**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

**Дискретна математика**

**Лабораторна робота №3**

**«Графи. Способи представлення графів. Остовні дерева. Пошук найкоротших шляхи»**

Виконав:

студент групи IО-63

Братун Андрій

Залікова книжка № 6305

Перевірив Новотарський М. А.

Київ 2016 р.

**Мета:** Вивчення властивостей графів, способів їх представлення та основних алгоритмів на графах.

**Загальне завдання:**

створити програму, яка реалізує один з алгоритмів на графах.

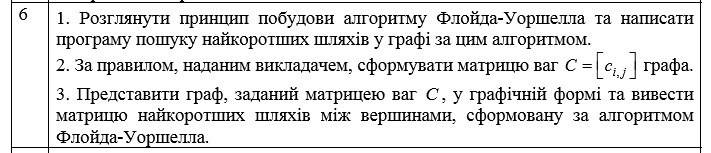
**Варіант виразу відповідно до індивідуального завдання – 6**

Граф G задають множиною точок або вершин x1, x2, ..., хn (яку позначають через X) і множиною ліній або ребер a1, a2, ..., (яку позначають символом А), що з'єднують між собою всі або частину цих точок. Таким чином, граф G повністю задають (і позначають) парою (X, А).

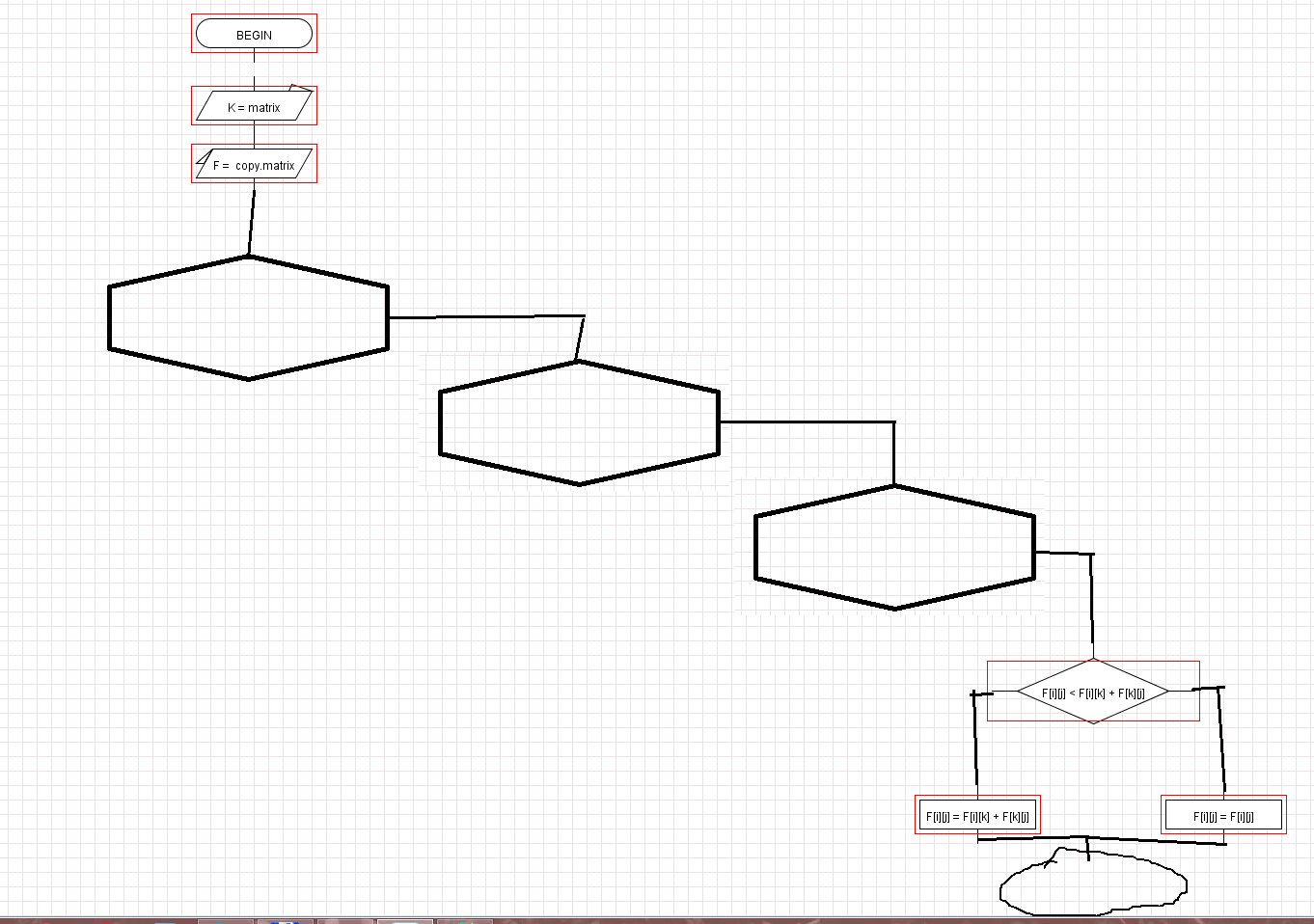
Якщо ребра з множини А орієнтовані, що зазвичай показується стрілкою, то їх називають дугами, і граф з такими ребрами називають орієнтованим графом.

Інший, вживаний частіше, опис орієнтованого графа G полягає у задаванні множини вершин Х і відповідності Г, яка показує, як між собою пов'язані вершини. Відповідність Г називають відображенням множини Х в Х, а граф в цьому випадку позначають парою G

**Завдання:**

****

**Блок-схеми, які відповідають алгоритмам, що використані в лабораторній роботі.**

****

END

i := 1…len(K)

j := 1…len(K)

k := 1…len(K)

**Роздруківка того фрагменту тексту програми, який написаний індивідуально.**

**Module Main**

**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
**import** Create\_matrix  
**import** Create\_graph  
**import** floyd  
  
root = Tk()  
root.title("Main Window")  
  
#root.maxsize(width=320, height=190)  
#root.minsize(width=320, height=190)  
  
  
# My uniq task  
NZK = 6305  
number = NZK % 10 + 1  
task = "Personal task number - " + str(number)  
  
  
# #####################MAINLOOOOOOOP################  
  
# Info about student  
lab1 = Label(root, text='Create by Andrew Bratun', font='arial 20')  
lab2 = Label(root, text='My number NZK - 6305', font='arial 20')  
lab4 = Label(root, text=task, font='arial 20')  
but1 = Button(root, text='Set the matrix of weights', font='arial 20')  
but1.bind("<Button-1>", Create\_matrix.create\_window\_2)  
but2 = Button(root, text='See the graph', font='arial 20')  
but2.bind("<Button-1>", Create\_graph.created\_graph)  
but3 = Button(root, text='Shortest ways', font='arial 20')  
but3.bind("<Button-1>", floyd.floyd)  
  
# Info about student  
lab1.grid(row=1, column=1, sticky=W+E+N+S, pady=5, padx=5)  
lab2.grid(row=2, column=1, sticky=W+E+N+S, pady=5, padx=5)  
lab4.grid(row=3, column=1, sticky=W+E+N+S, pady=5, padx=5)  
  
but1.grid(row=4, column=1, sticky=W+E+N+S, pady=5, padx=5)  
but2.grid(row=5, column=1, sticky=W+E+N+S, pady=5, padx=5)  
but3.grid(row=6, column=1, sticky=W+E+N+S, pady=5, padx=5)  
  
# Menu  
m = Menu(root)  
root.config(menu=m)  
fm = Menu(m)  
m.add\_cascade(label="Windows", menu=fm)  
fm.add\_command(label="Window№2", )  
fm.add\_command(label="Window№3", )  
fm.add\_command(label="Window№4", )  
  
  
'''def exitwindow():  
 if messagebox.askokcancel("Quit", "Do you really wish to quit?"):  
 root.destroy()  
root.protocol("WM\_DELETE\_WINDOW", exitwindow)  
'''  
  
# Xz  
root.mainloop()

**module Create\_graph**

**from** tkinter **import** \*  
**import** networkx **as** nwx, matplotlib.pyplot **as** plt  
**import** Create\_matrix  
  
  
**def created\_graph**(event):  
 graf1 = nwx.Graph()  
 edges = []  
 **for** i **in** Create\_matrix.value:  
 edges.append(i[:2])  
 **for** i **in** edges:  
 graf1.add\_nodes\_from(i)  
 **for** i **in** edges:  
 graf1.add\_edges\_from(edges)  
 nwx.draw(graf1, pos=nwx.shell\_layout(graf1), alpha=0.6, edge\_color='b', font\_size=20,  
 node\_size=700, arrows=**True**, with\_labels=**True**)  
 plt.show()

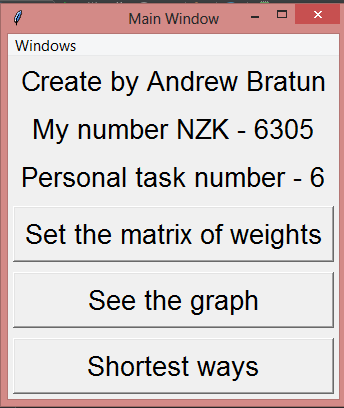
**module Create\_matrix**

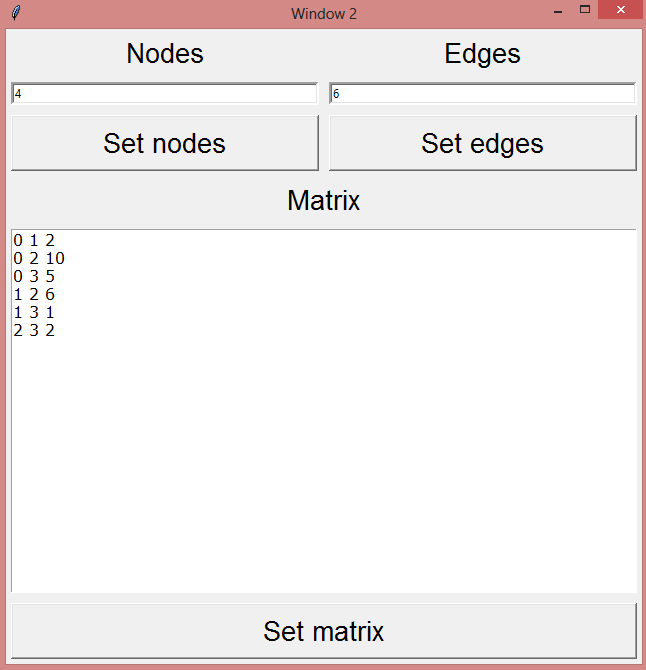
**from** tkinter **import** \*  
  
**def create\_window\_2**(event):  
 #Create Toplevel for window 2  
 window2 = Toplevel()  
 window2.title("Window 2")  
 #window2.maxsize(width=475,height=290)  
 #window2.minsize(width=475,height=290)  
  
  
  
 **def get\_node**(event):  
 **global** N  
 N = int(ent1.get())  
  
  
 **def get\_edge**(event):  
 **global** M  
 M = int(ent2.get())  
  
 **def get\_matrix**(event):  
 **global** weight  
 weight2 = []  
 **global** value  
 value = []  
 weight = [[float("+inf")] \* N **for** j **in** range(N)]  
 **for** i **in** range(N):  
 weight[i][i] = 0  
 **for** i **in** range(M):  
 step = tex1.get(float(i + 1), float(i + 2)).rstrip("\n")  
 weight2.append(step)  
 **for** i **in** weight2:  
 apee = i.split()  
 value.append(apee)  
 **for** i **in** range(len(value)):  
 **for** j **in** range(len(value[i])):  
 value[i][j] = int(value[i][j])  
 **for** i **in** value:  
 weight[i[0]][i[1]] = i[2]  
 weight[i[1]][i[0]] = i[2]  
  
  
  
 lab1 = Label(window2, text='Nodes', font='arial 20')  
 lab2 = Label(window2, text='Edges', font='arial 20')  
 lab3 = Label(window2, text='Matrix', font='arial 20')  
  
  
  
 ent1 = Entry(window2, width=50, bd=3)  
 ent2 = Entry(window2, width=50, bd=3)  
 tex1 = Text(window2, width=10, height=20, font="Verdana 12", wrap=WORD)  
  
 but1 = Button(window2, text='Set nodes', font='arial 20')  
 but1.bind("<Button-1>", get\_node)  
 but2 = Button(window2, text='Set edges', font='arial 20')  
 but2.bind("<Button-1>", get\_edge)  
 but3 = Button(window2, text='Set matrix', font='arial 20')  
 but3.bind("<Button-1>", get\_matrix)  
  
 lab1.grid(row=1, column=1, sticky=W + E + N + S, pady=5, padx=5)  
 lab2.grid(row=1, column=2, sticky=W + E + N + S, pady=5, padx=5)  
 lab3.grid(row=4, column=1, columnspan=2, sticky=W + E + N + S, pady=5, padx=5)  
  
 ent1.grid(row=2, column=1, sticky=W + E + N + S, pady=5, padx=5)  
 ent2.grid(row=2, column=2, sticky=W + E + N + S, pady=5, padx=5)  
 tex1.grid(row=5, column=1, columnspan=2, sticky=W + E + N + S, pady=5, padx=5)  
  
 but1.grid(row=3, column=1, sticky=W + E + N + S, pady=5, padx=5)  
 but2.grid(row=3, column=2, sticky=W + E + N + S, pady=5, padx=5)  
 but3.grid(row=6, column=1, columnspan=2, sticky=W + E + N + S, pady=5, padx=5)

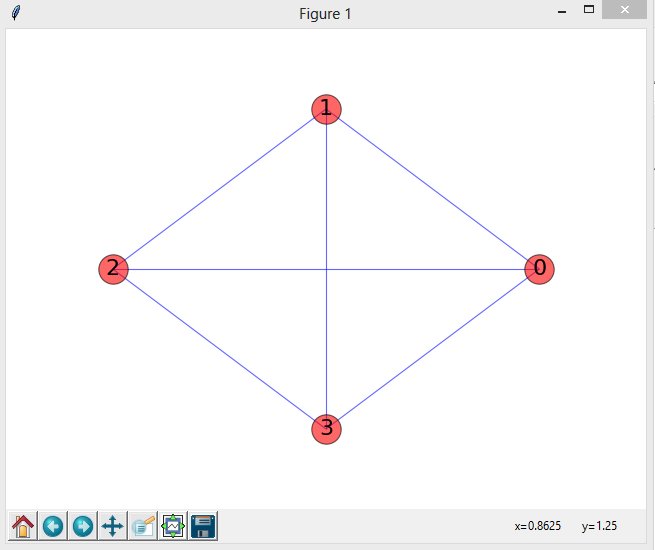
**Module floyd**

**import** Create\_matrix  
**from** copy **import** deepcopy  
**from** tkinter **import** \*  
  
**def floyd**(event):  
  
  
 top = Toplevel(height=500, width=50, relief=GROOVE)  
 top.title("Matrix")  
  
 K = len(Create\_matrix.weight)  
 F = deepcopy(Create\_matrix.weight)  
 **for** k **in** range(K):  
 **for** i **in** range(K):  
 **for** j **in** range(K):  
 F[i][j] = min(F[i][j], F[i][k] + F[k][j])  
  
 lab4 = Label(top, text='N', font='arial 20')  
 lab4.grid(row=0, column=0)  
  
  
 **for** i **in** range(len(F)):  
 lab1 = Label(top, text=i, font='arial 20')  
 lab2 = Label(top, text=i, font='arial 20')  
 lab1.grid(row=i + 1, column=0)  
 lab2.grid(row=0, column=i + 1)  
 **for** i **in** range(len(F)):  
 f = 1  
 **for** j **in** F[i]:  
 lab1 = Label(top, text=j, font='arial 20')  
 lab1.grid(row=i+1, column=f)  
 f += 1

**Роздруківка результатів виконання програми з контрольним прикладом**

****

****

****

2

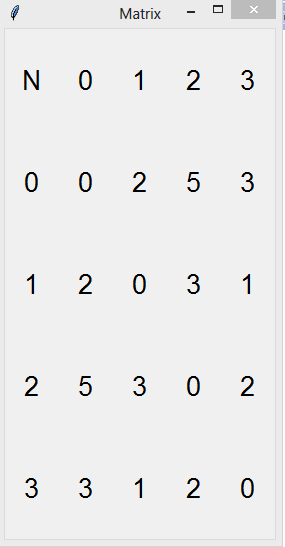
1

6

5

10

2

****

**Аналіз результатів та висновки**

Протягом виконання роботи я вивчив основні методи роботи з графами, вивчив їх види та алгоритми пошуку шляхів, закріпив занання GUI в модулі tkinter в Python.

Складним завдання виявилося від лагодження програми на етапі розробки, оскільки виникпало багато логічних помилок які найскладніше виявити, проте завдання було цікавим, розробка своєї програми з інтерфейсом мені сподобалася. В ході розробки програми виникали деякі питання, але все вдалося знайти за допомогою додаткової інформації в інтернеті.