***Поиск компонент связности графа***

Граф задан его матрицей смежности. Требуется определить количество компонент связности этого графа (*по материалам главы 3, п. 3.2.3 и 3.4*). При этом должны быть конкретно перечислены вершины, входящие в каждую компоненту связности.

Выбор алгоритма поиска компонент связности – произвольный. Например, приветствуется использование одного из видов обхода (поиск в глубину или поиск в ширину *по материалам п. 3.4.3*).

Пользователю должна быть предоставлена возможность редактировать исходную матрицу, т.е. изменять исходный граф без выхода из программы. Предусмотреть также возможность изменения количества вершин.

При выполнении работы разрешается (даже рекомендуется!) использовать матрицу бинарных отношений из лабораторной работы №2.

Вход программы: число вершин графа и матрица смежности.

Выход: разбиение множества вершин на подмножества, соответствующие компонентам связности.

Дополнительно:

Заданный граф рассматривать как ориентированный. Выполнять поиск компонент сильной связности.

**Решение:**

Алгоритм решения задачи: Вводим количество n вершин, заполняем матрицу n X n элементов, далее просматриваем, какая точка наиболее интересна для обхода, далее начинаем от это точки путь, постепенно записывая в путь. При выполнении максимального количества шагов по вершинам (все послед шаги приводящие к циклу) выводим конечное выражение обхода графа.

Исходный код :

**uses**

Crt;

**const**

Nmax = 100; //Максимальное количество вершин графа

**var**

A: **Array**[1..Nmax, 1..Nmax] **of** Shortint; //Матрица смежности

N: Integer; //Количество вершин графа

S: **Array**[1..Nmax] **of** Boolean; //Массив состояния для поиска в глубину

**procedure** DFS(**const** u: Integer);

**var**

v: Integer;

**begin**

Write(u, ' ');// Вводим вершину

S[u] := True; // Помечаем вершину как просмотренную

//Посещаем смежные с u вершины, которые ещё не были просмотрены

**for** v := 1 **to** N **do**

**if** (A[u, v] > 0) **and** (**not** S[v]) **then** DFS(v);

**end**;

**procedure** DFS\_Forest;//Построение леса поиска в глубину

**var**

v, k: Integer;

**begin**

{ S[v] = True,если вершина уже была просмотрена, и False иначе

  Изначально ни одна вершина ещё не просмотрена }

**for** v := 1 **to** N **do**

S[v] := False;

k := 0; // Номер компоненты связности

**for** v := 1 **to** N **do** //Проходим по всем вершинам, которые ещё не просмотрены

**if not** S[v] **then begin**

Inc(k);

Write(k, ') '); // Выводим номер компоненты связности

{ Строим дерево поиска в глубину }

DFS(v); //Запускаем построение дерева поиска в глубину с корнем v

WriteLn;

**end**;

**end**;

**var**

u, v, w: Integer;

Com: String;

c: Char;

E: Boolean;

**begin**

**for** u := 1 **to** Nmax **do**

**for** v := 1 **to** Nmax **do**

A[u, v] := 0;

N := 0;

//Меню

clrscr;

WriteLn('Выберите пункт меню:');

WriteLn('1 – Ввод графа');

WriteLn('2 – Вывод компонент связности');

WriteLn('3 – Изменение количество вершин ');

WriteLn('4 – Изменение матрицы смежности');

WriteLn('5 – Вывод матрицы смежности');

WriteLn('6 – Очистка экрана  ');

WriteLn('7 - Выход');

WriteLn();

Com := '';

**repeat**

**if** Com = '6' **then begin**

ClrScr;

Write('> ');

**end**

**else if** Com = '1' **then begin**

WriteLn('Введите число вершин:');

{$I-}

ReadLn(N);

{$I+}

**while** (IOResult <> 0) **or** (N < 1) **or** (N > Nmax) **do begin**

WriteLn('Введено неверно.Введите заново число вершин:');

{$I-}

ReadLn(N);

{$I+}

**end**;

WriteLn('Введите матрицу смежности (без пробелов,элемент матрицы - это либо 0, либо 1)');

E := False;

**for** u := 1 **to** N **do begin**

**for** v := 1 **to** N **do begin**

Read(c);

**if not** (c **in** ['0', '1']) **then begin**

A[u, v] := -1;

E := True;

**end**

**else**

A[u, v] := Ord(c) - Ord('0');

**end**;

Reset(Input);

**end**;

**if** E **then begin**

WriteLn('Ошибка .Введите заново:');

Sound(300);

Reset(Input);

**for** u := 1 **to** N **do**

**for** v := 1 **to** N **do**

**if** A[u, v] < 0 **then begin**

Write('A[', u, ', ', v, '] = ');

ReadLn(c);

**while not** (c **in** ['0', '1']) **do begin**

WriteLn('Введено не верно. Введите заново:');

Sound(300);

Write('A[', u, ', ', v, '] = ');

ReadLn(c);

**end**;

A[u, v] := Ord(c) - Ord('0');

**end**;

**end**;

WriteLn('Ввод графа завершён.');

WriteLn;

Write('> ');

**end**

**else if** Com = '2' **then begin**

WriteLn('Компоненты связности:');

DFS\_Forest;

WriteLn;

Write('> ');

**end**

**else if** Com = '3' **then begin**

WriteLn('Введите новое число вершин:');

{$I-}

ReadLn(N);

{$I+}

**while** (IOResult <> 0) **or** (N < 1) **or** (N > Nmax) **do begin**

WriteLn('Введено неверно. Введите заново число вершин:');

Sound(300);

{$I-}

ReadLn(N);

{$I+}

**end**;

WriteLn('Изменение выполнено.');

WriteLn;

Write('> ');

**end**

**else if** Com = '4' **then begin**

WriteLn('Введите через пробел тройку чисел u, v, w.');

WriteLn('u – Номер строки, v – номер столбца , w – новое значение элемента .');

WriteLn('Симметричный относительно главной диагонали элемент будет также изменен.');

Это почему же? Как следует из приведенного примера, вполне успешно вводится несимметричная матрица?

Reset(Input);

**while not** SeekEOLn **do begin**

{$I-}

ReadLn(u, v, w);

{$I+}

**while** (IOResult <> 0) **or not** (w **in** [0, 1]) **or** (u < 1) **or** (v < 1) **or**

(u > N) **or** (v > N) **do begin**

WriteLn('Введено не верно. Введите заново тройку чисел:');

Sound(300);

{$I-}

ReadLn(u, v, w);

{$I+}

**end**;

A[u, v] := w;

A[v, u] := w;

WriteLn('Введите ещё одну тройку или нажмите Enter для завершения ввода.');

**end**;

WriteLn('Изменение выполнено');

WriteLn;

Write('> ');

**end**

**else if** Com = '5' **then begin**

WriteLn('Матрица смежности графа :');

**for** u := 1 **to** N **do begin**

**for** v := 1 **to** N **do**

Write(A[u, v]);

WriteLn;

**end**;

WriteLn;

Write('> ');

**end**

**else if** Com = '' **then**

Write('> ')

**else begin**

WriteLn('Неизвестная команда.');

Write('> ');

**end**;

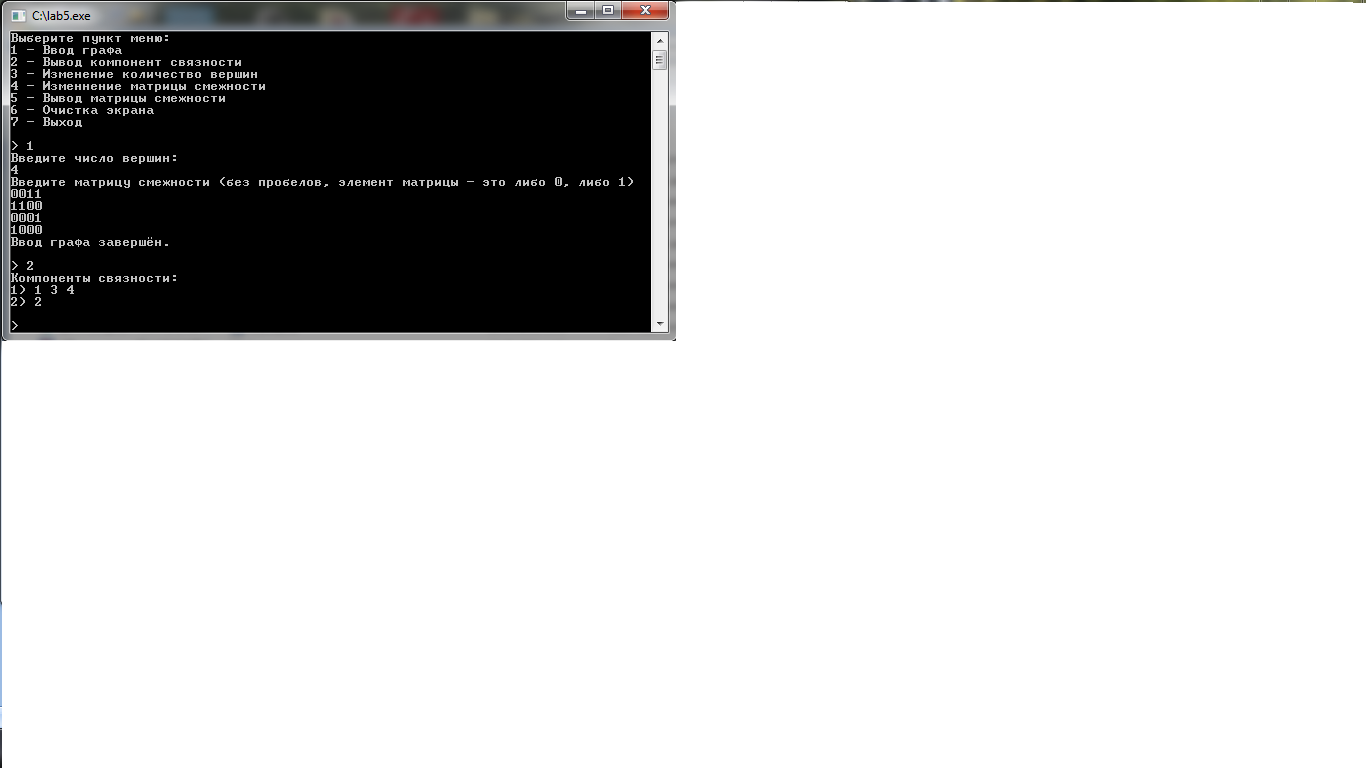
Reset(Input);

ReadLn(Com); // Получаем новую команду

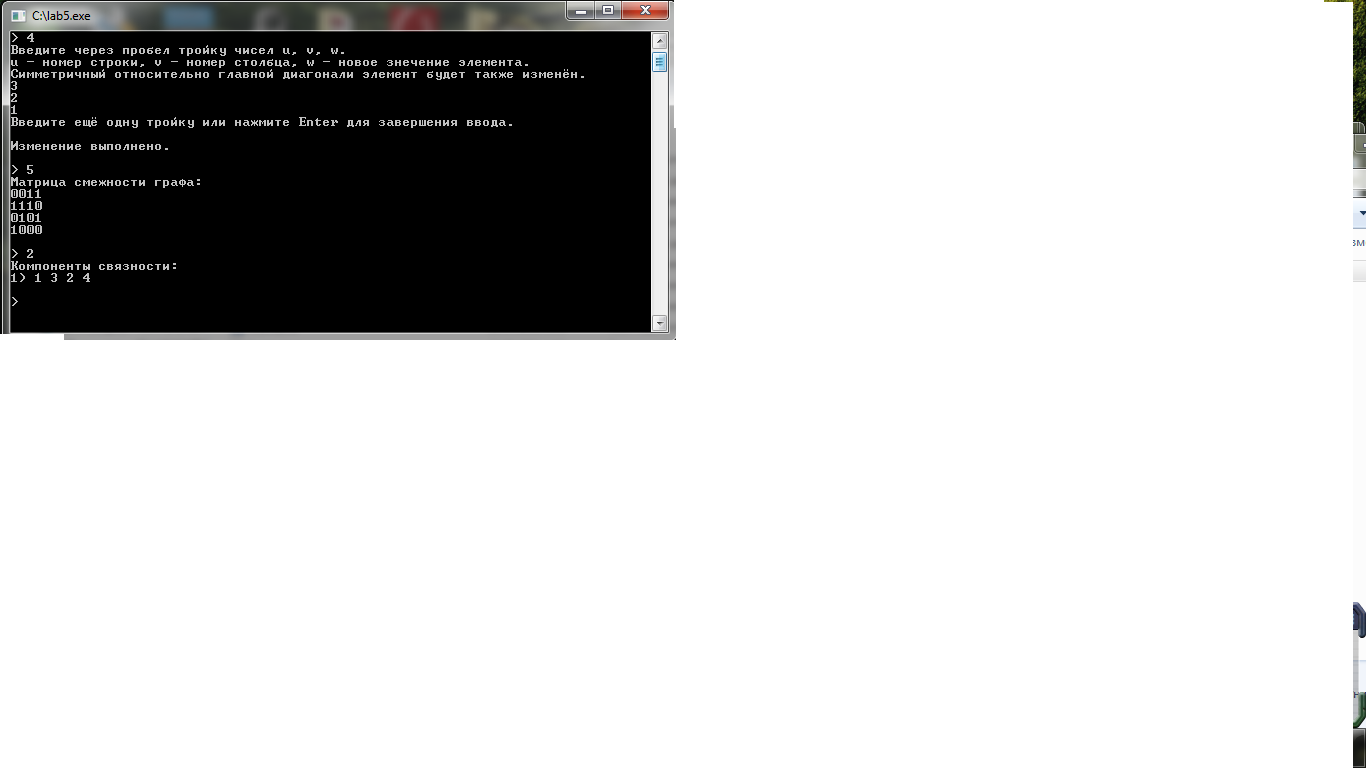
**until** Com = '7';

**end**.

Программа:

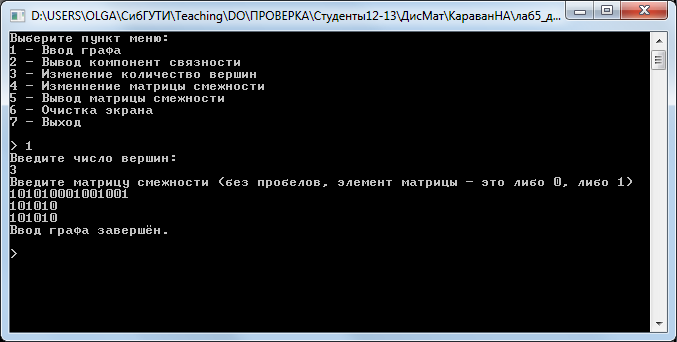


Изменение матрицы смежности , пункт-4



Непонятно – граф ориентированный или нет? Если он ориентированный – не должен изменяться симметричный элемент, как Вы пишете выше. Если неориентированный – вводимая матрица сразу должна быть симметричной, для чего можно вводить ее в треугольном виде.

Кроме того, вот такой ввод (внизу) недопустим. Если задали 3 вершины, значит, в строке должно быть три элемента. И больше не ввести.



Определитесь и доработайте программу.