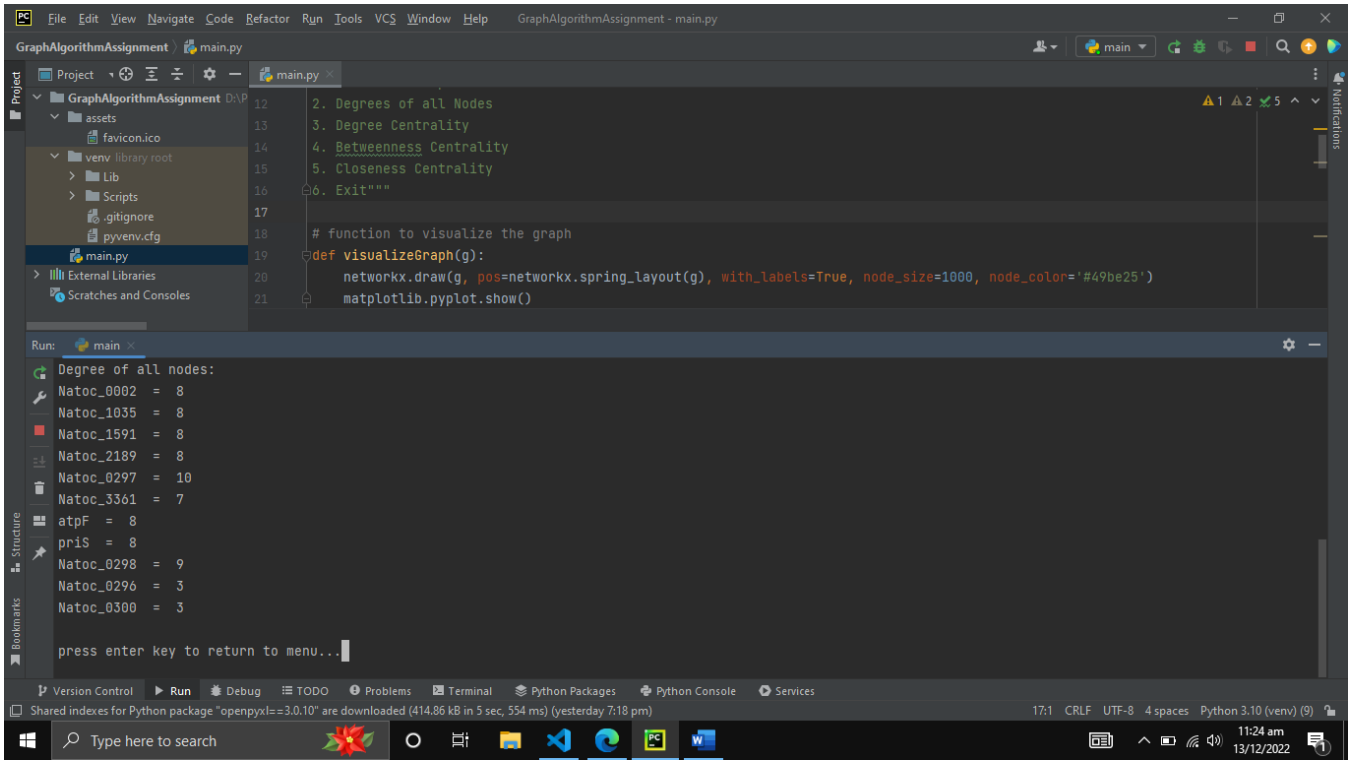


Figure 3: Degree of all nodes

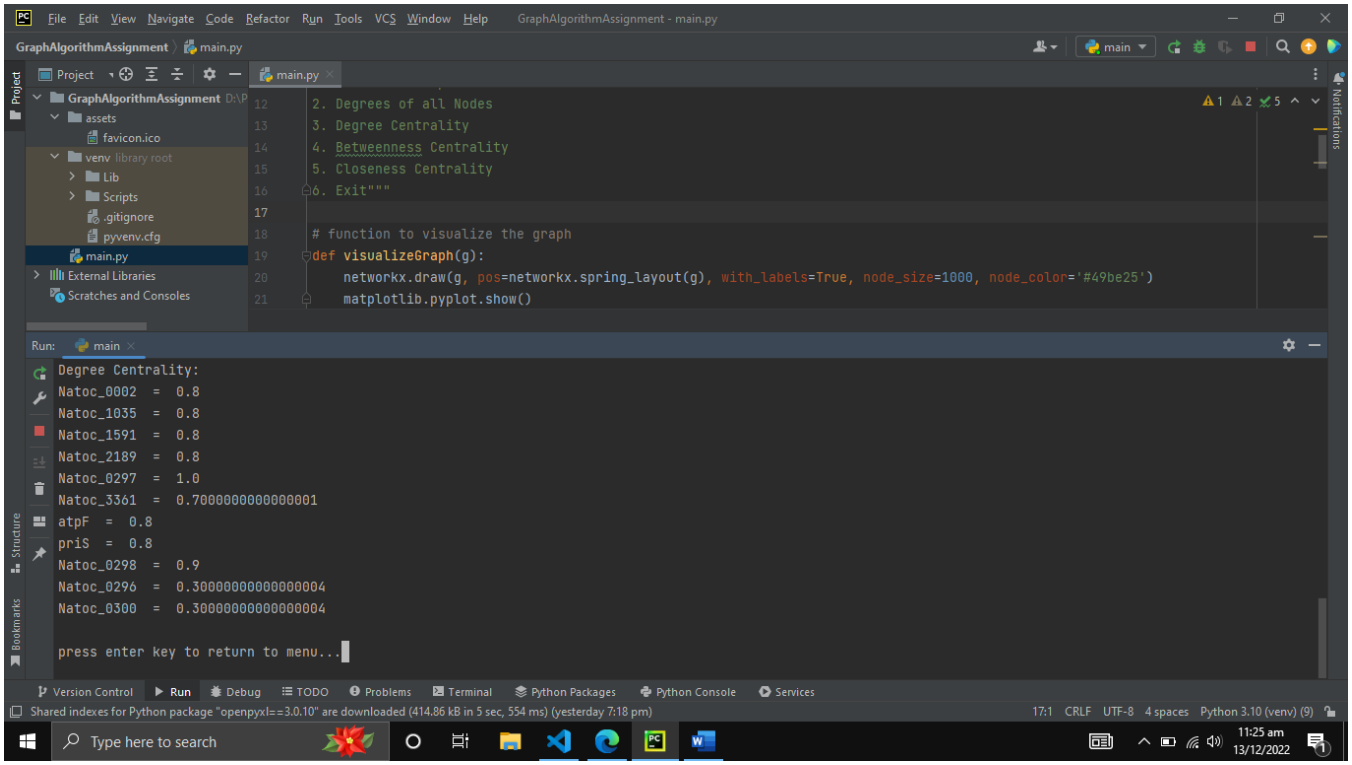


Figure 4: Degree Centrality

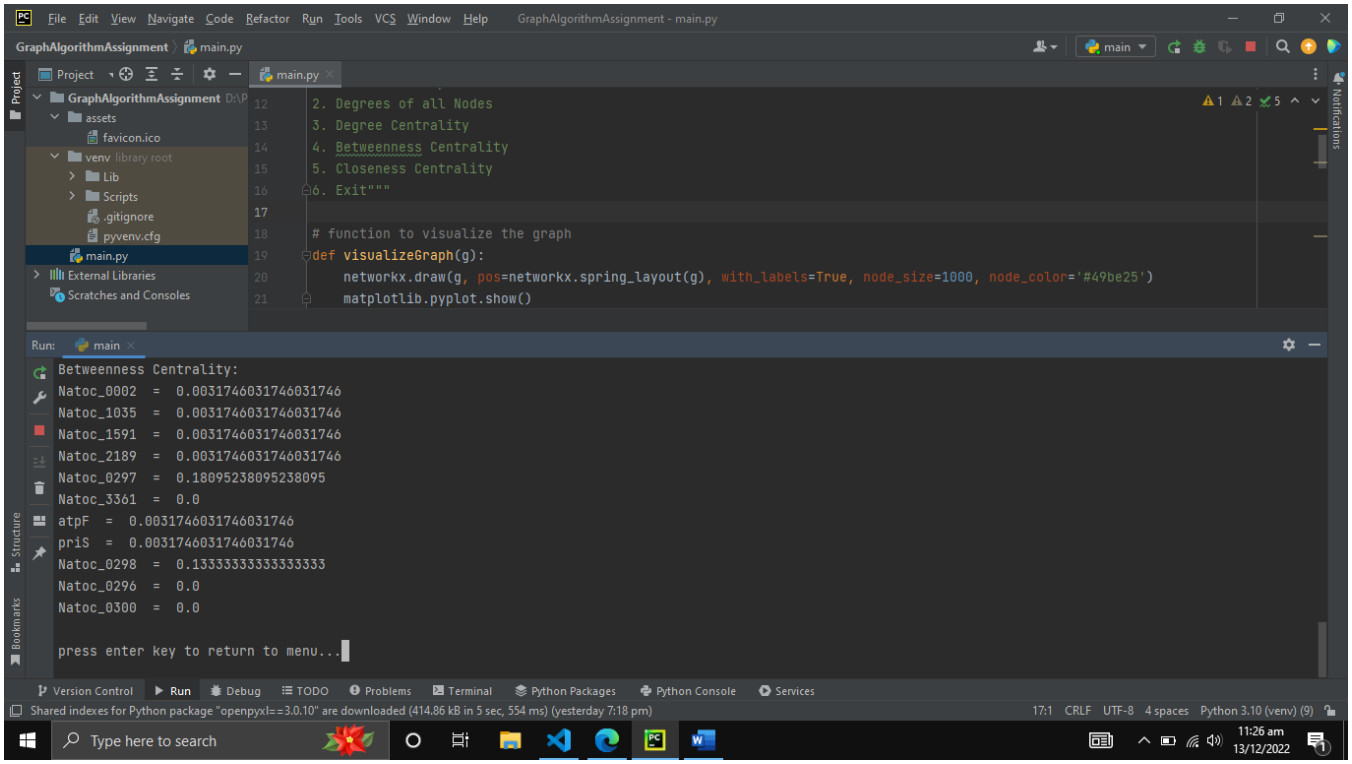


Figure 5: Betweenness Centrality

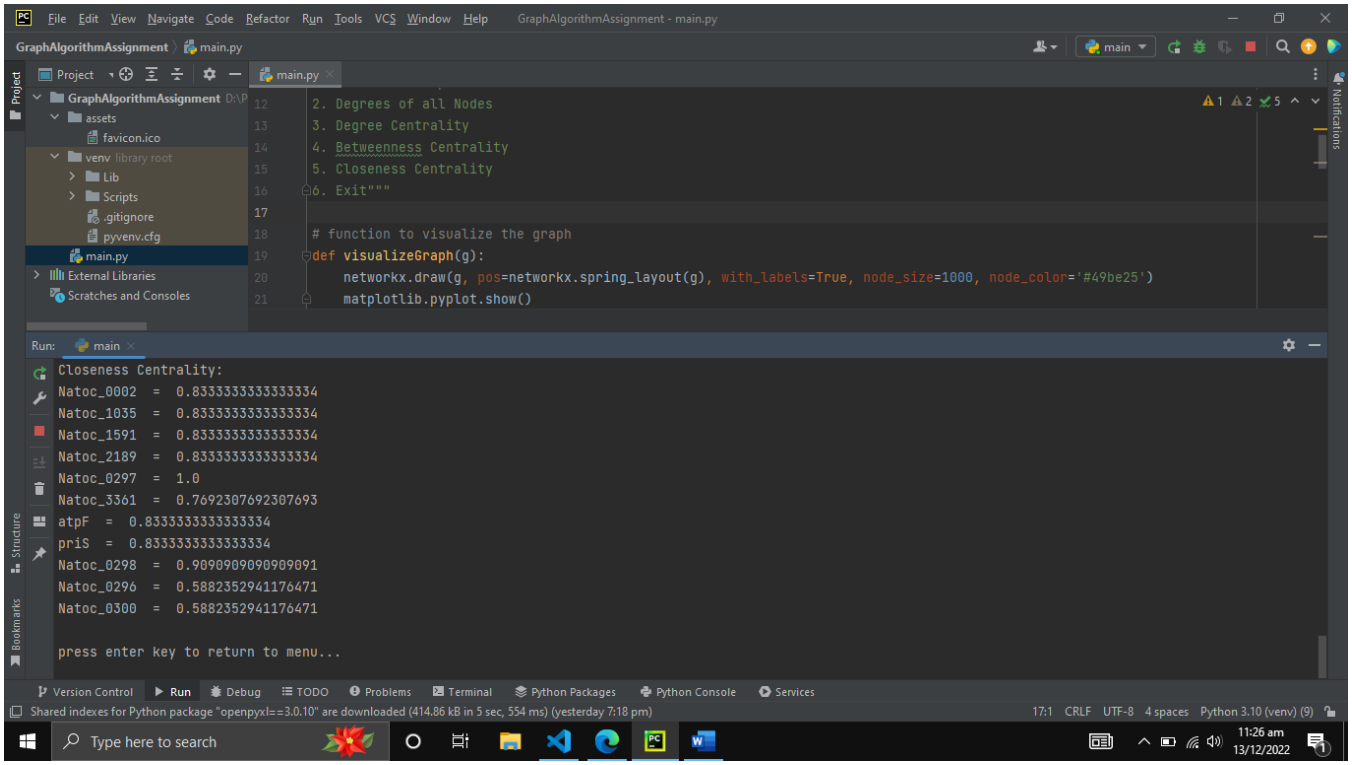


Figure 6: Closeness Centrality

Python Program with UI

Code

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithms - Assignment 3

import networkx
import pandas
import matplotlib.pyplot
import openpyxl
from tkinter import *
import tkinter.filedialog

# init graph
mainGraph = networkx.Graph()

# init fileDirectory
fileDirectory = 'NULL'
```

```
# function to select excel file and create graph from it
def selectExcelFile():
    global fileDirectory
    global statusLabel
    global mainGraph
    fileTypes = (('excel files', '*.xlsx'), ('All files', '*.*'))
    fileDirectory = tkinter.filedialog.askopenfilename(title='Select
excel file to create graph', initialdir='/', filetypes=fileTypes)
    statusLabel.config(text='STATUS: excel file selected',
fg='#2ce20f')
    excelData = pandas.read_excel(fileDirectory)
    for i in range(0, len(excelData)):
        mainGraph.add_edge(excelData.iloc[i, 0], excelData.iloc[i, 1])

# function to visualize the graph
def visualizeGraph():
    networkx.draw(mainGraph, pos=networkx.spring_layout(mainGraph),
with_labels=True, node_size=1000, node_color='#49be25')
    matplotlib.pyplot.show()

# create window that displays degree of nodes
def displayNodeDegree():

    # create node list
    nodeList = mainGraph.nodes()

    # create new window
    new = Tk()
    new.title('Degree of Nodes')
    new.geometry('560x520')
    new.resizable(False, False)
    new.iconbitmap(r'D:\Project\GraphAlgorithmAssignment\assets\favico
n.ico')

    # null label
    nullLabel3 = Label(new, text='\n')
    nullLabel3.pack()

    textLabel = Label(new, text='DEGREE OF ALL NODES')
    textLabel.pack()

    # null label
```

```
    nullLabel4 = Label(new, text='\n')
    nullLabel4.pack()

    # display nodes along with degree
    for n in nodeList:
        label = Label(new, text=n + ': ' + str(mainGraph.degree(n)))
        label.pack()

# create window that displays degree centrality of nodes
def displayDegreeCentrality():
    # create node list
    nodeList = mainGraph.nodes()

    # create new window
    new = Tk()
    new.title('Degree Centrality of Nodes')
    new.geometry('560x520')
    new.resizable(False, False)
    new.iconbitmap(r'D:\Project\GraphAlgorithmAssignment\assets\favico
n.ico')

    # null label
    nullLabel3 = Label(new, text='\n')
    nullLabel3.pack()

    textLabel = Label(new, text='Degree Centrality')
    textLabel.pack()

    # null label
    nullLabel4 = Label(new, text='\n')
    nullLabel4.pack()

    # get degree centrality as a dictionary
    degreeCentrality = networkx.degree_centrality(mainGraph)

    # display nodes along with degree centrality
    for n in nodeList:
        label = Label(new, text=n + ': ' + str(degreeCentrality[n]))
        label.pack()

# create window that displays betweenness centrality of nodes
def displayBetweennessCentrality():
```

```
    # create node list
    nodeList = mainGraph.nodes()

    # create new window
    new = Tk()
    new.title('Betweenness Centrality of Nodes')
    new.geometry('560x520')
    new.resizable(False, False)
    new.iconbitmap(r'D:\Project\GraphAlgorithmAssignment\assets\favico
n.ico')

    # null label
    nullLabel3 = Label(new, text='\n')
    nullLabel3.pack()

    textLabel = Label(new, text='Betweenness Centrality')
    textLabel.pack()

    # null label
    nullLabel4 = Label(new, text='\n')
    nullLabel4.pack()

    # get betweenness centrality as a dictionary
    betweennessCentrality = networkx.betweenness_centrality(mainGraph)

    # display nodes along with degree centrality
    for n in nodeList:
        label = Label(new, text=n + ': ' +
str(betweennessCentrality[n]))
        label.pack()

# create window that displays closeness centrality of nodes
def displayClosenessCentrality():
    # create node list
    nodeList = mainGraph.nodes()

    # create new window
    new = Tk()
    new.title('Closeness Centrality of Nodes')
    new.geometry('560x520')
    new.resizable(False, False)
```

```
    new.iconbitmap(r'D:\Project\GraphAlgorithmAssignment\assets\favico
n.ico')

    # null label
    nullLabel3 = Label(new, text='\n')
    nullLabel3.pack()

    textLabel = Label(new, text='Closeness Centrality')
    textLabel.pack()

    # null label
    nullLabel4 = Label(new, text='\n')
    nullLabel4.pack()

    # get closeness centrality as a dictionary
    closenessCentrality = networkx.closeness_centrality(mainGraph)

    # display nodes along with degree centrality
    for n in nodeList:
        label = Label(new, text=n + ': ' +
str(closenessCentrality[n]))
        label.pack()

# main

# creating UI
root = Tk()
root.geometry('560x520')
root.resizable(False, False)

# add favicon to window
root.iconbitmap(r'D:\Project\GraphAlgorithmAssignment\assets\favicon.i
co')

# adding window title
root.title('Graph Algorithms - Assignment 3 - MUHAMMAD HARRIS -
BCS203193')

# null label
nullLabel0 = Label(root, text='\n')
nullLabel0.pack()
```

```
# creating label
headingLabel = Label(root, text='GRAPH ALGORITHMS')
headingLabel.pack()

# null label
nullLabel = Label(root, text='\n')
nullLabel.pack()

# file frame
fileFrame = LabelFrame(root, text='Select excel file to create graph',
padx=10, pady=10)
fileFrame.pack()

# select file
selectFileButton = Button(fileFrame, text='Open File',
command=selectExcelFile)
selectFileButton.pack()

# status label for excel file
statusLabel = Label(fileFrame, text='STATUS: no file selected', bd=1,
relief=SUNKEN, padx=3, fg='#E2260f')
statusLabel.pack(pady=10)

# null label
nullLabel2 = Label(root, text='\n')
nullLabel2.pack()

# creating menu frame
menuFrame = LabelFrame(root, text='Options for graph', padx=10,
pady=10, labelanchor='n')
menuFrame.pack(padx=10, pady=10)

# creating button for visualizing graph
button_visualizeGraph = Button(menuFrame, text='Visualize Graph',
command=lambda: visualizeGraph(), padx=22.5)
button_visualizeGraph.pack(pady=5)

# creating button for degree of nodes
button_nodeDegrees = Button(menuFrame, text='Degree of Nodes',
padx=18.5, command=lambda: displayNodeDegree())
button_nodeDegrees.pack(pady=5)
```

```
# creating button for degree centrality
button_degreeCentrality = Button(menuFrame, text='Degree Centrality',
padx=16.5, command=lambda: displayDegreeCentrality())
button_degreeCentrality.pack(pady=5)

# creating button for betweenness centrality
button_betweennessCentrality = Button(menuFrame, text='Betweenness
Centrality', command=lambda: displayBetweennessCentrality())
button_betweennessCentrality.pack(pady=5)

# creating button for closeness centrality
button_closenessCentrality = Button(menuFrame, text='Closeness
Centrality', padx=9, command=lambda: displayClosenessCentrality())
button_closenessCentrality.pack(pady=5)

# run UI
root.mainloop()
```

Screenshot

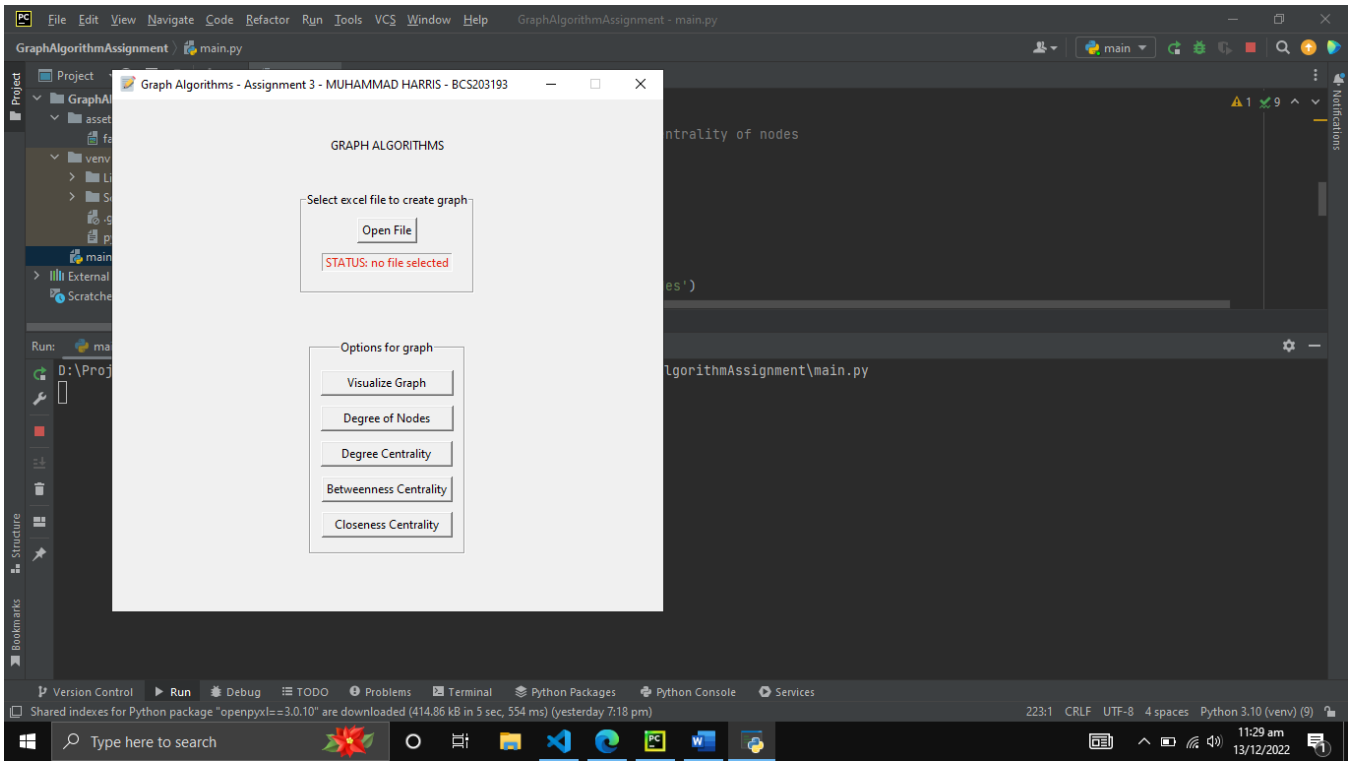


Figure 7: Initial Screen

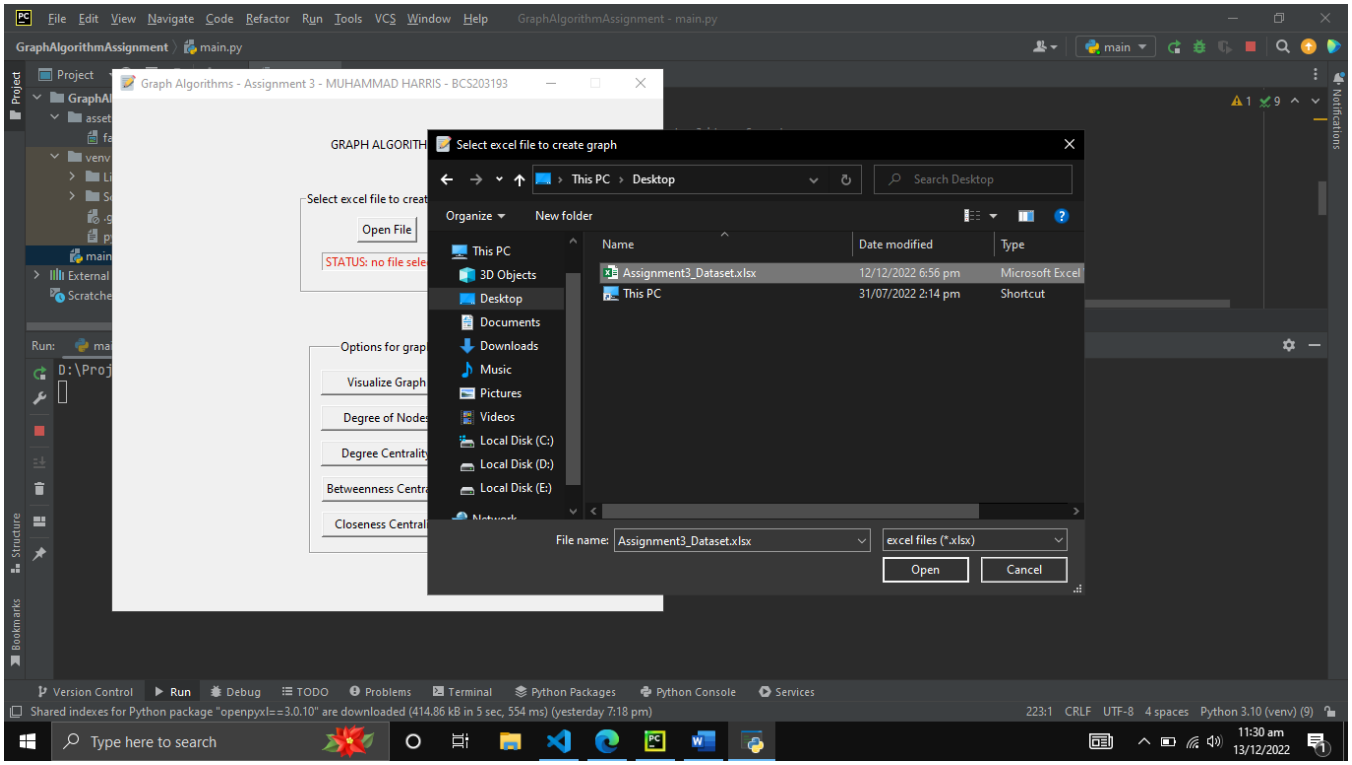


Figure 8: Excel File Selection

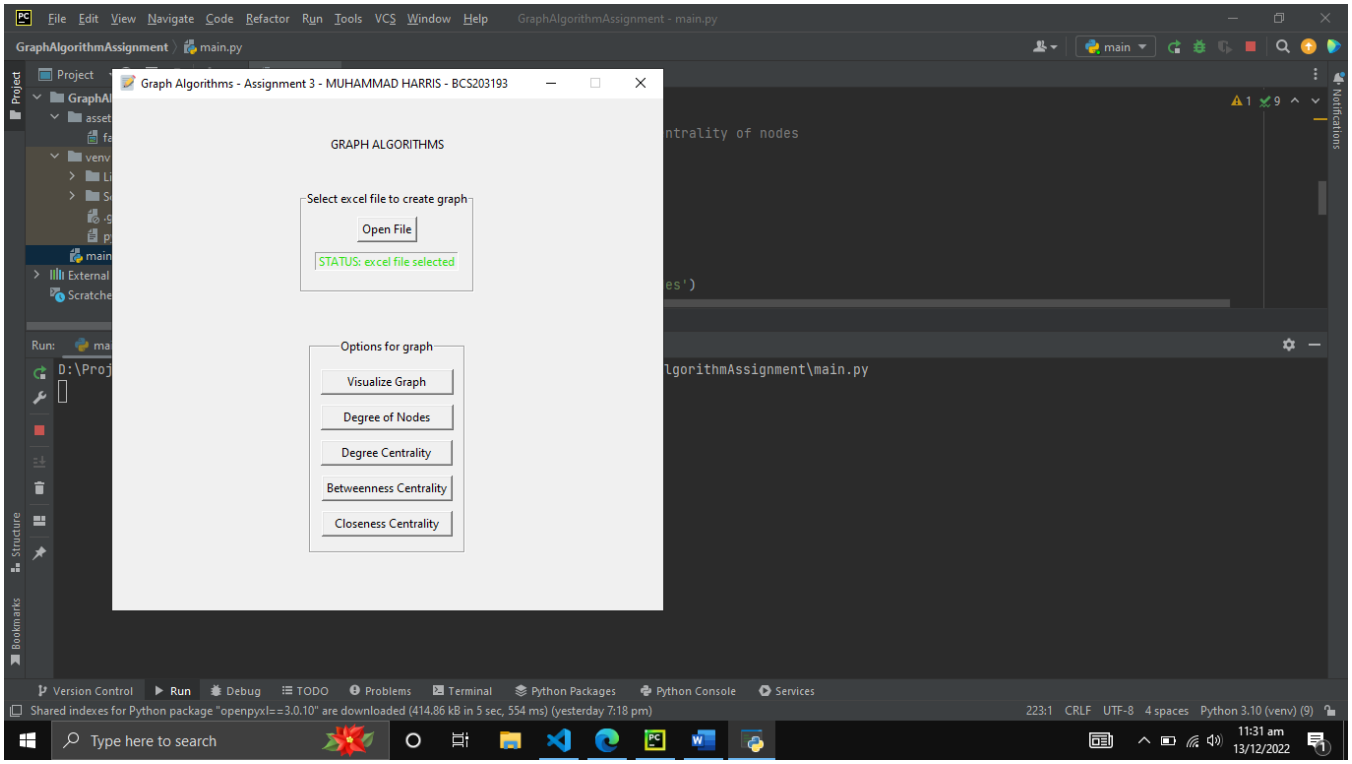


Figure 9: Post Selection Screen

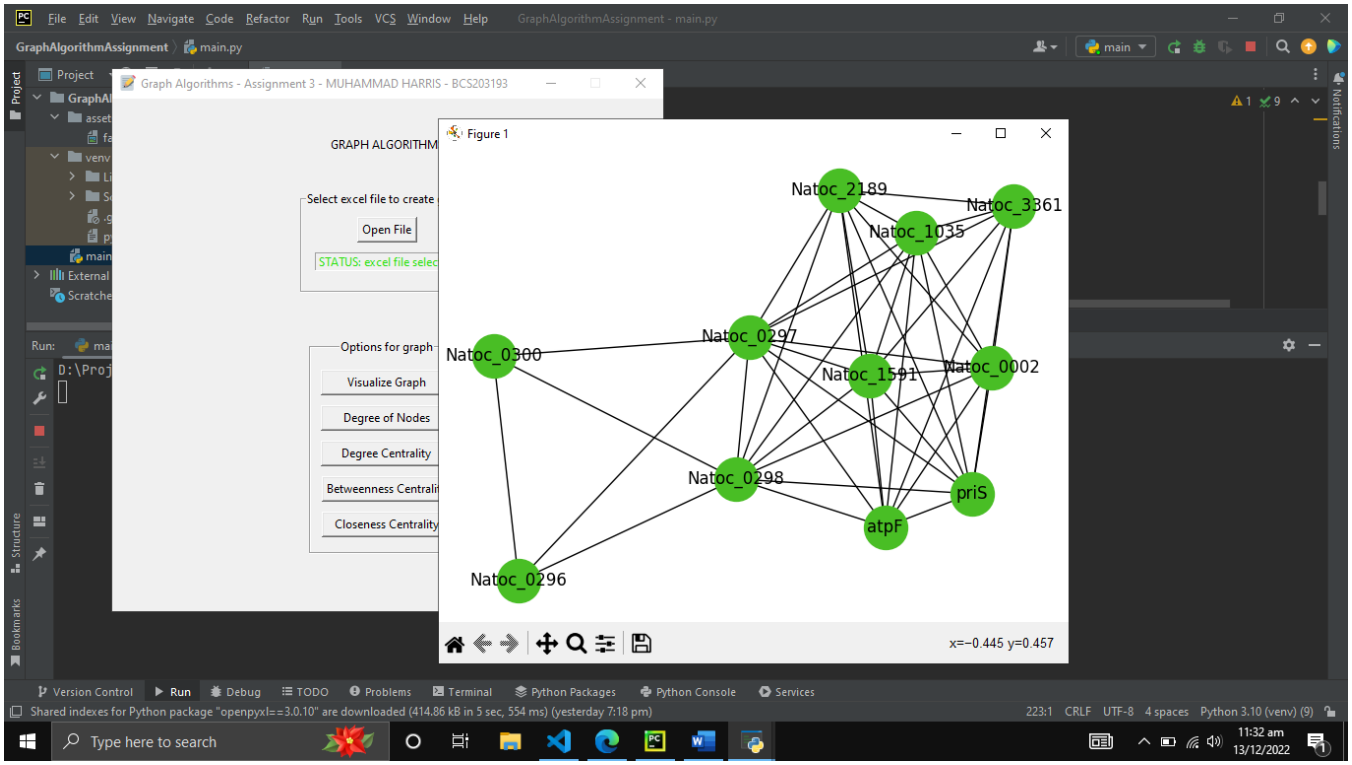


Figure 10: Visualize Graph UI

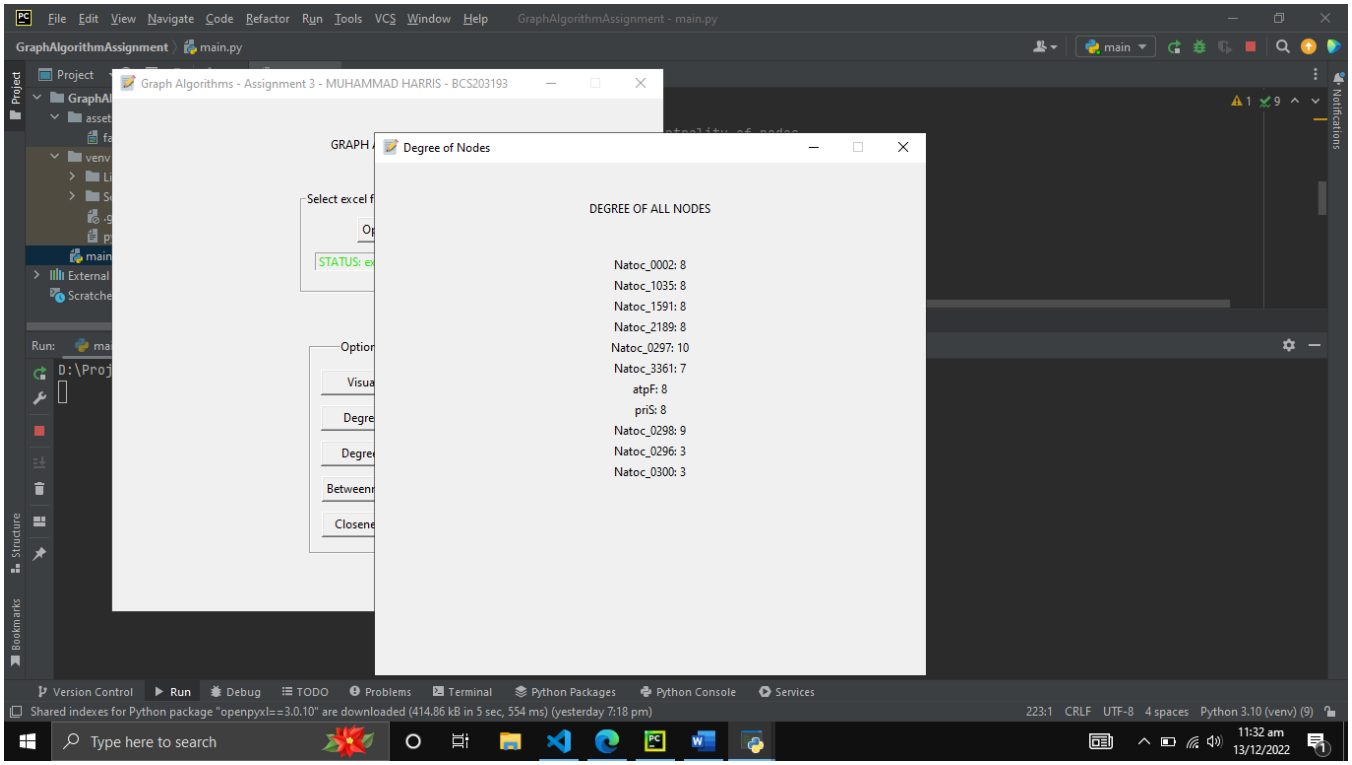


Figure 11: Degree of Nodes UI

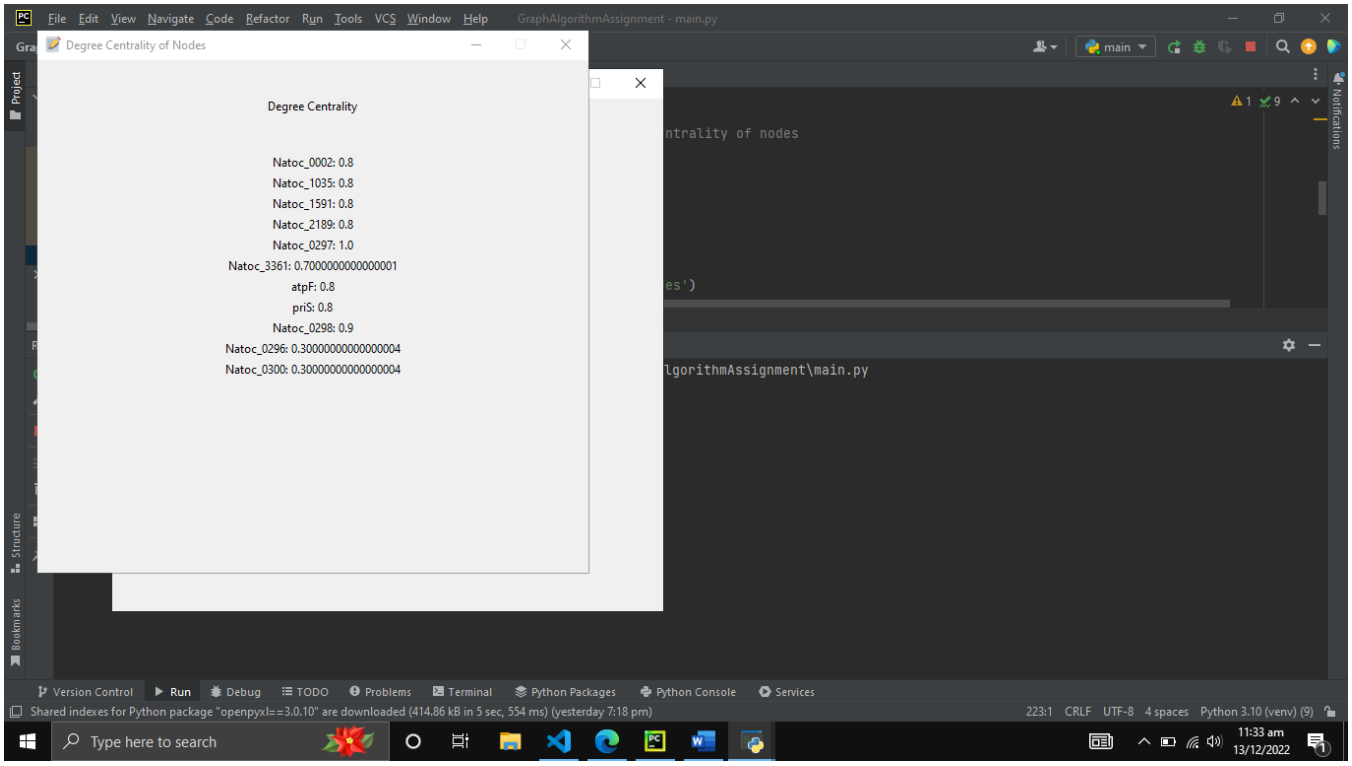


Figure 12: Degree Centrality UI

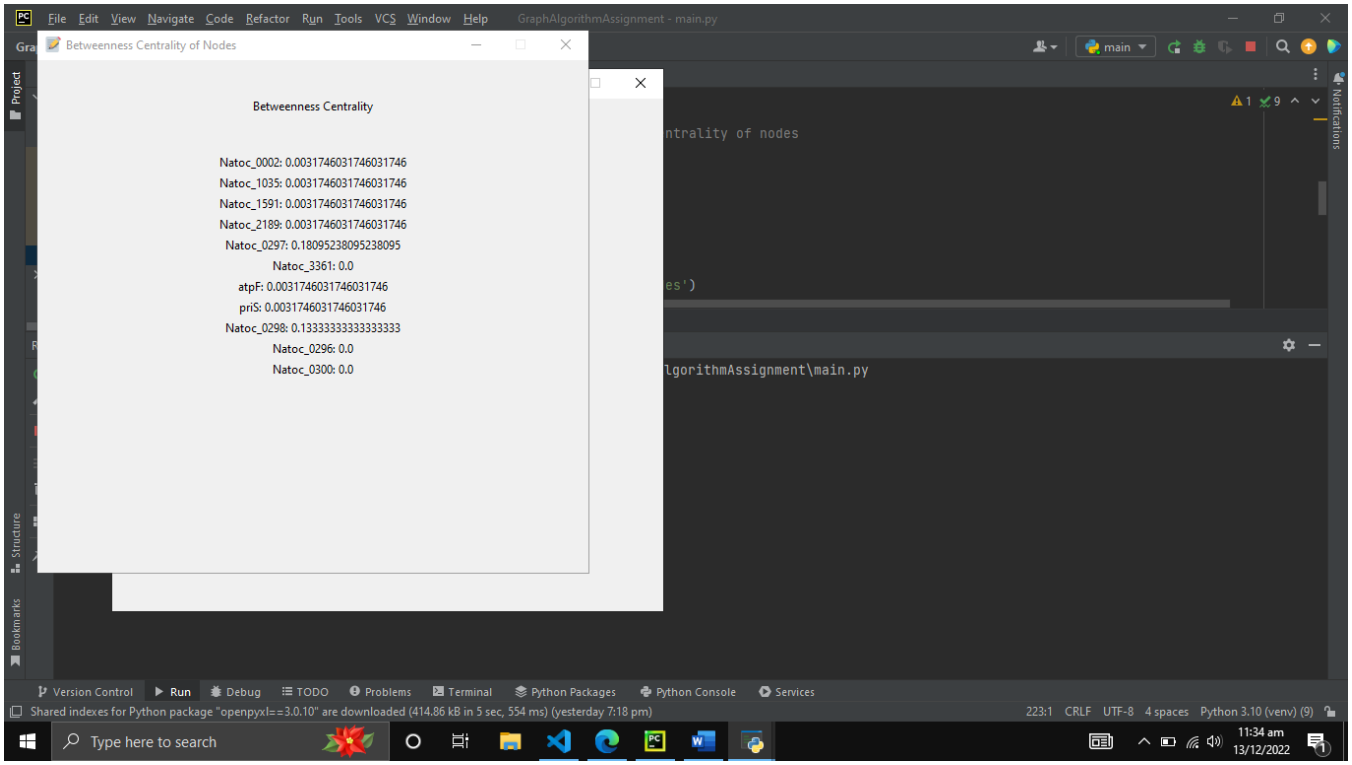


Figure 13: Betweenness Centrality

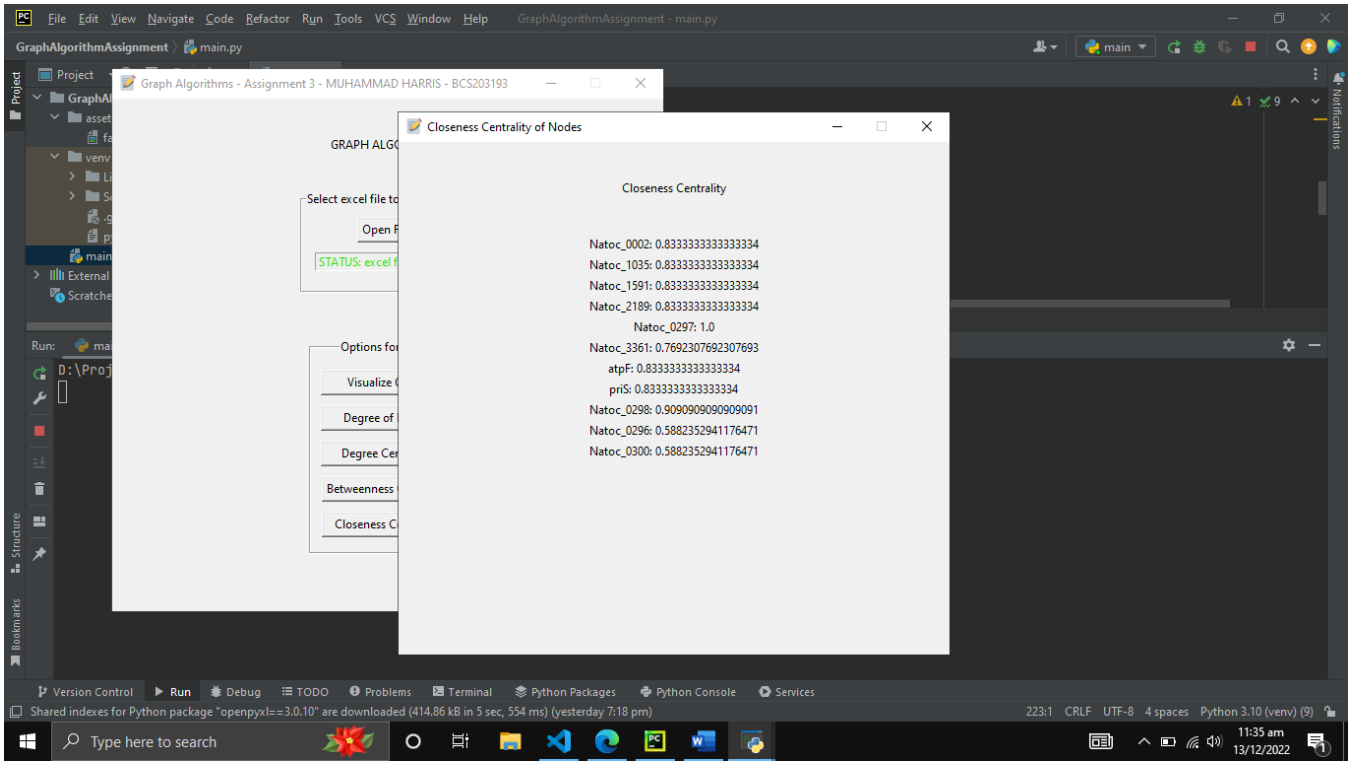


Figure 14: Closeness Centrality