**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ**

**УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**



ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ

**Лабораторная работа №4.2**

по дисциплине: Дискретная математика

тема: «Циклы»

Выполнил: ст. группы ПВ-221

Лоёк Никита Викторович

Проверили:

Бондаренко Татьяна Владимировна

Рязанов Юрий Дмитриевич

Белгород 2023 г.

Л а б о р а т о р н а я р а б о т а № 4.2

**Цель работы:** 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**

1. Разработать и реализовать алгоритм генерации случайного графа, содержащего n вершин и m ребер.

void GenRandomMatrixByVerEdge\_(vector<vector<bool>> &matrix,  
 vector<vector<int>> matrixOfIndex,  
 size\_t countOfEdges,  
 vector<vector<vector<bool>>> &resMatrix,  
 int lim) {  
  
 if (matrixOfIndex.size() > 0 && matrixOfIndex.size() >= countOfEdges && countOfEdges > 0 &&  
 resMatrix.size() < lim) {  
 unsigned int indexNow = rand() % matrixOfIndex.size();  
 srand(indexNow);  
  
 int indexNowX = matrixOfIndex[indexNow][0];  
 int indexNowY = matrixOfIndex[indexNow][1];  
 matrixOfIndex.erase((matrixOfIndex.cbegin() + indexNow));  
  
 matrix[indexNowX][indexNowY] = true;  
 matrix[indexNowY][indexNowX] = true;  
 GenRandomMatrixByVerEdge\_(matrix,  
 matrixOfIndex,  
 countOfEdges - 1,  
 resMatrix,  
 lim);  
  
 matrix[indexNowX][indexNowY] = false;  
 matrix[indexNowY][indexNowX] = false;  
 GenRandomMatrixByVerEdge\_(matrix,  
 matrixOfIndex,  
 countOfEdges,  
 resMatrix,  
 lim);  
 } else if (countOfEdges == 0) {  
 vector<vector<bool>> buf;  
 buf.assign(matrix.begin(), matrix.end());  
 resMatrix.push\_back(buf);  
 }  
}

vector<vector<vector<bool>>> GenRandomMatrixByVerEdge(size\_t countOfVertexes,  
 size\_t countOfEdges,  
 int lim) {  
 if (lim == -1) {  
 lim = 1000000;  
 }  
  
  
 vector<vector<vector<bool>>> resMatrix;  
  
 vector<vector<int>> matrixOfIndex;  
  
 for (int index\_x = 0; index\_x < countOfVertexes && countOfEdges; ++index\_x) {  
 for (int index\_y = index\_x + 1; index\_y < countOfVertexes; ++index\_y) {  
 matrixOfIndex.push\_back(**{**index\_x, index\_y**}**);  
 }  
 }  
 vector<vector<bool>> matrix(countOfVertexes, vector<bool>(countOfVertexes, false));  
 GenRandomMatrixByVerEdge\_(matrix,  
 matrixOfIndex,  
 countOfEdges,  
 resMatrix,  
 lim);  
  
 return resMatrix;  
}

1. Написать программу, которая:

а) в течение десяти секунд генерирует случайные графы, содержащие n вершин и m ребер;

б) для каждого полученного графа определяет, является ли он эйлеровым или гамильтоновым; в) подсчитывает общее количество сгенерированных графов и количество графов каждого типа.

#include <iostream>  
#include <vector>  
#include <random>  
  
  
using namespace std;

void GenRandomMatrixByVerEdge\_(vector<vector<bool>> &matrix,  
 vector<vector<int>> matrixOfIndex,  
 size\_t countOfEdges,  
 vector<vector<vector<bool>>> &resMatrix,  
 int lim) {  
  
 if (matrixOfIndex.size() > 0 && matrixOfIndex.size() >= countOfEdges && countOfEdges > 0 &&  
 resMatrix.size() < lim) {  
 unsigned int indexNow = rand() % matrixOfIndex.size();  
 srand(indexNow);  
  
 int indexNowX = matrixOfIndex[indexNow][0];  
 int indexNowY = matrixOfIndex[indexNow][1];  
 matrixOfIndex.erase((matrixOfIndex.cbegin() + indexNow));  
  
 matrix[indexNowX][indexNowY] = true;  
 matrix[indexNowY][indexNowX] = true;  
 GenRandomMatrixByVerEdge\_(matrix,  
 matrixOfIndex,  
 countOfEdges - 1,  
 resMatrix,  
 lim);  
  
 matrix[indexNowX][indexNowY] = false;  
 matrix[indexNowY][indexNowX] = false;  
 GenRandomMatrixByVerEdge\_(matrix,  
 matrixOfIndex,  
 countOfEdges,  
 resMatrix,  
 lim);  
 } else if (countOfEdges == 0) {  
 vector<vector<bool>> buf;  
 buf.assign(matrix.begin(), matrix.end());  
 resMatrix.push\_back(buf);  
 }  
}  
  
vector<vector<vector<bool>>> GenRandomMatrixByVerEdge(size\_t countOfVertexes,  
 size\_t countOfEdges,  
 int lim) {  
 if (lim == -1) {  
 lim = 1000000;  
 }  
  
  
 vector<vector<vector<bool>>> resMatrix;  
  
 vector<vector<int>> matrixOfIndex;  
  
 for (int index\_x = 0; index\_x < countOfVertexes && countOfEdges; ++index\_x) {  
 for (int index\_y = index\_x + 1; index\_y < countOfVertexes; ++index\_y) {  
 matrixOfIndex.push\_back(**{**index\_x, index\_y**}**);  
 }  
 }  
 vector<vector<bool>> matrix(countOfVertexes, vector<bool>(countOfVertexes, false));  
 GenRandomMatrixByVerEdge\_(matrix,  
 matrixOfIndex,  
 countOfEdges,  
 resMatrix,  
 lim);  
  
 return resMatrix;  
}  
  
  
bool IsHamilton\_(vector<vector<bool>> &matrix, vector<int> &G) {  
 bool isHamilton = false;  
 bool areNoWaysToAnyPoint = true;  
 for (int i = 0; i < matrix.size() && G.size() <= matrix.size() && !isHamilton; i++) {  
 if (matrix[i][G[G.size() - 1] - 1]) {  
 bool isNoPointInG = true;  
 for (int j = 0; j < G.size() && isNoPointInG; j++) {  
 if (G[j] == (i + 1)) {  
 isNoPointInG = false;  
 }  
 }  
  
 areNoWaysToAnyPoint = areNoWaysToAnyPoint && !isNoPointInG;  
  
 if (isNoPointInG) {  
 G.push\_back(i + 1);  
 isHamilton = IsHamilton\_(matrix, G);  
 G.pop\_back();  
 }  
 }  
 }  
  
 if (areNoWaysToAnyPoint && G.size() == matrix.size() && matrix[G[0] - 1][G[G.size() - 1] - 1] && G.size() > 2) {  
 return true;  
 }  
  
 return isHamilton;  
}

bool IsHamilton(vector<vector<bool>> &matrix) {  
 vector<int> G**{**1**}**;  
 return IsHamilton\_(matrix, G);  
}  
  
  
bool IsEuler\_(vector<vector<bool>> &matrix, vector<int> &G, int countOfEdges) {  
 bool isEuler = false;  
 for (int i = 0; i < matrix.size() && G.size() <= countOfEdges && !isEuler; i++) {  
 if (matrix[i][G[G.size() - 1] - 1]) {  
 bool isNoEdgeInG = true;  
 for (int j = 0; j < G.size() - 1 && isNoEdgeInG; j++) {  
 if (G[j] == G[G.size() - 1] && G[j + 1] == (i + 1) ||  
 G[j] == (i + 1) && G[j + 1] == G[G.size() - 1]) {  
 isNoEdgeInG = false;  
 }  
 }  
  
  
 if (isNoEdgeInG) {  
 G.push\_back(i + 1);  
 isEuler = IsEuler\_(matrix, G, countOfEdges);  
 G.pop\_back();  
 }  
 }  
 }  
  
 if (G[0] == G[G.size() - 1]) {  
 if (countOfEdges == (G.size() - 1)) {  
 return true;  
 }  
 }  
  
 return isEuler;  
}

bool IsEuler(vector<vector<bool>> &matrix) {  
 int countOfEdges = 0;  
 bool edgesEven = true;  
  
 for (int i = 0; i < matrix.size() && edgesEven; ++i) {  
 int bufSum = 0;  
 for (int j = 0; j < matrix.size(); ++j) {  
 bufSum += matrix[i][j];  
 }  
 edgesEven = edgesEven && (bufSum % 2 == 0);  
  
 }  
  
 if (edgesEven) {  
 for (int i = 0; i < matrix.size() && edgesEven; ++i) {  
 for (int j = i + 1; j < matrix.size(); ++j) {  
 countOfEdges += matrix[i][j];  
 }  
 }  
  
 vector<int> G**{**1**}**;  
  
 return edgesEven && IsEuler\_(matrix, G, countOfEdges);  
  
 }  
  
 return false;  
}

int EulerCount(vector<vector<vector<bool>>> &matrix) {  
 int counter = 0;  
 for (int i = 0; i < matrix.size(); ++i) {  
 counter += IsEuler(matrix[i]);  
 }  
  
 return counter;  
}

int HamiltonCount(vector<vector<vector<bool>>> &matrix) {  
 int counter = 0;  
 for (int i = 0; i < matrix.size(); ++i) {  
 counter += IsHamilton(matrix[i]);  
 }  
 return counter;  
}  
  
  
void CountOfEulerHamilton(int n, int limitOfEdges) {  
 int h = 1;  
 int limitMax = 30;  
  
 int countStr = 0;  
 int m = n;  
 while (countStr < limitMax && m < limitOfEdges) {  
 countStr++;  
 vector<vector<vector<bool>>> res = GenRandomMatrixByVerEdge(n, m, 100000);  
  
 int eulerCount = EulerCount(res);  
 int hamiltonCount = HamiltonCount(res);  
  
 cout << n << "\t"  
 << m << "\t"  
 << eulerCount << "\t"  
 << hamiltonCount << "\t"  
 << res.size() << endl;  
 m += h;  
 }  
 cout << endl;  
}  
  
  
int main() {  
 CountOfEulerHamilton(8, 28);  
 CountOfEulerHamilton(9, 36);  
 CountOfEulerHamilton(10, 45);   
  
 return 0;  
}

Результат работы программы представить в виде таблицы (табл. 6). Величину h подобрать такой, чтобы в таблице количество строк было в диапазоне от 20 до 30.

При n = 8 и h = 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Количество вершин | Количество ребер | Количество графов | | |
| эйлеровых | гамильтоновых | всех |
| 8 | 8 | 744 | 96 | 100000 |
| 8 | 9 | 582 | 1394 | 100000 |
| 8 | 10 | 679 | 6200 | 100000 |
| 8 | 11 | 673 | 15884 | 100000 |
| 8 | 12 | 651 | 32422 | 100000 |
| 8 | 13 | 784 | 44111 | 100000 |
| 8 | 14 | 742 | 70422 | 100000 |
| 8 | 15 | 798 | 69788 | 100000 |
| 8 | 16 | 778 | 95195 | 100000 |
| 8 | 17 | 744 | 89873 | 100000 |
| 8 | 18 | 773 | 98976 | 100000 |
| 8 | 19 | 754 | 99571 | 100000 |
| 8 | 20 | 744 | 99846 | 100000 |
| 8 | 21 | 783 | 99947 | 100000 |
| 8 | 22 | 876 | 99989 | 100000 |
| 8 | 23 | 840 | 98280 | 98280 |
| 8 | 24 | 105 | 20475 | 20475 |
| 8 | 25 | 0 | 3276 | 3276 |
| 8 | 26 | 0 | 378 | 378 |
| 8 | 27 | 0 | 28 | 28 |

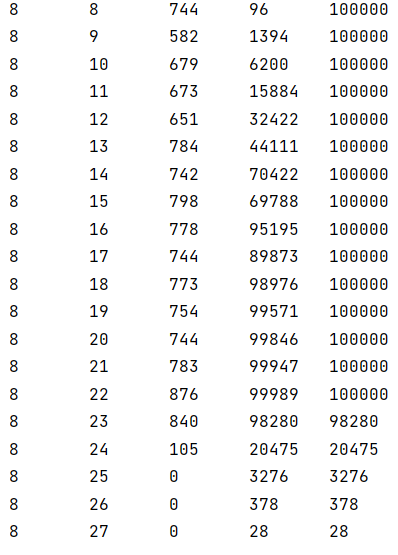
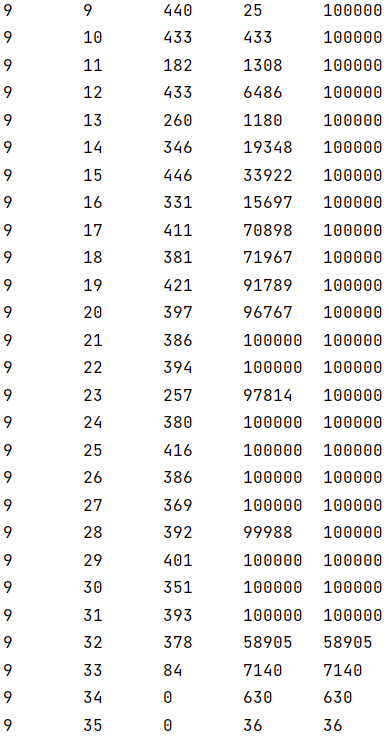
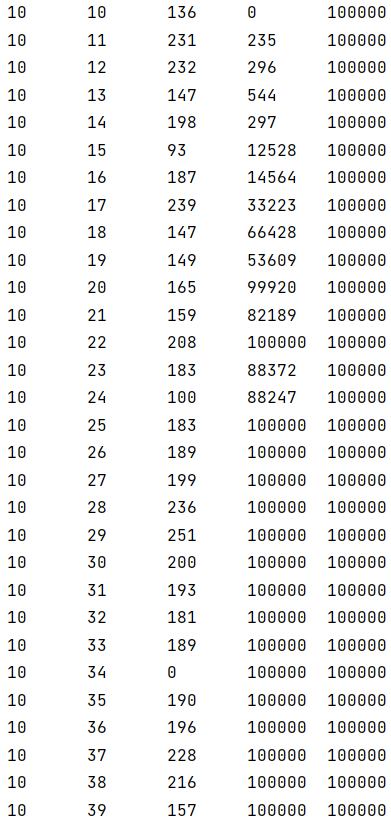
При n = 9 и h = 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Количество вершин | Количество ребер | Количество графов | | |
| эйлеровых | гамильтоновых | всех |
| 9 | 9 | 440 | 25 | 100000 |
| 9 | 10 | 433 | 433 | 100000 |
| 9 | 11 | 182 | 1308 | 100000 |
| 9 | 12 | 433 | 6486 | 100000 |
| 9 | 13 | 260 | 1180 | 100000 |
| 9 | 14 | 346 | 19348 | 100000 |
| 9 | 15 | 446 | 33922 | 100000 |
| 9 | 16 | 331 | 15697 | 100000 |
| 9 | 17 | 411 | 70898 | 100000 |
| 9 | 18 | 381 | 71967 | 100000 |
| 9 | 19 | 421 | 91789 | 100000 |
| 9 | 20 | 397 | 96767 | 100000 |
| 9 | 21 | 386 | 100000 | 100000 |
| 9 | 22 | 394 | 100000 | 100000 |
| 9 | 23 | 257 | 97814 | 100000 |
| 9 | 24 | 380 | 100000 | 100000 |
| 9 | 25 | 416 | 100000 | 100000 |
| 9 | 26 | 386 | 100000 | 100000 |
| 9 | 27 | 369 | 100000 | 100000 |
| 9 | 28 | 392 | 99988 | 100000 |
| 9 | 29 | 401 | 100000 | 100000 |
| 9 | 30 | 351 | 100000 | 100000 |
| 9 | 31 | 393 | 100000 | 100000 |
| 9 | 32 | 378 | 58905 | 58905 |
| 9 | 33 | 84 | 7140 | 7140 |
| 9 | 34 | 0 | 630 | 630 |
| 9 | 35 | 0 | 36 | 36 |

При n = 10 и h = 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Количество вершин | Количество ребер | Количество графов | | |
| эйлеровых | гамильтоновых | всех |
| 10 | 10 | 136 | 0 | 100000 |
| 10 | 11 | 231 | 235 | 100000 |
| 10 | 12 | 232 | 296 | 100000 |
| 10 | 13 | 147 | 544 | 100000 |
| 10 | 14 | 198 | 297 | 100000 |
| 10 | 15 | 93 | 12528 | 100000 |
| 10 | 16 | 187 | 14564 | 100000 |
| 10 | 17 | 239 | 33223 | 100000 |
| 10 | 18 | 147 | 66428 | 100000 |
| 10 | 19 | 149 | 53609 | 100000 |
| 10 | 20 | 165 | 99920 | 100000 |
| 10 | 21 | 159 | 82189 | 100000 |
| 10 | 22 | 208 | 100000 | 100000 |
| 10 | 23 | 183 | 88372 | 100000 |
| 10 | 24 | 100 | 88247 | 100000 |
| 10 | 25 | 183 | 100000 | 100000 |
| 10 | 26 | 189 | 100000 | 100000 |
| 10 | 27 | 199 | 100000 | 100000 |
| 10 | 28 | 236 | 100000 | 100000 |
| 10 | 29 | 251 | 100000 | 100000 |
| 10 | 30 | 200 | 100000 | 100000 |
| 10 | 31 | 193 | 100000 | 100000 |
| 10 | 32 | 181 | 100000 | 100000 |
| 10 | 33 | 189 | 100000 | 100000 |
| 10 | 34 | 0 | 100000 | 100000 |
| 10 | 35 | 190 | 100000 | 100000 |
| 10 | 36 | 196 | 100000 | 100000 |
| 10 | 37 | 228 | 100000 | 100000 |
| 10 | 38 | 216 | 100000 | 100000 |
| 10 | 39 | 157 | 100000 | 100000 |

1. Выполнить программу при n = 8,9,10 и сделать выводы.

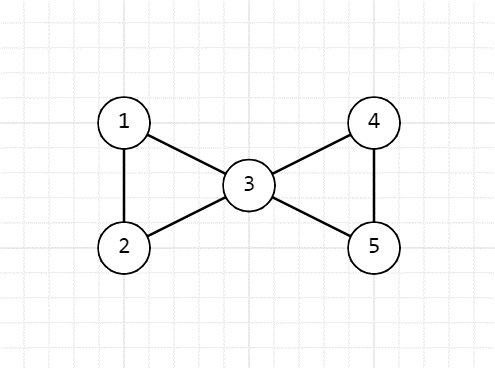
  

1. Привести пример диаграммы графа, который является эйлеровым, но не гамильтоновым. Найти в нем все эйлеровы циклы.

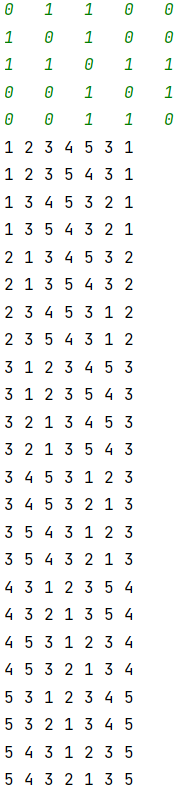
Матрица смежности:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| G | 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 2 | 1 | 0 | 1 | 0 | 0 |
| 3 | 1 | 1 | 0 | 1 | 1 |
| 4 | 0 | 0 | 1 | 0 | 1 |
| 5 | 0 | 0 | 1 | 1 | 0 |

Граф:



Все эйлеровы циклый:

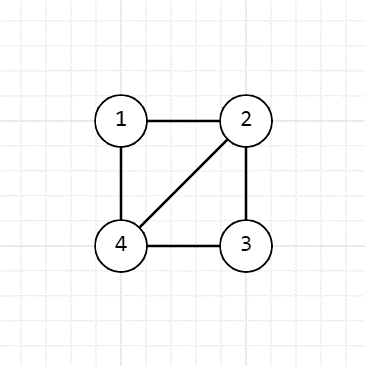


1. Привести пример диаграммы графа, который является гамильтоновым, но не эйлеровым. Найти в нем все гамильтоновы циклы.

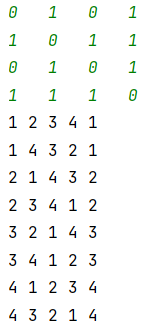
Матрица смежности:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| G | 1 | 2 | 3 | 4 |
| 1 | 0 | 1 | 0 | 1 |
| 2 | 1 | 0 | 1 | 1 |
| 3 | 0 | 1 | 0 | 1 |
| 4 | 1 | 1 | 1 | 0 |

Граф:



Все гамильтоновы циклый:

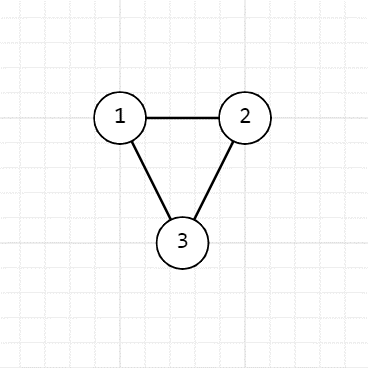


1. Привести пример диаграммы графа, который является эйлеровым и гамильтоновым. Найти в нем все эйлеровы и гамильтоновы циклы.

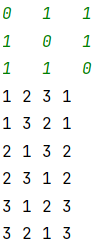
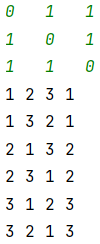
Матрица смежности:

|  |  |  |  |
| --- | --- | --- | --- |
| G | 1 | 2 | 3 |
| 1 | 0 | 1 | 1 |
| 2 | 1 | 0 | 1 |
| 3 | 1 | 1 | 0 |

Граф:



Все эйлеровы циклый: Все гамильтоновы циклый:

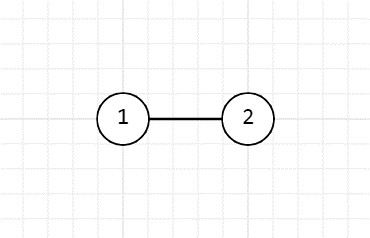
 

1. Привести пример диаграммы графа, который не является ни эйлеровым, ни гамильтоновым.

Матрица смежности:

|  |  |  |
| --- | --- | --- |
| G | 1 | 2 |
| 1 | 0 | 1 |
| 2 | 1 | 0 |

Граф:



**Вывод**

**Вывод**: в ходе работы я изучил разновидности циклов в графах, научился генерировать случайные графы, определять их принадлежность к множеству эйлеровых и гамильтоновых графов, научился находить все эйлеровы и гамильтоновы циклы в графах.