Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе № 3

по дисциплине: "Логика и основы алгоритмизации в инженерных задачах"

на тему: "Унарные и бинарные операции над графами"

Выполнили:

студенты группы 20ВВ2

Макарова А. Ю.

Тельнова А. Д.

Принял:

д.т.н., профессор

Митрохин М. А.

к.т.н., доцент

Юрова О. В.

Пенза, 2021

**Задание 1**

1. Сгенерируйте (используя генератор случайных чисел) две матрицы M1, М2 смежности неориентированных помеченных графов G1, G2. Выведите сгенерированные матрицы на экран.

2. \* Для указанных графов преобразуйте представление матриц смежности в списки смежности. Выведите полученные списки на экран.

**Описание работы программы:**

**1.** Объявляются переменные int N1, N2 – количество вершин для графов, M1 и M2 – матрицы смежности неориентированных помеченных графов G1, G2.

Количество вершин графа вводится с клавиатуры. Далее генерируется матрица смежности для графа G1. Если индексы элементов i и j равны, то элемент матрицы заполняется нулем. В противном случае с помощью генератора случайных чисел заполняются остальные элементы матрицы нулями и единицами случайным образом. В результате матрица смежности графа G1 представляет собой симметричную матрицу, элементы главной диагонали которой – нули. Аналогично генерируется матрица смежности графа G2.

  /\*Генерация матрицы смежности графа G1\*/

    for (int i = 0; i < N1; i++)

        for (int j = i; j < N1; j++)

            if (i == j)

                M1[i][j] = 0;

            else

            {

                int k = rand() % 10;

                if (k > 2)

                    M1[i][j] = 1;

                else

                    M1[i][j] = 0;

                M1[j][i] = M1[i][j];

            }

Сгенерированные матрицы смежности M1, М2 для графов G1, G2 выводятся на экран.

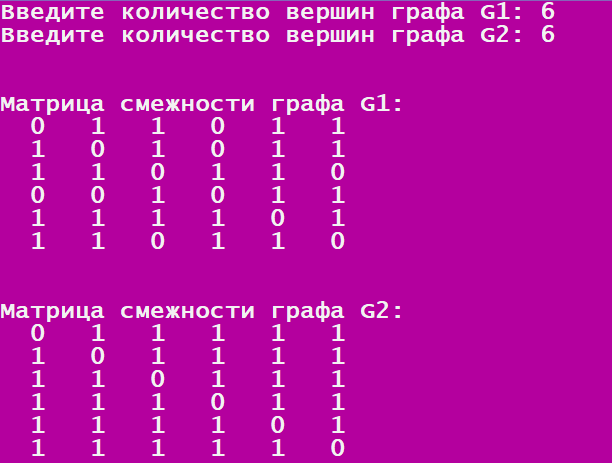


Рисунок - Вывод сгенерированных матриц смежности на экран

2\*. Преобразование матриц смежности в списки связности осуществляется с помощью функции CreateList.

void CreateList(int\*\* matrix, Node\*\* head, int N)

{

for (int i = 0; i < N; i++)

{

AddFirst(&head[i], i); //Создание первых элементов списков

for (int j = 0; j < N; j++)

if (matrix[i][j] == 1)

AddLast(head[i], j); //Добавляем в конец списка вершин, смежных с iой вершиной, вершину j

}

}

Перед этим создается тип данных struct Node, содержащий два поля – vertex, содержащее номер вершины графа, и next – указатель на следующий элемент списка.

Сначала создаются первые элементы списков с помощью функции AddFirst.

void AddFirst(Node\*\* head, int data)

{

Node\* NewNode = (Node\*)malloc(sizeof(Node)); // Создаем новый элемент

NewNode->vertex = data;

NewNode->next = (\*head); //Присваиваем адрес следующего элемента

(\*head) = NewNode; //Присваиваем указателю head новое значение

}

На вход функция получает адрес голову списка и номер вершины, который нужно сохранить в первом элементе, например, 3 для списка третьей вершины. Под элемент выделяется помять, номер вершины записывается в поле vertex, поле next получает адрес следующего элемента, а указателю, переданному функции присваивается весь новый элемент.

Затем просматриваются все элементы исходной матрицы смежности. Если элемент matrix[i][j] равен 1, то в конец списка i-ой вершины добавляется элемент с номером j-ой вершины. Это осуществляется с помощью функции AddLast.

void AddLast(Node\* head, int data)

{

Node\* last = FindLast(head); //Получение адреса последнего элемента списка

Node\* NewNode = (Node\*)malloc(sizeof(Node)); //Выделение памяти под новый элемент

NewNode->vertex = data;

NewNode->next = NULL;

last->next = NewNode; //Указатель на новый элемент записываеся последним элементом списка

}

Сначала находится адрес последнего элемента в списке и выделяется память под новый элемент. В созданном элементе поле vertex получает значение, переданное функции на вход – номер новой смежной вершины, а поле next устанавливается на NULL. Поле next элемента, на данный момент являющегося последним, устанавливается на созданный элемент.

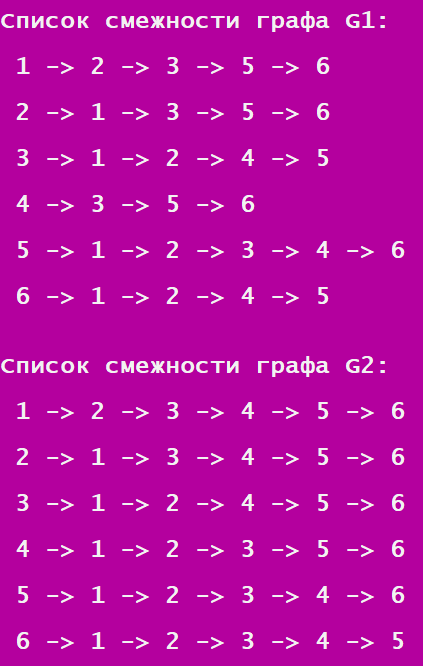


Рисунок - Вывод списков смежности на экран

**Задание 2**

1. Для матричной формы представления графов выполните операцию:

а) отождествления вершин

б) стягивания ребра

в) расщепления вершины

Номера выбираемых для выполнения операции вершин ввести с клавиатуры. Результат выполнения операции выведите на экран.

2. \* Для представления графов в виде списков смежности выполните операцию:

а) отождествления вершин

б) стягивания ребра

в) расщепления вершины

Номера выбираемых для выполнения операции вершин ввести с клавиатуры. Результат выполнения операции выведите на экран.

**Описание работы программы:**

Для того, чтобы выполнить желаемую операцию над графом, необходимо ввести номер операции из меню, выводимого в консоли.

А) Операция отождествления вершин выполняется для графа G1.

Объявляются переменные int vertexmin, vertexmax, предназначенные для хранения номеров вершин, для которых будет выполнено отождествление, M1result – массив для записи результата.

В случае, если введены номера вершин, которые не существуют в графе, пользователь видит сообщение об ошибке и требование ввести корректные номера вершин.

Отождествление вершин выполняется с помощью функции otozh. На вход функция получает номера отождествляемых вершин, матрицу смежности графа, размерность матрицы смежности.

Далее выполняется проверка на то, что переменная vertexmax действительно хранит номер большей отождествляемой вершины. Если это не так, то значения переменных vertexmin и vertexmax меняются местами. Операция отождествления осуществляется следующим образом: если индексы i и j элемента меньше номеров введенных вершин, то в матрицу результата записываются все элементы из матрицы смежности графа без изменений.

            if (i < vertexmax && j < vertexmax)

                M1result[i][j] = matrix[i][j];

Если же индекс i < vertexmax, а j > vertexmax - 1, то элементы результирующей матрицы с индексами [i][j] и [j][i] получают значение элемента исходной матрицы с индексом [i][j+1]. Таким образом, обращение к элементу [i][vertexmax] исходной матрицы никогда не произойдет.

            if (i < vertexmax && j > vertexmax - 1 && j != size - 1)

            {

                M1result[i][j] = matrix[i][j + 1];

                M1result[j][i] = matrix[i][j + 1];

            }

Если индексы i и j элемента исходной матрицы больше номеров отождествляемых вершин, то в матрицу результата записываются все элементы из матрицы смежности графа со сдвигом на один столбец влево и на одну строку вверх.

            if (j > vertexmax && i > vertexmax)

                M1result[i - 1][j - 1] = matrix[i][j];

     Единицы из вершины с большим номером переносятся в вершину с меньшим номером следующим образом:

    for (int i = 0; i < size; i++)

        if (i != vertexmax && (matrix[vertexmax][i] == 1 || matrix[vertexmin][i] == 1))

        {

            int j = i;

            if (j > vertexmax)

                j--;

            M1result[vertexmin][j] = 1;

            M1result[j][vertexmin] = 1;

        }

Функция otozh возвращает результирующую матрицу. Полученная матрица выводится на экран.

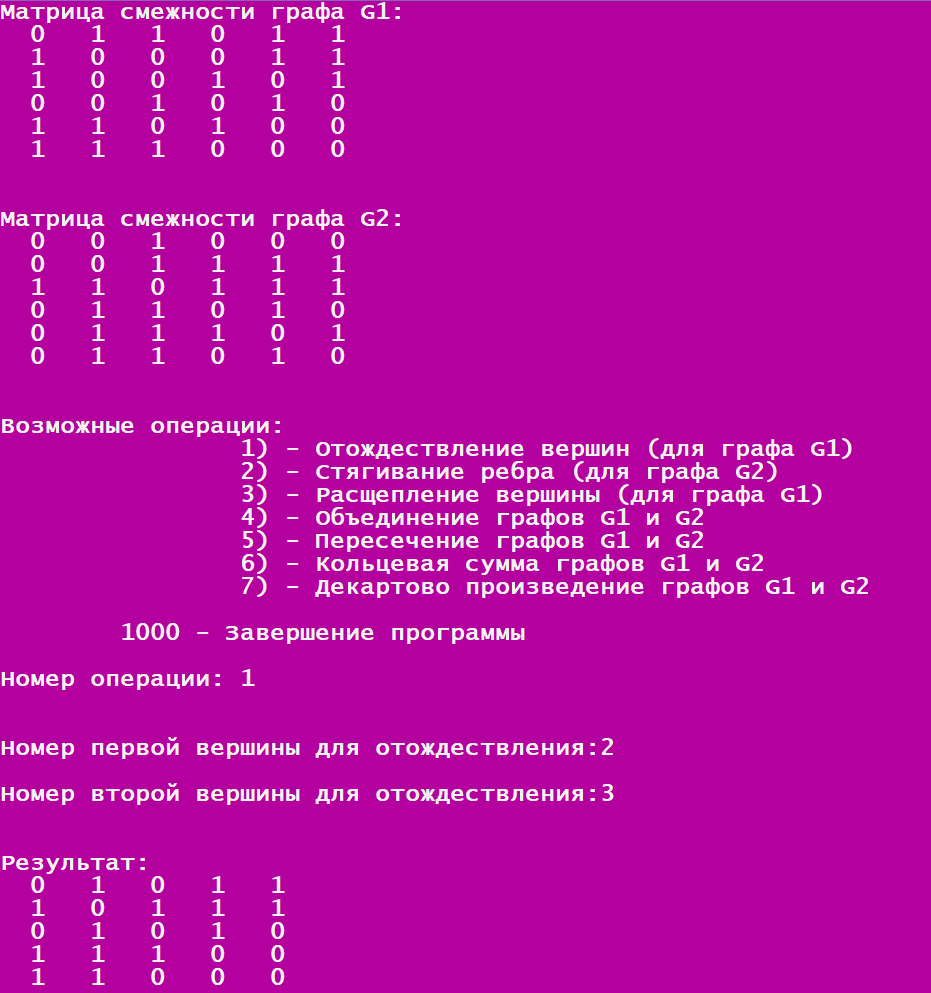


Рисунок - Отождествление вершин 2 и 3 графа G1

Б) Операция стягивания ребра выполняется для графа G2.

Объявляются переменные int vertexstart, vertexend, предназначенные для хранения номеров вершин, для которых будет выполнено стягивание, M2result – массив для записи результата.

В случае, если введены номера вершин, которые не существуют в графе или между ними не существует ребра, пользователь видит сообщение об ошибке и требование ввести корректные номера вершин.

Операция стягивания осуществляется следующим образом: между введенными вершинами удаляется ребро, вызывается функция отождествления вершин, инцидентных ребру, и удаляется петля для стянутых вершин.

//Удаление ребра

            M2[vertexstart][vertexend] = 0;

            M2[vertexend][vertexstart] = 0;

            //Отождествление вершин, инцидентных ребру

            M2result = otozh(vertexstart, vertexend, M2, N2);

            //Удаление петли

            for (int i = 0; i < N2 - 1; i++)

                M2result[i][i] = 0;

Полученная матрица выводится на экран.

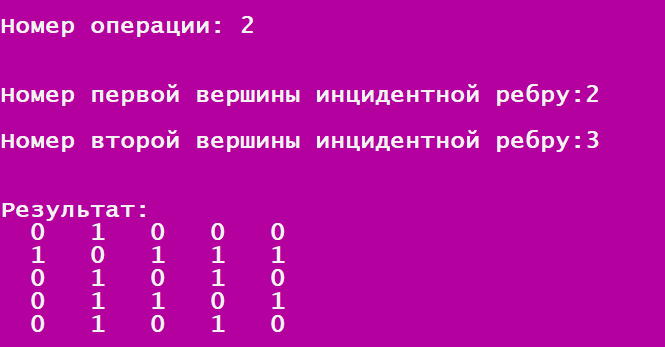


Рисунок - Стягивание ребра между вершинами 2 и 3 графа G2 (Исходный граф G2 на рис. 3)

В) Операция расщепления вершины выполняется для графа G1.

Объявляется переменная int vertex предназначенная для хранения номера вершины, для которой будет выполнено расщепление, M3result – массив для записи результата.

В случае, если введен номер вершины, которой не существуют в графе, пользователь видит сообщение об ошибке и требование ввести корректные номера вершин.

Операция расщепления вершины осуществляется следующим образом: если индексы i < vertex и j < vertex, то в матрицу результата записываются все элементы из матрицы смежности графа без изменений.

                    if (i < vertex && j < vertex)

                        M3result[i][j] = M1[i][j];

Если же индекс i < vertex, а j > vertex + 1, то элементы результирующей матрицы с индексами [i][j] и [j][i] получают значение элемента исходной матрицы с индексом [i][j-1]. Таким образом, обращение к элементу [i][vertex] исходной матрицы никогда не произойдет.

                    if (i < vertex && j > vertex + 1)

                    {

                        M3result[i][j] = M1[i][j-1];

                        M3result[j][i] = M1[i][j-1];

                    }

Если i > vertex+1 и j > vertex+1, то элементы записываются со сдвигом вниз и вправо на одну строку и столбец соответственно относительно их положения в исходной матрице.

                    if (i > vertex+1 && j > vertex+1)

                    {

                        M3result[i][j] = M1[i-1][j-1];

                        M3result[j][i] = M1[i-1][j-1];

                    }

Таким образом, в результирующей матрице заполнятся все строки и столбцы, кроме получившихся в результате расщепления (с номерами vertex и vertex+1). Заполнение оставшихся элементов реализуется следующим образом:

                for (int i = 0; i < N1 + 1; i++)

                {

                    if (i < vertex)

                    {

                        M3result[vertex][i] = M1[vertex][i];

                        M3result[i][vertex] = M1[vertex][i];

                        M3result[vertex+1][i] = 0;

                        M3result[i][vertex+1] = 0;

                    }

                    if (i > vertex + 1)

                    {

                        M3result[vertex + 1][i] = M1[vertex][i - 1];

                        M3result[i][vertex + 1] = M1[vertex][i - 1];

                        M3result[vertex][i] = 0;

                        M3result[i][vertex] = 0;

                    }

                }

Элементы на главной диагонали обнуляются.

                M3result[vertex][vertex] = 0;

                M3result[vertex + 1][vertex + 1] = 0;

Ребро между новыми вершинами создается.

                M3result[vertex][vertex + 1] = 1;

                M3result[vertex + 1][vertex] = 1;

Полученная матрица выводится на экран.

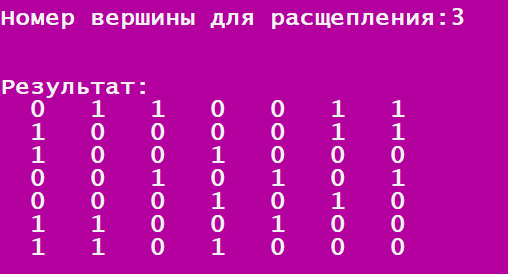


Рисунок - Расщепление вершины 3 графа G1(исходный граф G1 на рис. 3)

**Задание 3**

1. Для матричной формы представления графов выполните операцию:

а) объединения

б) пересечения

в) кольцевой суммы

Результат выполнения операции выведите на экран.

**Описание работы программы:**

А) Операция объединения графов выполняется для графов G1 и G2.

Переменная UnionResult предназначена для хранения массива результата.

Матрица смежности результирующего графа получается операцией поэлементного логического сложения матриц смежности исходных графов G1 и G2.

Полученная матрица выводится на экран с помощью функции printmatrix.

   for (int i = 0; i < N1; i++)

                for (int j = 0; j < N1; j++)

                    if (M1[i][j] == 1 || M2[i][j] == 1)

                        UnionResult[i][j] = 1;

                    else

                        UnionResult[i][j] = 0;

printmatrix(UnionResult, N1);

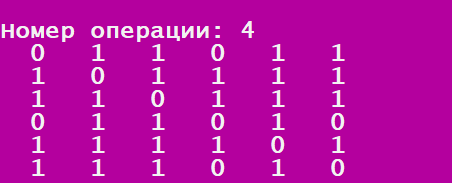


Рисунок 6 - Объединение графов G1 и G2 (Исходные графы на рис.3)

Б) Операция пересечения графов выполняется для графов G1 и G2.

Переменная ProductResult предназначена для хранения массива результата.

Результирующая матрица смежности получается операцией поэлементного логического умножения матриц смежности исходных графов G1 и G2.

Полученная матрица выводится на экран.

  for (int i = 0; i < N1; i++)

                for (int j = 0; j < N1; j++)

                    if (M1[i][j] == 1 && M2[i][j] == 1)

                        ProductResult[i][j] = 1;

                    else

                        ProductResult[i][j] = 0;

            printf("\n\nРезультат:\n");

            printmatrix(ProductResult, N1);

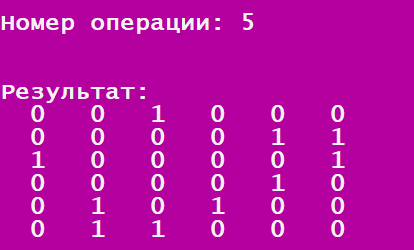


Рисунок - Пересечение графов G1 и G2 (Исходные графы на рис.3)

В) Операция кольцевой суммы графов выполняется для графов G1 и G2.

Переменная SumResult предназначена для хранения массива результата.

Результирующая матрица смежности получается операцией поэлементного логического сложения по модулю 2 матриц смежности исходных графов G1 и G2.

Полученная матрица выводится на экран.

  for (int i = 0; i < N1; i++)

                for (int j = 0; j < N1; j++)

                    if (M1[i][j] + M2[i][j] == 1)

                        SumResult[i][j] = 1;

                    else

                        SumResult[i][j] = 0;

            printf("\n\nРезультат:\n");

            printmatrix(SumResult, N1);

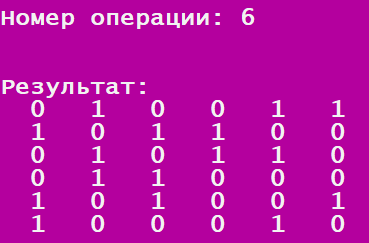


Рисунок - Кольцевая сумма графов G1 и G2 (Исходны графы на рис. 3)

**Задание 4 \***

1. Для матричной формы представления графов выполните операцию декартова произведения графов G = G 1 X G 2 .

Результат выполнения операции выведите на экран.

**Описание работы программы:**

Операция декартового произведения графов выполняется для графов G1 и G2.

Элементы с индексами i и j принадлежат матрице M1, элементы с индексами k и l принадлежат матрице M2.

Результирующая матрица строится следующим образом: если индексы i и j элемента матрицы M1 равны, то печатается элемент матрицы M2 с индексами k и l. Иначе, если индексы k и l элемента матрицы M2 равны, то печатается элемент матрицы M1 с индексами i и j. Если же индексы i и j и индексы k и l не равны между собой, то записывается ноль.

            for (int i = 0; i < N1; i++)

                for (int k = 0; k < N2; k++)

                {

                    printf("\n");

                    for (int j = 0; j < N1; j++)

                        for (int l = 0; l < N2; l++)

                        {

                            if (i == j)

                                printf("%3d ", M2[k][l]);

                            else if (k == l)

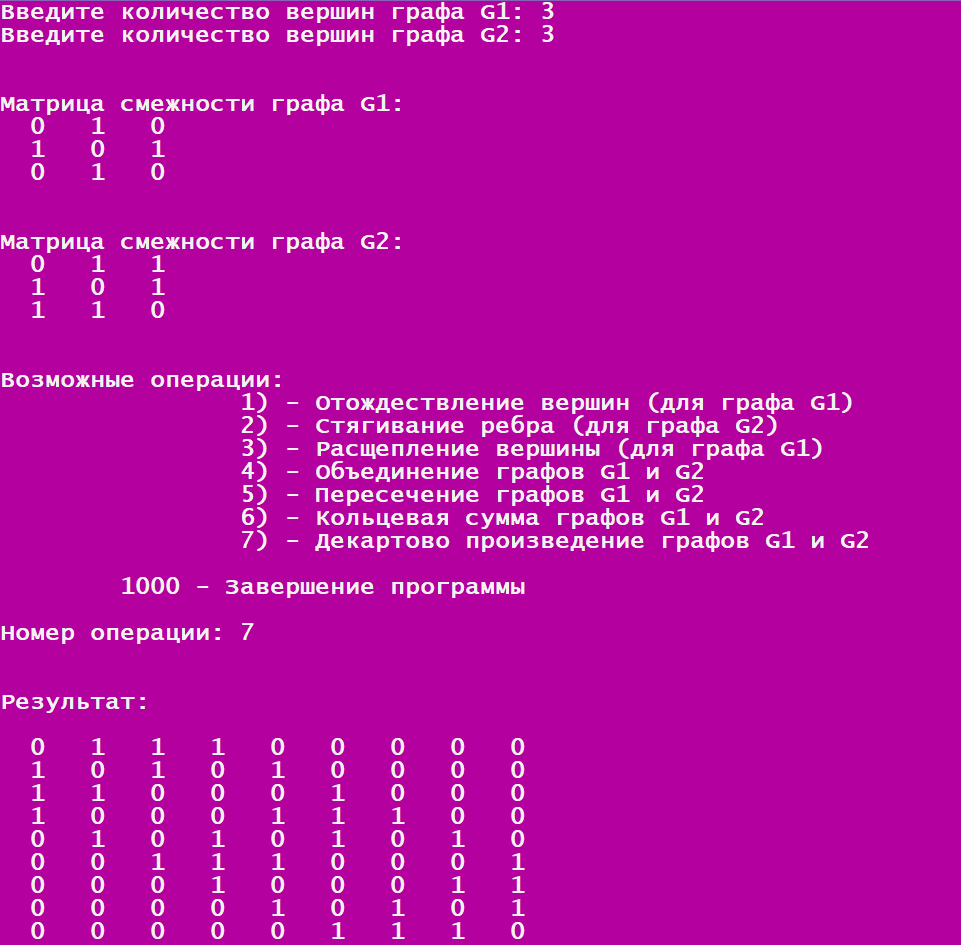
                                printf("%3d ", M1[i][j]);

                            else if (i != j && k != l)

                                printf("%3d ", 0);

                        }

                }

Рисунок - Декартово произведение графов G1 и G2

**Вывод**: Написали программу, осуществляющую генерацию двух матриц смежности для графов G1 и G2, преобразование матриц смежности в списки связности. Для матричных представлений графов реализованы операции отождествления ребра, стягивания ребра, расщепления вершины, а также объединения, пересечения, кольцевой суммы и декартового произведения графов. Выбор выполняемой операции осуществляется с помощью меню, сгенерированные матрицы, списки связности и результаты операций выводятся на экран.