МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

**ИНТЕЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ**

ОТЧЁТ

По лабораторной работе № \_\_

Выполнил:

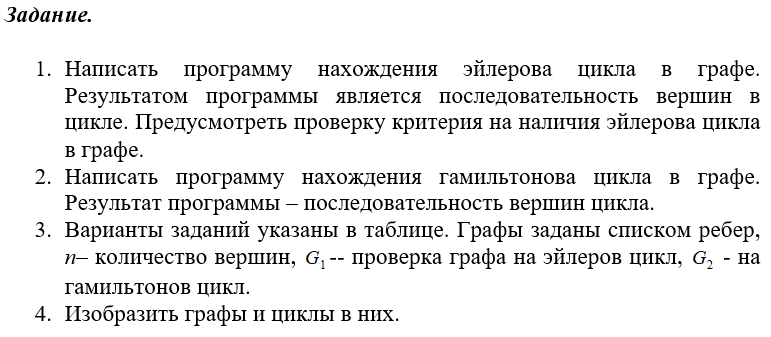
Студент группы ИИ-22

Копанчук Евгений Романович

Проверил\_:

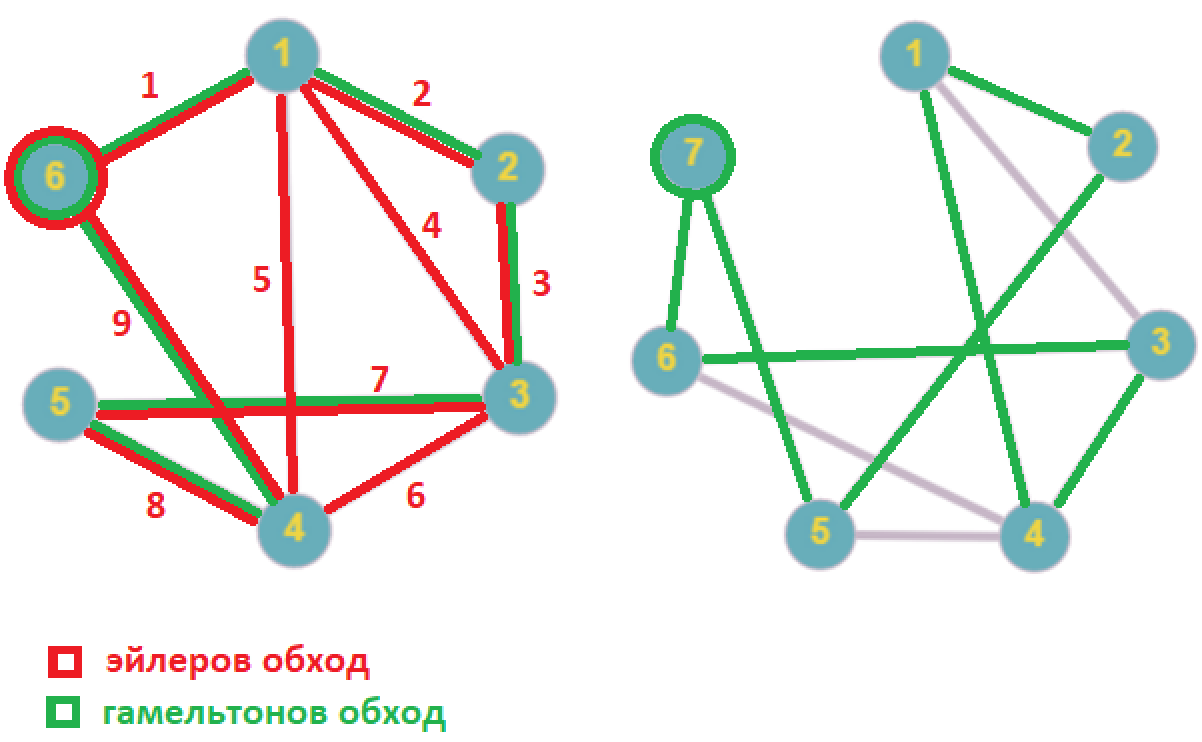
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Брест – 2022



**Ход работы**

**ВИЗУАЛИЗАЦИЯ ОБХОДОВ:**

****

**КОД ПРОГПРАММЫ:**

|  |
| --- |
| bool Graph::HAMILTONIAN\_PATH(int\*& arr, int& size, int start) {  // is Gamelton ?  if (vertices < 2) return false;  if (CALC\_COMPONENTS\_BFS() > 1) return false;  if (CALC\_COMPONENTS\_BFS() > 1) return false;  for (int i = 0; i < vertices; i++) {  int P = 0;  for (int j = 0; j < vertices; j++)  if (matrix[i][j]) P++;  if (P < 2) return false;  }  // find path  bool\* used = new bool[vertices];  for (int i = 0; i < vertices; i++)  used[i] = 0;  int s\_size = 0;  int\* stack = new int[s\_size];  push\_elem(stack, s\_size, start - 1);  used[start - 1] = true;  for (int i = 0; i < vertices; i++)  for (int j = 0; j < vertices; j++)  for (int k = 0; k < vertices; k++)  if (!used[k] && matrix[j][k]) {  push\_elem(stack, s\_size, k);  used[k] = true;  break;  }  if (matrix[start - 1][stack[s\_size - 1]])  push\_elem(stack, s\_size, start - 1);  if (stack[s\_size - 1] == start - 1 && s\_size == vertices + 1) {  for (int i = 0; i < s\_size; i++)  push\_elem(arr, size, stack[i]);  delete[] stack;  delete[] used;  return true;  }  else {  delete[] stack;  delete[] used;  return false;  }  }  bool Graph::EULER\_PATH(int\*& arr, int& size, int start) {  // is Euler ?  if (vertices < 2) return false;  if (CALC\_COMPONENTS\_BFS() > 1) return false;  for (int i = 0; i < vertices; i++) {  int P = 0;  for (int j = 0; j < vertices; j++)  if (matrix[i][j]) P++;  if (P % 2 != 0 || P < 2) return false;  }  // find path  bool\*\* used = new bool\*[vertices];  for (int i = 0; i < vertices; i++) {  used[i] = new bool[vertices];  for (int j = 0; j < vertices; j++)  used[i][j] = (matrix[i][j]) ? 0 : 1;  }  int s\_size = 0;  int\* stack = new int[s\_size];  push\_elem(stack, s\_size, start - 1);  while (s\_size != 0) {  int w = stack[s\_size - 1];  bool found\_edge = false;  for (int i = 0; i < vertices; i++)  if (!used[w][i]) {  push\_elem(stack, s\_size, i);  used[w][i] = 1;  used[i][w] = 1;  found\_edge = true;  break;  }  if (!found\_edge)  push\_elem(arr, size, pop\_back\_elem(stack, s\_size));  }  return ((arr[0] == arr[size - 1]) ? true : false);  } |

