**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

дисциплина: Дискретная математика

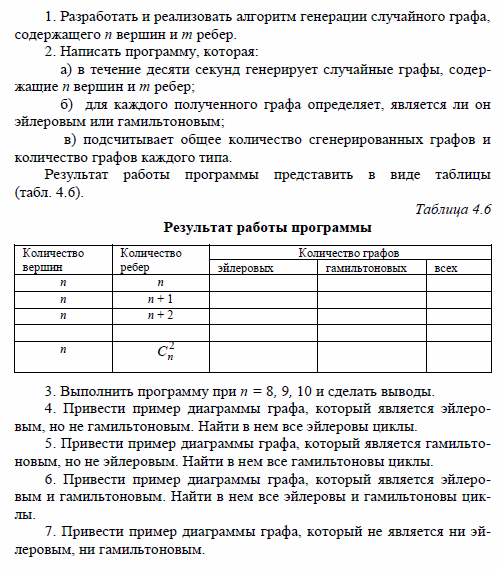
Лабораторная работа №4.2

тема: «Циклы»

|  |  |
| --- | --- |
|  | Выполнил: ст. группы  Проверил: Рязанов Ю.Д. |

**Цель занятия:** изучить разновидности циклов в графах, научиться генерировать случайные графы, определять их принадлежность к множеству эйлеровых и гамильтоновых графов, находить все эйлеровы и гамильтоновы циклы в графах.

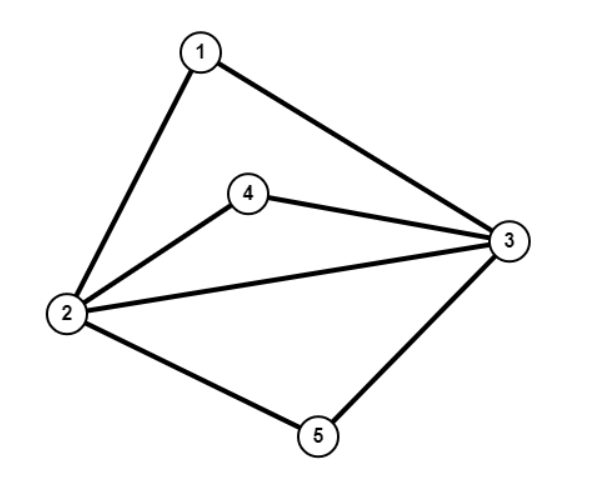
**Задания**



#include **"stdio.h"**#include **"time.h"**

#include **"conio.h"**#define **N** 15  
  
**int** \*log\_v;*// 3 глобальные переменные для проверки гамильтонова графа***int** \*\*graf;  
**int** \*a;  
  
  
*//Выделение памяти для графа а ввиде матрицы инцидентности n\*m***int** \*\* get\_memory\_matr(**int** n, **int** m) {  
 **int** i;  
 **int** \*\*a;  
  
 *// Выделение памяти под указатели на строки* a = (**int**\*\*)malloc((n + 1) \* **sizeof**(**int**\*));  
 **for** (i = 1; i <= n; i++) {  
 *// Выделение памяти под хранение строк* a[i] = (**int**\*)malloc((m + 1) \* **sizeof**(**int**));  
 }  
 **return** a;  
}  
  
*// Очищение памяти для матрицы***void** free\_memory(**int** \*\*a, **int** n) {  
 **int** i;  
 **for** (i = 1; i <= n; i++) {  
 free(a[i]);  
 }  
 free(a);  
}  
**int** \* init\_posl(**int** n) {  
 **int** \*a;  
 *// Выделение памяти* a = (**int**\*)malloc(n \* **sizeof**(**int**));  
 **return** a;  
}  
  
*// Заполнение матрицы n\*m нулями***void** clear\_matr(**int** \*\* a, **int** n, **int** m) {  
 **int** i, j;  
  
 **for** (i = 1; i <= n; i++) {  
 **for** (j = 1; j <= m; j++) {  
 a[i][j] = 0;  
 }  
 }  
 **return**;  
}  
  
*// Генерация случайного графа n вершин и m ребер***void** init\_matr\_rand\_numbers(**int** \*\* a, **int** n, **int** m) {  
 **int** i, j, k;  
 srand(clock());  
 **for** (j = 1; j <= m; j++) {  
 i = rand() % (n)+1;  
 a[i][j] = 1;  
 k = i;  
 **while** (i == k)  
 k = rand() % (n)+1;  
 a[k][j] = 1;  
 }  
 **return**;  
}  
  
  
*// Является ли граф связным***int** is\_sviaz(**int** \*\* graf, **int** n, **int** m) {  
 **int** a[**N**] = { 0 };  
 **int** i = 1, j, k;  
 a[i] = 1;  
 **int** count = 1;  
 **for** (i = 1; i <= n; i++) {  
 **if** (a[i] == i) {  
 **for** (j = 1; j <= m; j++) {  
 **if** (graf[i][j] == 1) {  
 **for** (k = 1; k <= n; k++) {  
 **if** ((k != i) && (graf[k][j] == 1) && (a[k] == 0))  
 a[k] = i + 1;  
 }  
 }  
 }  
 }  
 }  
 **for** (i = 1; i <= n; i++) {  
 **if** (a[i] == 0) {  
 **return** 0;  
 }  
 i++;  
 }  
 **return** 1;  
}  
  
**int** is\_smezh\_ver(**int** a, **int** b, **int** m) {  
 **for** (**int** j = 1; j <= m; j++) {  
 **if** (graf[a][j] && graf[b][j])  
 **return** 1;  
 }  
 **return** 0;  
}  
  
*// Является ли данный граф Эйлеровым***int** is\_eil\_graf(**int** \*\* graf, **int** n, **int** m) {  
 **int** count = 0;  
 **int** flag = 1;  
 **int** i = 1, j = 1;  
 **if** (is\_sviaz(graf, n, m)) {  
 **while** ((i <= n) && flag) {  
 **while** ((j <= m) && flag) {  
 **if** (graf[i][j] == 1)  
 count++;  
 j++;  
 }  
 **if** (((count + 1) % 2) == 0)  
 flag = 0;  
 i++;  
 j = 0;  
 }  
 }  
 **else** flag = 0;  
 **return** flag;  
}  
  
**int** find\_gam\_cikl(**int** i, **int** n, **int** m) {  
 **int** x;  
 **for** (x = 1; x <= n; x++) {  
 **if** ((is\_smezh\_ver(a[i - 1], x, m)) && (x != a[i - 1]) && (log\_v[x] == 0)) {  
 a[i] = x;  
 **if** ((a[i] == a[0]) && (i == n))  
 **return** 1;  
 **else** {  
 log\_v[x] = 1;  
 **if** (find\_gam\_cikl((i + 1), n, m))  
 **return** 1;  
 log\_v[x] = 0;  
 }  
 }  
 }  
 **return** 0;  
}  
  
  
*// Является ли данный граф гамильтоновым***int** is\_gam\_graf(**int** \*\* graf, **int** n, **int** m) {  
 **if** (is\_sviaz(graf, n, m)) {  
 **int** i;  
  
 **for** (i = 0; i < n; i++)  
 a[i] = 0;  
  
 **for** (i = 0; i <= n; i++)  
 log\_v[i] = 0;  
  
 a[0] = 1;  
  
 **if** (find\_gam\_cikl(1, n, m))  
 **return** 1;  
 **else  
 return** 0;  
 }  
 **else  
 return** 0;  
}  
  
  
  
  
**void** ten\_sec\_gen(**int** n, **int** last) {  
 **double** time\_spent = 0;  
 **int** count\_eil = 0, count\_gam = 0, count\_un = 0;  
 **int** m = n + 1;  
 *// выделение памяти* graf = get\_memory\_matr(n, m);  
 a = init\_posl((n));  
 log\_v = init\_posl((n + 1));  
 printf(**"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n"**);  
 printf(**"| Кол-во вершин | Количество ребер | Колво Эйлеровых графом | Гамильтоновых графов | Всех графов |\n"**);  
 **while** (m <= last) {  
 **while** (time\_spent < 10) {  
 clock\_t begin = clock();  
 clear\_matr(graf, n, m);  
 init\_matr\_rand\_numbers(graf, n, m);  
 clock\_t end = clock();  
 time\_spent += (**double**)(end - begin) / **CLOCKS\_PER\_SEC**;  
  
 *// Проверка, является ли сгенерированный граф гамельтоновым, если да, то count\_gam++* **if** (is\_gam\_graf(graf, n, m))  
 count\_gam++;  
  
 *// Проверка, является ли сгенерированный граф эйлеровым, если да, то count\_eil++* **if** (is\_eil\_graf(graf, n, m))  
 count\_eil++;  
  
 count\_un++;  
 }  
 *// вывод для n и m результатов красиво* printf(**"| %13d | %16d | %22d | %20d | %11d |\n"**, n, m, count\_eil, count\_gam, count\_un);  
 *// очищение памяти* free\_memory(graf, n);  
 m++;  
 *// выделение памяти* graf = get\_memory\_matr(n, m);  
 time\_spent = 0;  
 count\_eil = 0;  
 count\_gam = 0;  
 count\_un = 0;  
 }  
  
 **return**;  
}  
  
  
  
  
  
**int** main() {  
  
  
 printf(**"\n\nТаблица для 8 вершин: \n"**);  
 ten\_sec\_gen(8, 28);  
 free(a);  
 free(log\_v);  
  
 printf(**"\n\nТаблица для 9 вершин: \n"**);  
 ten\_sec\_gen(9, 36);  
 free(a);  
 free(log\_v);  
  
 printf(**"\n\nТаблица для 10 вершин: \n"**);  
 ten\_sec\_gen(10, 45);  
 free(a);  
 free(log\_v);  
  
 getchar();  
 getchar();  
 getchar();  
 **return** 0;  
}

**Привести пример диаграммы графа, который является эйлеровым, но не гамильтоновым. Найти в нем все эйлеровы циклы:**



**Эйлеровы циклы:**

1, 2, 4, 3, 5, 2, 3, 1

1, 2, 5, 3, 4, 2, 3, 1

1, 2, 3, 4, 2, 5, 3, 1

1, 2, 3, 5, 2, 4, 3, 1

1, 3, 2, 4, 3, 5, 2, 1

1, 3, 2, 5, 3, 4, 2, 1

1, 3, 4, 2, 5, 3, 2, 1

1, 3, 5, 2, 4, 3, 2, 1

2, 1, 3, 4, 2, 5, 3, 2

2, 1, 3, 5, 2, 4, 3, 2

2, 1, 3, 2, 4, 3, 5, 2

2, 1, 3, 2, 5, 3, 4, 2

2, 3, 4, 2, 5, 3, 1, 2

2, 3, 5, 2, 4, 3, 1, 2

2, 3, 1, 2, 4, 3, 5, 2

2, 3, 1, 2, 5, 3, 4, 2

3, 1, 2, 4, 3, 5, 2, 3

3, 1, 2, 5, 3, 4, 2, 3

3, 1, 2, 3, 4, 2, 5, 3

3, 1, 2, 3, 5, 2, 4, 3

3, 2, 1, 3, 4, 2, 5, 3

3, 2, 1, 3, 5, 2, 4, 3

3, 2, 4, 3, 5, 2, 1, 3

3, 2, 5, 3, 4, 2, 1, 3

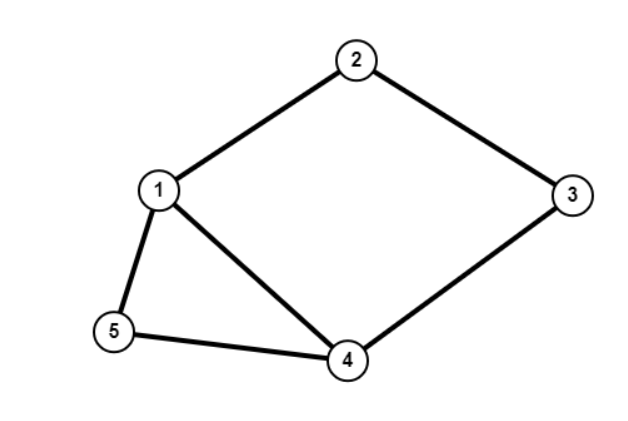
4, 3, 5, 2, 3, 1, 2, 4

4, 2, 5, 3, 2, 1, 3, 4

5, 2, 4, 3, 2, 1, 3, 5

5, 3, 4, 2, 1, 3, 2, 5

**Привести пример диаграммы графа, который является гамильтоновым, но не эйлеровым. Найти в нем все гамильтоновы циклы:**



**Гамильтоновы циклы :**

1, 2, 3, 4, 5, 1

1, 5, 4, 3, 2, 1

2, 3, 4, 5, 1, 2

2, 1, 5, 4, 3, 2

3, 4, 5, 1, 2, 3

3, 2, 1, 5, 4, 3

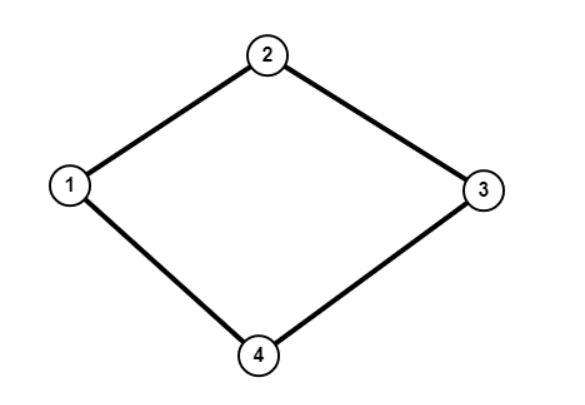
4, 5, 1, 2, 3, 4

4, 3, 2, 1, 5, 4

5, 1, 2, 3, 4, 5

5, 4, 3, 2, 1, 5

**Привести пример диаграммы графа, который является эйлеровым и гамильтоновым. Найти в нем все эйлеровы и гамильтоновы циклы:**



**Гамильтоновы циклы:**

1, 2, 3, 4, 1

1, 4, 3, 2, 1

2, 3, 4, 1, 2

2, 1, 4, 3, 2

3, 4, 1, 2, 3

3, 2, 1, 4, 3

4, 1, 2, 3, 4

4, 3, 2, 1, 4

**Эйлеровы циклы:**

1, 2, 3, 4, 1

1, 4, 3, 2, 1

2, 3, 4, 1, 2

2, 1, 4, 3, 2

3, 4, 1, 2, 3

3, 2, 1, 4, 3

4, 1, 2, 3, 4

4, 3, 2, 1, 4

**Пример графа не являющегося не эйлеровым, не гамильтоновым (так как он не связный):**

