**Національний технічний університет України**

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

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

**Дискретна математика**

**Лабораторна робота №2**

**«Бінарні відношення та їх основні властивості, операції над відношеннями»**

Виконав:

студент групи IО-63

Братун Андрій

Залікова книжка № 6305

Перевірив Новотарський М. А.

Київ 2016 р.

**Мета:** вивчити основні властивості бінарних відношень та оволодіти операціями над бінарними відношеннями.

**Загальне завдання:**

1. Написати в окремому модулі функцію для формування несуперечливих бінарних відношень.

2. Написати в окремому модулі функції виконання логічних операцій над бінарними відношеннями.

3. Пояснити правило формування несуперечливих відношень відповідно до Вашого варіанту

**Варіант виразу відповідно до індивідуального завдання – 9**

**Короткі теоретичні відомості:**

Упорядкована пара предметів – це сукупність, що складається із двох предметів, розташованих у деякому певному порядку.

При цьому впорядкована пара має наступні властивості:

а) для будь-яких двох предметів x і y існує об'єкт, який можна позначити як x, , y названий упорядкованою парою;

б) якщо x, y і u, – v упорядковані пари, то x, y = u,v тоді і тільки тоді, коли x = u , y = v . При цьому x будемо називати першою координатою, а y – другою координатою впорядкованої пари x, . y

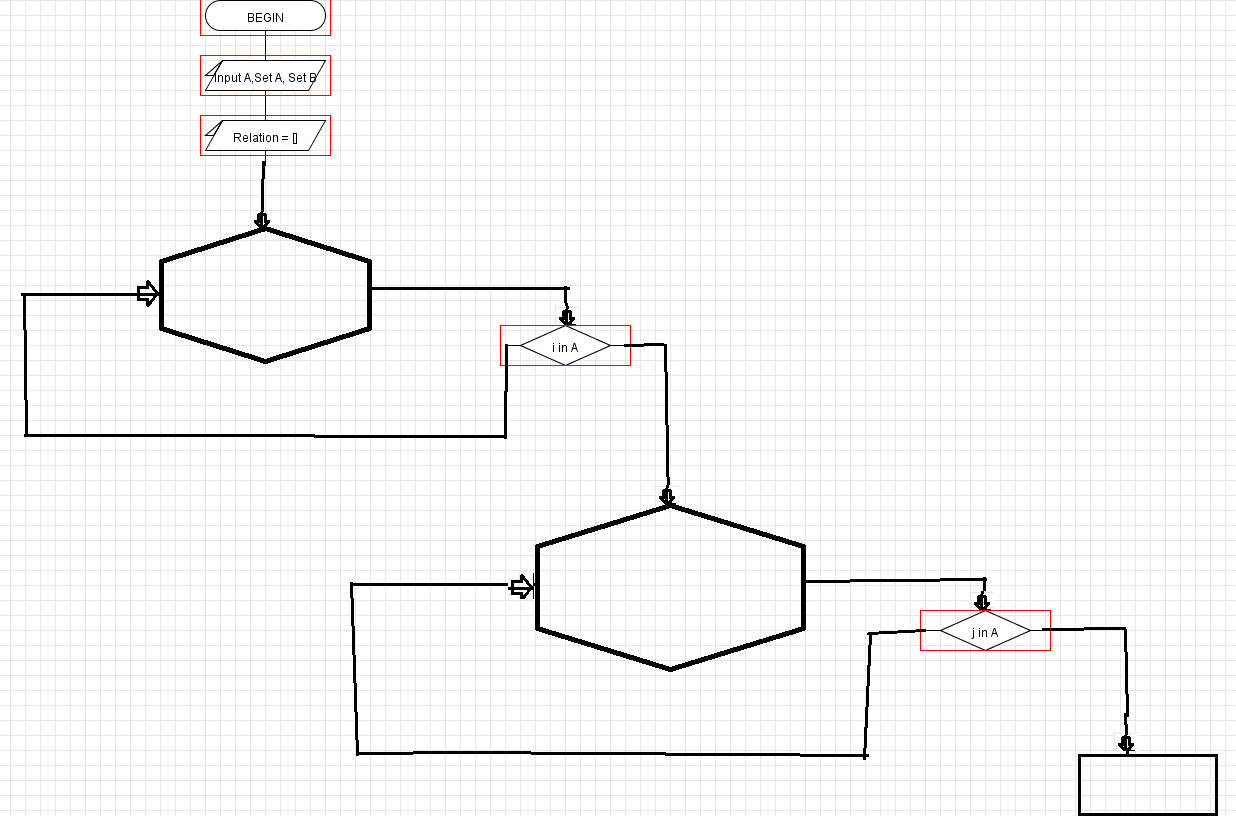
Бінарним (або двомісним) відношенням R називають підмножину впорядкованих пар, тобто множину, кожен елемент якої є впорядкованою парою. Якщо R є деяким відношенням, це записують як x, y ∈ R або xRy .

Один з типів відношень − це множина всіх таких пар x, , y що x є елементом деякої фіксованої множини X , а y − елементом деякої фіксованої множини Y . Таке відношення називають прямим або декартовим добутком. Декартовим добутком X ×Y множин X і Y є множина { x, . y x ∈ X, y ∈Y} При цьому множину X називають областю визначення відношення R , а Y – його областю значень:

D( ) R = {x x, y ∈ R}; E( ) R = {y x, y ∈ R} Бінарним відношенням R називають підмножину пар x, y ∈ R прямого добутку X ×Y , тобто R ⊆ X ×Y .

У силу визначення бінарних відношень, як спосіб їх задавання можуть бути використані будь-які способи задавання множин. Відношення, визначені на скінченних множинах, зазвичай задають:

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

****

Rel.app(j)

No

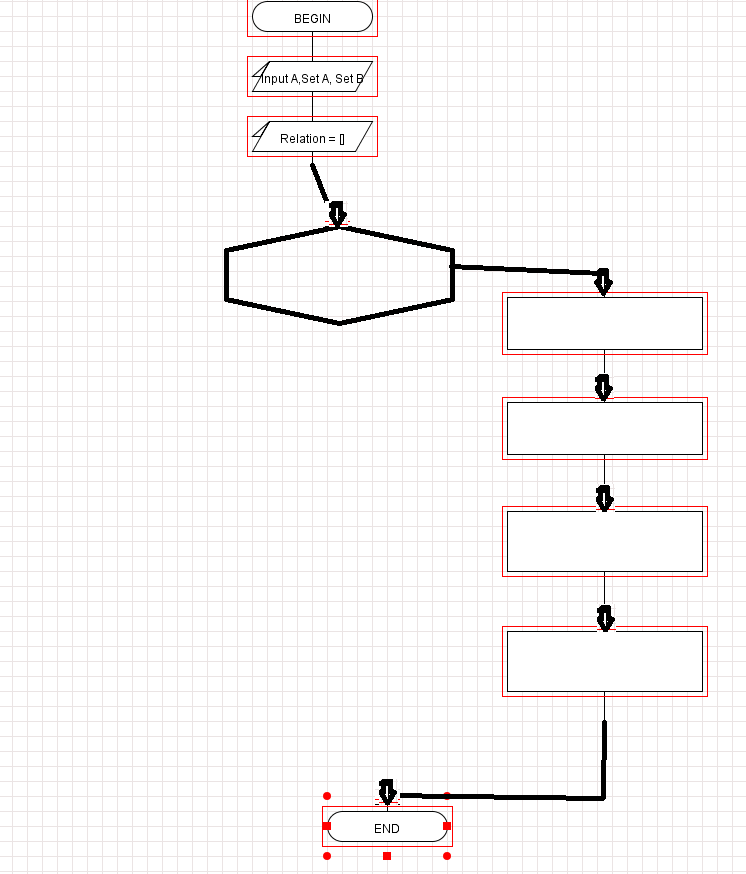
Yes

j := 1…len(B)

i := 1…len(A)

Yes

No

****

Relation.append(p)

P = (sister1,sister2)

Sister2 = random(set A)

Sister1 = random(set B)

i:=1…len(set A)

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

**Module Main**

**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
**import** window2,window3,window4  
  
root = Tk()  
root.title("Main Window")  
  
root.maxsize(width=320, height=190)  
root.minsize(width=320, height=190)  
  
  
# My uniq task  
Num = 5  
G = 63  
number = (Num + G % 60) % 30 + 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='Group number - IO - 63', font='arial 20')  
lab3 = Label(root, text='My number in group - 5', font='arial 20')  
lab4 = Label(root, text=task, font='arial 20')  
  
  
# 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)  
lab3.grid(row=3, column=1, sticky=W+E+N+S, pady=5, padx=5)  
lab4.grid(row=4, 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", command=window2.create\_window\_2)  
fm.add\_command(label="Window№3", command=window3.create\_window\_3)  
fm.add\_command(label="Window№4", command=window4.create\_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 window2**

**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
  
  
# Create window 2  
**def create\_window\_2**():  
  
 #Create Toplevel for window 2  
 window2 = Toplevel()  
 window2.title("Window 2")  
 #window2.maxsize(width=475,height=290)  
 #window2.minsize(width=475,height=290)  
  
  
 #########################FUNCTIONS####################################  
  
 **def save\_set\_A**():  
 **with** open(r"Set A.txt", "w", encoding="UTF-8") **as** f:  
 written = listbox3.get(0,END)  
 msg = messagebox.showinfo('Save set A', 'Set A was saved')  
 **for** i **in** written:  
 f.write(i + " ")  
  
 **def save\_set\_B**():  
 **with** open(r"Set B.txt", "w", encoding="UTF-8") **as** f:  
 written = listbox4.get(0,END)  
 msg = messagebox.showinfo('Save set B', 'Set B was saved')  
 **for** i **in** written:  
 f.write(i + " ")  
  
  
 #See set A  
 **def see\_set\_A**():  
 **try**:  
 **with** open(r"Set A.txt", "r", encoding="UTF-8") **as** f:  
 p = f.read()  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("Set A")  
 tex = Text(top,width=180, height=20,font="Verdana 12",wrap=WORD)  
 tex.insert(END,p)  
 tex.pack(fill=BOTH,expand=**True**)  
 **except**:  
 msg = messagebox.showinfo('Error', 'File not found')  
  
 **def see\_set\_B**():  
 **try**:  
 **with** open(r"Set B.txt", "r", encoding="UTF-8") **as** f:  
 p = f.read()  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("Set B")  
 tex = Text(top,width=180, height=20,font="Verdana 12",wrap=WORD)  
 tex.insert(END,p)  
 tex.pack(fill=BOTH,expand=**True**)  
 **except**:  
 msg = messagebox.showinfo('Error', 'File not found')  
  
 **def delete\_A**():  
 listbox3.delete(0,END)  
  
 **def delete\_B**():  
 listbox4.delete(0,END)  
  
  
  
 #labels  
 lab1 = Label(window2, text='Women', font='arial 20')  
 lab2 = Label(window2, text='Men', font='arial 20')  
 lab3 = Label(window2, text='Set A', font='arial 20')  
 lab4 = Label(window2, text='Set B', font='arial 20')  
  
  
 #Listbox  
 listbox1 = Listbox(window2, height=5,width=15, font=("Arial", 16), selectmode=EXTENDED)  
 listbox2 = Listbox(window2, height=5,width=15, font=("Arial", 16), selectmode=EXTENDED)  
 listbox3 = Listbox(window2, height=5,width=15, font=("Arial", 16), selectmode=EXTENDED)  
 listbox4 = Listbox(window2, height=5,width=15, font=("Arial", 16), selectmode=EXTENDED)  
  
  
 #AddToListBox  
 list1=["Вікторія","Світлана","Марія","Анна","Дарина","Катерина","Людмила","Зоя","Аліна","Олена","Юлія","Лариса","Анастасія","Антоніна","Оксана","Галина","Тетяна","Василина","Валентина","Інна"]  
 list2=["Андрій","Петро","Ігор","Віктор","Антон","Євген","Дмитро","Вадим","Олександр","Віталій","Богдан","Павел","Сергій","Микола","Володимир","Юрій","Олег","Михайло","Семен","Чіпка"]  
 **for** i **in** list1:  
 listbox1.insert(END, i)  
 **for** i **in** list2:  
 listbox2.insert(END, i)  
  
  
 #RadioButton  
 var1 = IntVar()  
 var2 = IntVar()  
 var1.set(1)  
 var2.set(3)  
 rbutton1 = Radiobutton(window2, text='Set A', variable=var1, value=1)  
 rbutton2 = Radiobutton(window2, text='Set B', variable=var1, value=2)  
 rbutton3 = Radiobutton(window2, text='Women', variable=var2, value=3)  
 rbutton4 = Radiobutton(window2, text='Men', variable=var2, value=4)  
  
 **def callback**():  
 **if** var1.get() == 1:  
 **if** var2.get() == 3:  
 value = listbox1.get(ACTIVE)  
 listbox1.delete(ACTIVE)  
 listbox3.insert(END, value)  
 **else**:  
 value = listbox2.get(ACTIVE)  
 listbox2.delete(ACTIVE)  
 listbox3.insert(END, value)  
 **else**:  
 **if** var2.get() == 3:  
 value = listbox1.get(ACTIVE)  
 listbox1.delete(ACTIVE)  
 listbox4.insert(END, value)  
 **else**:  
 value = listbox2.get(ACTIVE)  
 listbox2.delete(ACTIVE)  
 listbox4.insert(END, value)  
  
  
  
  
 #Button  
 but1 = Button(window2, text="Insert in set :", command=callback, width=10, font=("Arial", 20))  
  
  
 #Button  
 but2 = Button(window2, text="Save A in file", command=save\_set\_A, width=10, font=("Arial", 20))  
 but3 = Button(window2, text="Read A", command=see\_set\_A, width=10, font=("Arial", 20))  
 but4 = Button(window2, text="Clear A", command=delete\_A, width=10, font=("Arial", 20))  
 but5 = Button(window2, text="Save B in file", command=save\_set\_B, width=10, font=("Arial", 20))  
 but6 = Button(window2, text="Read B", command=see\_set\_B, width=10, font=("Arial", 20))  
 but7 = Button(window2, text="Clear B", command=delete\_B, width=10, font=("Arial", 20))  
  
 #labels  
 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=5, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 lab4.grid(row=5, column=2, sticky=W+E+N+S, pady=5, padx=5)  
  
  
 #Listbox  
 listbox1.grid(row=2, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 listbox2.grid(row=2, column=2, sticky=W+E+N+S, pady=5, padx=5)  
 listbox3.grid(row=6, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 listbox4.grid(row=6, column=2, sticky=W+E+N+S, pady=5, padx=5)  
  
  
 #Button  
 but1.grid(row=3, column=1, sticky=W+E+N+S, pady=5, padx=5)  
  
  
 #RadioButton  
 rbutton1.grid(row=3, column=2, sticky=W)  
 rbutton2.grid(row=3, column=2, sticky=E)  
 rbutton3.grid(row=4, column=2, sticky=W)  
 rbutton4.grid(row=4, column=2, sticky=E)  
  
  
 #Button  
 but2.grid(row=7, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 but3.grid(row=8, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 but4.grid(row=9, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 but5.grid(row=7, column=2, sticky=W+E+N+S, pady=5, padx=5)  
 but6.grid(row=8, column=2, sticky=W+E+N+S, pady=5, padx=5)  
 but7.grid(row=9, column=2, sticky=W+E+N+S, pady=5, padx=5)

**module window3**

**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
**import** random  
**import** copy  
relation\_S\_plus\_R = []  
**def create\_window\_3**():  
  
 #Create Toplevel for window 2  
 window3 = Toplevel()  
 window3.title("Window 3")  
 '''  
 window2.maxsize(width=475,height=290)  
 window2.minsize(width=475,height=290)  
 '''  
  
 list1=["Вікторія","Світлана","Марія","Анна","Дарина","Катерина","Людмила","Зоя","Аліна","Олена","Юлія","Лариса","Анастасія","Антоніна","Оксана","Галина","Тетяна","Василина","Валентина","Інна"]  
 list2=["Андрій","Петро","Ігор","Віктор","Антон","Євген","Дмитро","Вадим","Олександр","Віталій","Богдан","Павел","Сергій","Микола","Володимир","Юрій","Олег","Михайло","Семен","Чіпка"]  
  
 #Labels  
 lab1 = Label(window3, text='Set A', font='arial 20')  
 lab2 = Label(window3, text='Set B', font='arial 20')  
  
 #Listbox  
 listbox1 = Listbox(window3, height=5,width=15, font=("Arial", 16), selectmode=EXTENDED)  
 listbox2 = Listbox(window3, height=5,width=15, font=("Arial", 16), selectmode=EXTENDED)  
  
  
  
  
 #Labels  
 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)  
  
  
 #Listbox  
 listbox1.grid(row=2, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 listbox2.grid(row=2, column=2, sticky=W+E+N+S, pady=5, padx=5)  
  
  
  
  
  
 f1 = open(r"Set A.txt", "r", encoding="UTF-8")  
 set\_A = f1.read().split(" ")  
 f2 = open(r"Set B.txt", "r", encoding="UTF-8")  
 set\_B = f2.read().split(" ")  
  
  
 ######################################RELATION S####################################################################  
 set\_B\_use = copy.copy(set\_B)  
 relation\_S = []  
 **try**:  
 **for** i **in** set\_A[:-1]:  
 **if** i **in** list1:  
 deti = random.randint(1,3)  
 **for** j **in** range(1,deti+1):  
 rebenok = random.choice(set\_B\_use[:-1])  
 **if** rebenok **not in** list1:  
 **del** set\_B\_use[set\_B\_use.index(rebenok)]  
 printer = i,rebenok  
  
 relation\_S.append(printer)  
 **else**:**continue  
 else**:**continue  
 except**:print('')  
  
 all1 = []  
 **for** i **in** set\_A[:-1]:  
 **for** j **in** set\_B[:-1]:  
 p = i,j  
 all1.append(p)  
 topp = []  
 **for** i **in** relation\_S:  
 topp.append(i[0])  
 leftt = []  
 **for** j **in** relation\_S:  
 leftt.append(j[1])  
 score = []  
 **for** i **in** all1:  
 **if** i **in** relation\_S:  
 score.append(1)  
 **else**:score.append(0)  
  
  
  
 r1 = len(set\_A[:-1])  
 c1 = len(set\_B[:-1])  
 **with** open(r"Relation S.txt", "w", encoding="UTF-8") **as** f:  
 **for** i **in** relation\_S:  
 f.write(str(i))  
  
  
  
  
 **def callback**():  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("Relation S")  
 calc = 0  
 lab3 = Label(top, text="Relation S", font='arial 14')  
 lab3.grid(row=0, column=0)  
 z = 1  
 f = 1  
 **for** m **in** list(set\_A[:-1]):  
 lb = Label(top, text=m, font='arial 14')  
 lb.grid(row=z, column=0)  
 z += 1  
 **for** n **in** list(set\_B[:-1]):  
 lb = Label(top, text=n, font='arial 14')  
 lb.grid(row=0, column=f)  
 f += 1  
 **for** r **in** range(r1):  
 **for** c **in** range(c1):  
 lb = Label(top, text=score[calc], font='arial 14')  
 lb.grid(row=r+1, column=c+1)  
 calc += 1  
  
 but1 = Button(window3, text="Relation A",command=callback, width=10, font=("Arial", 20))  
 but1.grid(row=3, column=1, sticky=W+E+N+S, pady=5, padx=5)  
  
  
 ######################################RELATION R####################################################################  
 relation\_R = []  
 women\_in\_A = []  
 women\_in\_B = []  
 set\_A\_use1 = copy.copy(set\_A)  
 set\_B\_use1 = copy.copy(set\_B)  
  
 **for** i **in** set\_A\_use1[:-1]:  
 **if** i **in** list1:  
 women\_in\_A.append(i)  
 **for** j **in** set\_B\_use1[:-1]:  
 **if** j **in** list1:  
 women\_in\_B.append(j)  
  
 **if** len(women\_in\_A) >= len(women\_in\_B):  
 **for** i **in** range(len(women\_in\_B)):  
 sister1 = random.choice(women\_in\_B)  
 sister2 = random.choice(women\_in\_A)  
 p = (sister2,sister1)  
 relation\_R.append(p)  
 **del** women\_in\_B[women\_in\_B.index(sister1)]  
 **del** women\_in\_A[women\_in\_A.index(sister2)]  
 **else**:  
 **for** i **in** range(len(women\_in\_A)):  
 sister1 = random.choice(women\_in\_B)  
 sister2 = random.choice(women\_in\_A)  
 p = (sister2,sister1)  
 relation\_R.append(p)  
 **del** women\_in\_B[women\_in\_B.index(sister1)]  
 **del** women\_in\_A[women\_in\_A.index(sister2)]  
  
  
  
  
 **with** open(r"Relation R.txt", "w", encoding="UTF-8") **as** f:  
 **for** i **in** relation\_R:  
 f.write(str(i) + " ")  
  
  
 all2 = []  
 **for** i **in** set\_A[:-1]:  
 **for** j **in** set\_B[:-1]:  
 p = i,j  
 all2.append(p)  
 score1 = []  
 **for** i **in** all2:  
 **if** i **in** relation\_R:  
 score1.append(1)  
 **else**:score1.append(0)  
  
  
 **def callback1**():  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("Relation R")  
 calc = 0  
 lab3 = Label(top, text="Relation R", font='arial 14')  
 lab3.grid(row=0, column=0)  
 z = 1  
 f = 1  
 **for** m **in** list(set\_A[:-1]):  
 lb = Label(top, text=m, font='arial 14')  
 lb.grid(row=z, column=0)  
 z += 1  
 **for** n **in** list(set\_B[:-1]):  
 lb = Label(top, text=n, font='arial 14')  
 lb.grid(row=0, column=f)  
 f += 1  
 **for** r **in** range(r1):  
 **for** c **in** range(c1):  
 lb = Label(top, text=score1[calc], font='arial 14')  
 lb.grid(row=r+1, column=c+1)  
 calc += 1  
  
 but1 = Button(window3, text="Relation B",command=callback1, width=10, font=("Arial", 20))  
 but1.grid(row=3, column=2, sticky=W+E+N+S, pady=5, padx=5)  
  
 **global** relation\_S\_plus\_R  
 relation\_S\_plus\_R = relation\_S + relation\_R  
 **global** relation\_S\_on\_R  
 relation\_S\_on\_R = []  
 **for** i **in** relation\_S:  
 **if** i **in** relation\_R:  
 relation\_S\_on\_R.append(i)  
  
 **global** relation\_R\_diff\_S  
 relation\_R\_diff\_S = []  
 **for** i **in** relation\_R:  
 **if** i **not in** relation\_S:  
 relation\_R\_diff\_S.append(i)  
  
 **global** relation\_U\_diff\_R  
 relation\_U\_diff\_R = []  
 **for** i **in** all1:  
 **if** i **not in** relation\_R:  
 relation\_U\_diff\_R.append(i)  
  
 **global** s\_tra  
 s\_tra = []  
 s\_tra = relation\_S  
  
 **for** i **in** set\_A[:-1]:  
 listbox1.insert(END, i)  
 **for** i **in** set\_B[:-1]:  
 listbox2.insert(END, i)

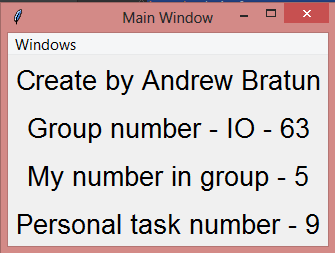
**Module window4**

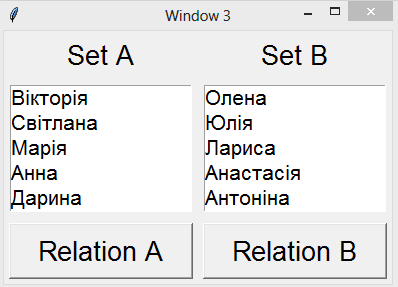
**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
**import** random  
**import** copy  
**import** func\_win\_4  
  
  
**def create\_window\_4**():  
  
 #Create Toplevel for window 2  
 window4 = Toplevel()  
 window4.title("Window 4")  
 #window2.maxsize(width=475,height=290)  
 #window2.minsize(width=475,height=290)  
  
  
 #########################FUNCTIONS####################################  
 but1 = Button(window4, text="R ∪ S ", command=func\_win\_4.R\_plus\_S, width=10, font=("Arial", 20))  
 but2 = Button(window4, text="R ∩ S", command=func\_win\_4.R\_on\_S, width=10, font=("Arial", 20))  
 but3 = Button(window4, text="R \ S", command=func\_win\_4.R\_diff\_S, width=10, font=("Arial", 20))  
 but4 = Button(window4, text="U \ R", command=func\_win\_4.U\_diff\_R, width=10, font=("Arial", 20))  
 but5 = Button(window4, text="s^(-1)", command=func\_win\_4.S\_tra, width=10, font=("Arial", 20))  
  
  
 but1.grid(row=0, column=0, sticky=W+E+N+S, pady=5, padx=5)  
 but2.grid(row=0, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 but3.grid(row=1, column=0, sticky=W+E+N+S, pady=5, padx=5)  
 but4.grid(row=1, column=1, sticky=W+E+N+S, pady=5, padx=5)  
 but5.grid(row=2, column=0,columnspan=2, sticky=W+E+N+S, pady=5, padx=5)

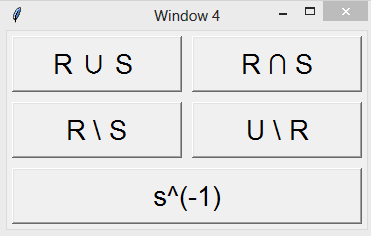
**module func\_win\_4**

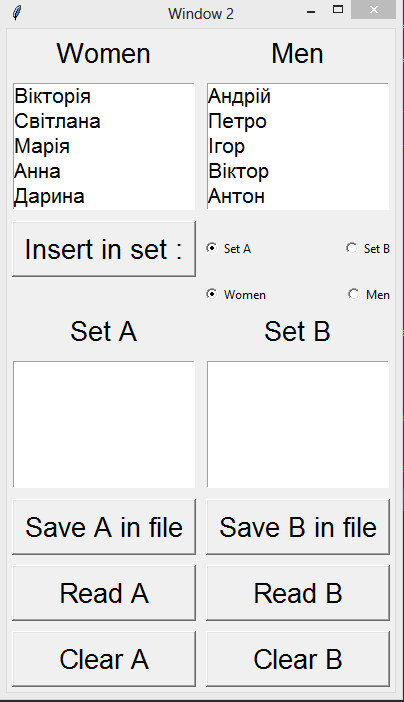
**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
**import** random  
**import** copy  
**import** window3  
  
  
**with** open(r"Relation R.txt", "r", encoding="UTF-8") **as** f:  
 relation\_R = f.read()  
**with** open(r"Relation S.txt", "r", encoding="UTF-8") **as** f:  
 relation\_S = f.read()  
  
f1 = open(r"Set A.txt", "r", encoding="UTF-8")  
set\_A = f1.read().split(" ")  
f2 = open(r"Set B.txt", "r", encoding="UTF-8")  
set\_B = f2.read().split(" ")  
**def R\_plus\_S**():  
 all1 = []  
 **for** i **in** set\_A[:-1]:  
 **for** j **in** set\_B[:-1]:  
 p = i,j  
 all1.append(p)  
  
 score = []  
 **for** i **in** all1:  
 **if** i **in** window3.relation\_S\_plus\_R:score.append(1)  
 **else**:score.append(0)  
  
 r1 = len(set\_A[:-1])  
 c1 = len(set\_B[:-1])  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("R ∪ S")  
 calc = 0  
 lab3 = Label(top, text="R ∪ S", font='arial 14')  
 lab3.grid(row=0, column=0)  
 z = 1  
 f = 1  
 **for** m **in** list(set\_A[:-1]):  
 lb = Label(top, text=m, font='arial 14')  
 lb.grid(row=z, column=0)  
 z += 1  
 **for** n **in** list(set\_B[:-1]):  
 lb = Label(top, text=n, font='arial 14')  
 lb.grid(row=0, column=f)  
 f += 1  
 **for** r **in** range(r1):  
 **for** c **in** range(c1):  
 lb = Label(top, text=score[calc], font='arial 14')  
 lb.grid(row=r+1, column=c+1)  
 calc += 1  
  
**def R\_on\_S**():  
 all1 = []  
 **for** i **in** set\_A[:-1]:  
 **for** j **in** set\_B[:-1]:  
 p = i,j  
 all1.append(p)  
  
 score = []  
 **for** i **in** all1:  
 **if** i **in** window3.relation\_S\_on\_R:score.append(1)  
 **else**:score.append(0)  
  
 r1 = len(set\_A[:-1])  
 c1 = len(set\_B[:-1])  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("R ∩ S")  
 calc = 0  
 lab3 = Label(top, text="R ∩ S", font='arial 14')  
 lab3.grid(row=0, column=0)  
 z = 1  
 f = 1  
 **for** m **in** list(set\_A[:-1]):  
 lb = Label(top, text=m, font='arial 14')  
 lb.grid(row=z, column=0)  
 z += 1  
 **for** n **in** list(set\_B[:-1]):  
 lb = Label(top, text=n, font='arial 14')  
 lb.grid(row=0, column=f)  
 f += 1  
 **for** r **in** range(r1):  
 **for** c **in** range(c1):  
 lb = Label(top, text=score[calc], font='arial 14')  
 lb.grid(row=r+1, column=c+1)  
 calc += 1  
  
**def R\_diff\_S**():  
 all1 = []  
 **for** i **in** set\_A[:-1]:  
 **for** j **in** set\_B[:-1]:  
 p = i,j  
 all1.append(p)  
  
 score = []  
 **for** i **in** all1:  
 **if** i **in** window3.relation\_R\_diff\_S:score.append(1)  
 **else**:score.append(0)  
  
 r1 = len(set\_A[:-1])  
 c1 = len(set\_B[:-1])  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("R \ S")  
 calc = 0  
 lab3 = Label(top, text="R \ S", font='arial 14')  
 lab3.grid(row=0, column=0)  
 z = 1  
 f = 1  
 **for** m **in** list(set\_A[:-1]):  
 lb = Label(top, text=m, font='arial 14')  
 lb.grid(row=z, column=0)  
 z += 1  
 **for** n **in** list(set\_B[:-1]):  
 lb = Label(top, text=n, font='arial 14')  
 lb.grid(row=0, column=f)  
 f += 1  
 **for** r **in** range(r1):  
 **for** c **in** range(c1):  
 lb = Label(top, text=score[calc], font='arial 14')  
 lb.grid(row=r+1, column=c+1)  
 calc += 1  
  
**def U\_diff\_R**():  
 all1 = []  
 **for** i **in** set\_A[:-1]:  
 **for** j **in** set\_B[:-1]:  
 p = i,j  
 all1.append(p)  
  
 score = []  
 **for** i **in** all1:  
 **if** i **in** window3.relation\_U\_diff\_R:score.append(1)  
 **else**:score.append(0)  
  
 r1 = len(set\_A[:-1])  
 c1 = len(set\_B[:-1])  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("U \ R")  
 calc = 0  
 lab3 = Label(top, text="U \ R", font='arial 14')  
 lab3.grid(row=0, column=0)  
 z = 1  
 f = 1  
 **for** m **in** list(set\_A[:-1]):  
 lb = Label(top, text=m, font='arial 14')  
 lb.grid(row=z, column=0)  
 z += 1  
 **for** n **in** list(set\_B[:-1]):  
 lb = Label(top, text=n, font='arial 14')  
 lb.grid(row=0, column=f)  
 f += 1  
 **for** r **in** range(r1):  
 **for** c **in** range(c1):  
 lb = Label(top, text=score[calc], font='arial 14')  
 lb.grid(row=r+1, column=c+1)  
 calc += 1  
  
  
**def S\_tra**():  
 all1 = []  
 **for** i **in** set\_A[:-1]:  
 **for** j **in** set\_B[:-1]:  
 p = i,j  
 all1.append(p)  
  
 score = []  
 **for** i **in** all1:  
 **if** i **in** window3.s\_tra:score.append(1)  
 **else**:score.append(0)  
  
 r1 = len(set\_A[:-1])  
 c1 = len(set\_B[:-1])  
 top = Toplevel(height=500, width=100, relief = GROOVE)  
 top.title("s^(-1)")  
 calc = 0  
 lab3 = Label(top, text="s^(-1)", font='arial 14')  
 lab3.grid(row=0, column=0)  
 z = 1  
 f = 1  
 **for** m **in** list(set\_A[:-1]):  
 lb = Label(top, text=m, font='arial 14')  
 lb.grid(row=z, column=0)  
 z += 1  
 **for** n **in** list(set\_B[:-1]):  
 lb = Label(top, text=n, font='arial 14')  
 lb.grid(row=0, column=f)  
 f += 1  
 **for** r **in** range(r1):  
 **for** c **in** range(c1):  
 lb = Label(top, text=score[calc], font='arial 14')  
 lb.grid(row=r+1, column=c+1)  
 calc += 1

**Роздруківка результатів виконання програми з контрольним прикладом**

****

****

****

****

**Аналіз результатів та висновки**

Протягом виконання роботи я вивчив основні властивості бінарних відношень та оволодів операціями над бінарними відношеннями, та закріпив занання GUI в модулі tkinter в Python.

Завдання було цікавим, розробка своєї програми з інтерфейсом мені сподобалася. В ході розробки програми виникало багато помилок та незрозумілостей, але все вдалося налагодити за допомогою додаткової інформації в інтернеті.