Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительной техники»

**Отчет**

По лабораторной работе №3

По курсу «Логика и основы алгоритмизации в инженерных задачах»

На тему «Унарные и бинарные операции над графами»

Выполнили студентка гр.20ВВ1

Тюгаева К.А.

Проверили:

Акифьев И. В.

Юрова О. В.

Пенза, 2021

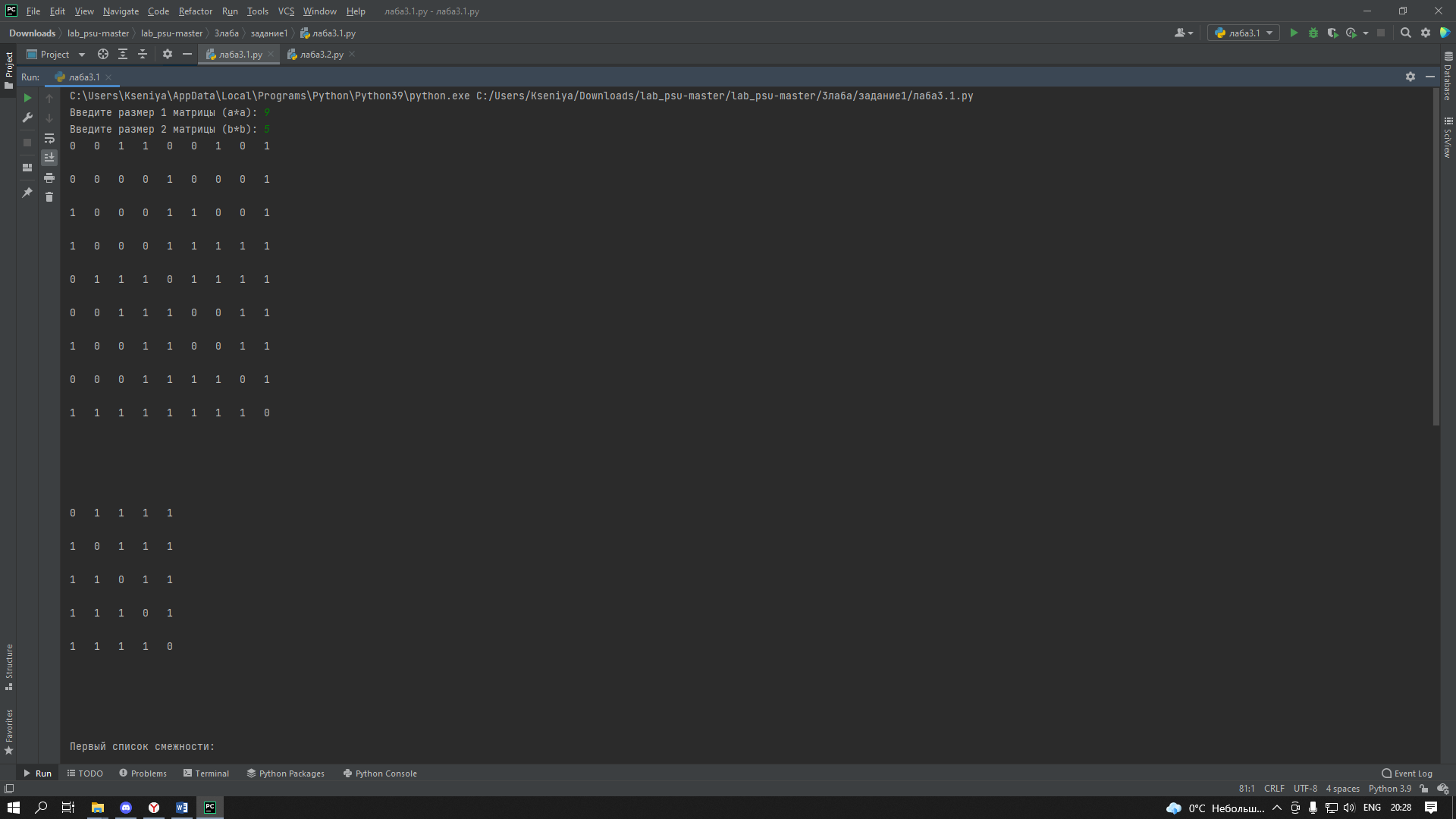
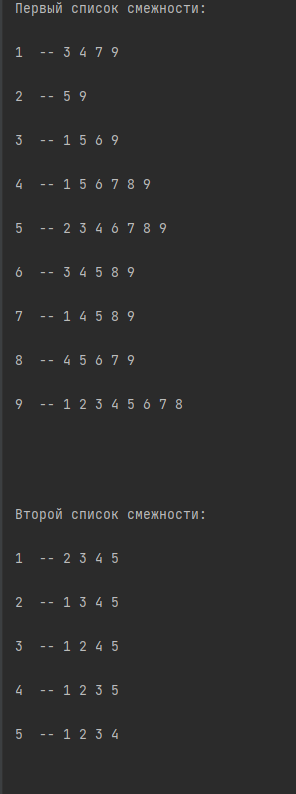
**Цель:** научится производить унарные и бинарные операции с матрицами смежности.

**Задание 1:**

1. Сгенерировала (используя генератор случайных чисел) две матрицы M1, М2 смежности неориентированных помеченных графов G1, G2.

2. \* Для указанных графов преобразовала представление матриц смежности в списки смежности.

Вывела сгенерированные матрицы и полученные списки на экран:

**Задание 2**

1. Для матричной формы представления графов выполнила операцию:

а) отождествления вершин

б) стягивания ребра

в) расщепления вершины

Номера выбираемых для выполнения операции вершин ввела с клавиатуры.

1. \* Для представления графов в виде списков смежности выполнила операцию:

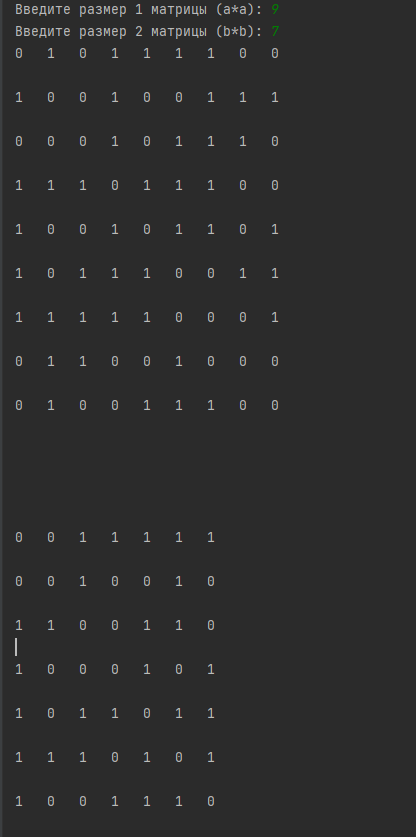
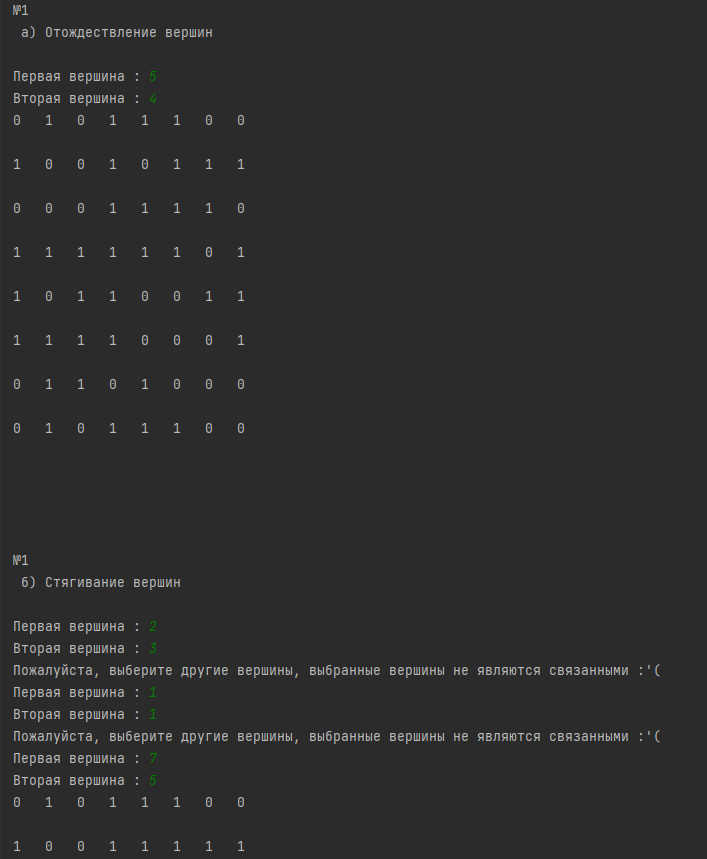
а) отождествления вершин

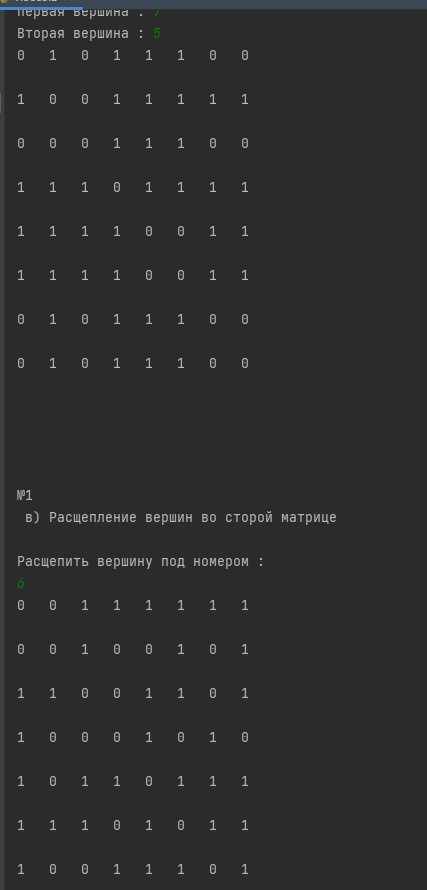
б) стягивания ребра

в) расщепления вершины

Номера выбираемых для выполнения операции вершин ввела с клавиатуры.

Результат выполнения операций:



**Задание 3**

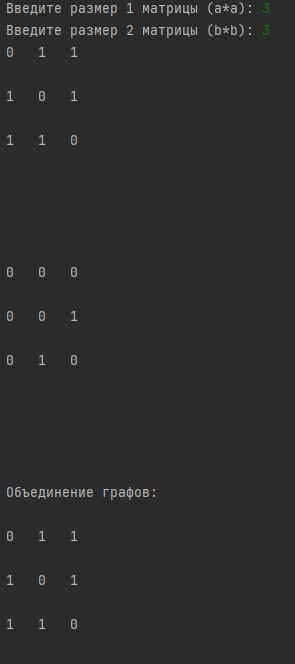
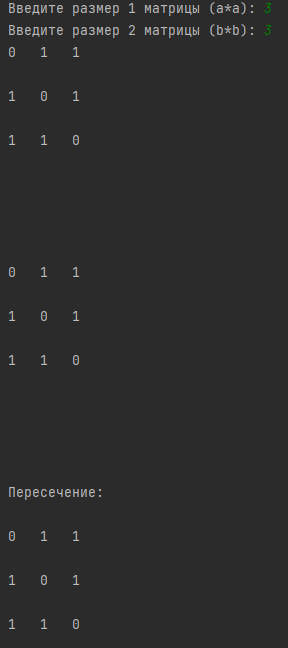
1. Для матричной формы представления графов выполнила операцию:

а) объединения *G* = *G*1  *G*2

б) пересечения *G* = *G*1  *G*2

в) кольцевой суммы *G* = *G*1  *G*2

Результат выполнения операции:

**Листинг задания 1:**

import random  
  
def CreateTwoMatrix():  
 global M1  
 global M2  
 global N1  
 global N2  
 M1 = []  
 M2 = []  
 N1 = int(input("Введите размер 1 матрицы (а\*а): "))  
 N2 = int(input("Введите размер 2 матрицы (b\*b): "))  
 chance = [0, 0, 0, 1, 1, 1, 1, 1, 1, 1]  
 # Шанс 1 - 70%  
 # Шанс 0 - 30%  
  
 # Создание 2 матриц смежности неориентированных помеченных графов  
 # № 1  
 for i in range(N1):  
 M1.append([])  
 for j in range(N1):  
 M1[i].append(random.choice(chance))  
 if (i == j):  
 M1[i][j] = 0  
  
 for i in range(N1):  
 for j in range(N1):  
 M1[i][j] = M1[j][i]  
 print(M1[i][j], end='\t')  
 print('\n')  
 print('\n' \* 3)  
 # № 2  
  
 for i in range(N2):  
 M2.append([])  
 for j in range(N2):  
 M2[i].append(random.choice(chance))  
 if (i == j):  
 M2[i][j] = 0  
  
 for i in range(N2):  
 for j in range(N2):  
 M2[i][j] = M2[j][i]  
 print(M2[i][j], end='\t')  
 print('\n')  
 print('\n' \* 3)  
  
  
CreateTwoMatrix()  
# Преобразование в список смежности  
List\_S1 = []  
print("Первый список смежности:\n")  
for i in range(N1):  
 List\_S1.append([])  
 for j in range(N1):  
 if (M1[i][j] == 1):  
 List\_S1[i].append(j + 1)  
  
for i in range(N1):  
 print(i + 1, ' -- ', end='')  
 for j in range(len(List\_S1[i])):  
 print(List\_S1[i][j], end=' ')  
 print('\n')  
print("\n" \* 2)  
List\_S2 = []  
print("Второй список смежности:\n")  
for i in range(N2):  
 List\_S2.append([])  
 for j in range(N2):  
 if (M2[i][j] == 1):  
 List\_S2[i].append(j + 1)  
  
for i in range(N2):  
 print(i + 1, ' -- ', end='')  
 for j in range(len(List\_S2[i])):  
 print(List\_S2[i][j], end=' ')  
 print('\n')  
print('\n' \* 3)  
  
M11 = M1.copy()  
M22 = M2.copy()

**Листинг задания 2:**

import random  
  
def CreateTwoMatrix():  
 global M1  
 global M2  
 global N1  
 global N2  
 M1 = []  
 M2 = []  
 N1 = int(input("Введите размер 1 матрицы (а\*а): "))  
 N2 = int(input("Введите размер 2 матрицы (b\*b): "))  
 chance = [0,0,0,1,1,1,1,1,1,1]  
 #Шанс 1 - 70%  
 #Шанс 0 - 30%  
  
 #Создание 2 матриц смежности неориентированных помеченных графов   
 # № 1  
 for i in range(N1):  
 M1.append([])  
 for j in range(N1):  
 M1[i].append(random.choice(chance))  
 if(i == j):  
 M1[i][j] = 0  
   
 for i in range(N1):  
 for j in range(N1):  
 M1[i][j] = M1[j][i]  
 print(M1[i][j],end='\t')  
 print('\n')  
 print('\n'\*3)  
 # № 2  
  
 for i in range(N2):  
 M2.append([])  
 for j in range(N2):  
 M2[i].append(random.choice(chance))  
 if(i == j):  
 M2[i][j] = 0  
   
 for i in range(N2):  
 for j in range(N2):  
 M2[i][j] = M2[j][i]  
 print(M2[i][j],end='\t')  
 print('\n')  
 print('\n'\*3)  
  
CreateTwoMatrix()  
  
M11 = M1.copy()  
M22 = M2.copy()  
  
  
print("№1\n a) Отождествление вершин\n")  
  
a1 = int(input('Первая вершина : ')) - 1  
a2 = int(input('Вторая вершина : ')) - 1  
  
def identification(M1,a1,a2):  
   
 if(a1 > a2):  
 Max = a1  
 Min = a2  
 else:  
 Max = a2  
 Min = a1  
  
 G = []  
 for i in range(N1):  
 G.append(M1[Min][i] + M1[Max][i])  
 if(G[i] == 2):  
 G[i] = 1  
  
 for i in range(N1):  
 M1[Min][i] = G[i]  
 M1[i][Min] = G[i]  
  
 for i in range(N1-1):  
 for j in range(N1-1):  
 if(i >= Max):  
 M1[i][j] = M1[i+1][j]  
 elif(j >= Max):  
 M1[i][j] = M1[i][j + 1]  
 if(i >= Max and j >= Max):  
 M1[i][j] = M1[i + 1][j + 1]  
  
  
 return M1  
  
  
G = identification(M1,a1,a2)  
for i in range(len(G)-1):  
 for j in range(len(G)-1):  
 print(G[i][j],end="\t")  
 print("\n")  
print("\n"\*3)  
  
print("№1\n б) Стягивание вершин\n")  
  
M1 = M11.copy()  
def constriction(M1,a1,a2):  
 if(M1[a1][a2] != 1):  
 print("Пожалуйста, выберите другие вершины, выбранные вершины не являются связанными :'(")  
 return 0  
 F = identification(M1,a1,a2)  
 for i in range(len(F)-1):  
 for j in range(len(F)-1):  
 if(i == j):  
 F[i][j] = 0  
 return F  
F=0  
while(F==0):  
 a1 = int(input('Первая вершина : ')) - 1  
 a2 = int(input('Вторая вершина : ')) - 1  
 F = constriction(M1,a1,a2)  
for i in range(len(F)-1):  
 for j in range(len(F)-1):  
 print(F[i][j],end="\t")  
 print("\n")  
print("\n"\*3)  
  
print("№1\n в) Расщепление вершин во сторой матрице\n")  
  
a1 = int(input('Расщепить вершину под номером : \n')) - 1  
  
M2.append(M2[a1].copy())  
for i in range(len(M2)-1):  
 M2[i].append(M2[i][a1])  
  
M2[a1][len(M2)-1] = M2[len(M2)-1][a1] = 1  
  
M2[len(M2)-1].append(0)  
  
for i in range(len(M2)):  
 for j in range(len(M2[i])):  
 print(M2[i][j],end='\t')  
 print('\n')  
print('\n'\*3)

**Листинг задания 3:**

import random  
  
def CreateTwoMatrix():  
 global M1  
 global M2  
 global N1  
 global N2  
 M1 = []  
 M2 = []  
 N1 = int(input("Введите размер 1 матрицы (а\*а): "))  
 N2 = int(input("Введите размер 2 матрицы (b\*b): "))  
 chance = [0,0,0,1,1,1,1,1,1,1]  
 #Шанс 1 - 70%  
 #Шанс 0 - 30%  
  
 #Создание 2 матриц смежности неориентированных помеченных графов   
 # № 1  
 for i in range(N1):  
 M1.append([])  
 for j in range(N1):  
 M1[i].append(random.choice(chance))  
 if(i == j):  
 M1[i][j] = 0  
   
 for i in range(N1):  
 for j in range(N1):  
 M1[i][j] = M1[j][i]  
 print(M1[i][j],end='\t')  
 print('\n')  
 print('\n'\*3)  
 # № 2  
  
 for i in range(N2):  
 M2.append([])  
 for j in range(N2):  
 M2[i].append(random.choice(chance))  
 if(i == j):  
 M2[i][j] = 0  
   
 for i in range(N2):  
 for j in range(N2):  
 M2[i][j] = M2[j][i]  
 print(M2[i][j],end='\t')  
 print('\n')  
 print('\n'\*3)  
  
def combineGraphs(M1,M2):  
  
 if(len(M1) < len(M2)):  
 M1, M2 = M2, M1  
 G = []  
 for i in range(len(M1)):  
 G.append([])  
 for j in range(len(M1)):  
 try:  
 if(M1[i][j] or M2[i][j]):  
 G[i].append(1)  
 else:  
 G[i].append(0)  
 except IndexError:  
 G[i].append(M1[i][j])  
 return G  
   
  
def confluence(M1,M2):  
  
 if(len(M1) < len(M2)):  
 M1, M2 = M2, M1  
 G = []  
 for i in range(len(M1)):  
 G.append([])  
 for j in range(len(M1[i])):  
 try:  
 if(M1[i][j] and M2[i][j]):  
 G[i].append(1)  
 else:  
 G[i].append(0)  
  
 except IndexError:  
 G[i].append(0)  
   
 return G  
   
def annularSum(M1,M2):  
 if(len(M1) < len(M2)):  
 M1, M2 = M2, M1  
 G = []  
 for i in range(len(M1)):  
 G.append([])  
 for j in range(len(M1[i])):  
 try:  
 if((M1[i][j] == 1 and M2[i][j] == 0) or (M2[i][j] == 1 and M1[i][j] == 0)):  
 G[i].append(1)  
 else:  
 G[i].append(0)  
 except IndexError:  
 G[i].append(M1[i][j])  
 return G  
  
  
CreateTwoMatrix()  
# Объединение графов  
#G = combineGraphs(M1,M2)  
#print("Объединение графов:", '\n')  
# Пересечение  
G = []  
print("Пересечение:", '\n')  
G = confluence(M1,M2)  
  
# Кольцевая сумма  
#G = []  
#print("кольцевая сумма:", '\n')  
#G = annularSum(M1,M2)  
  
for i in range(len(G)):  
 for j in range(len(G[i])):  
 print(G[i][j],end='\t')  
 print('\n')  
print('\n'\*3)

**Вывод:** научилась производить унарные и бинарные операции с матрицами смежности.