МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

**ИНТЕЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ**

ОТЧЁТ

По лабораторной работе № \_\_

Выполнил:

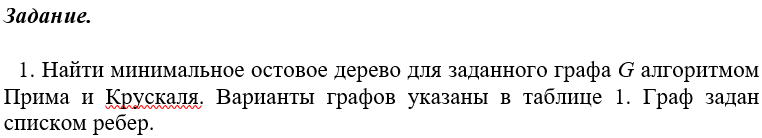
Студент группы ИИ-22

Копанчук Евгений Романович

Проверил\_:

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

Брест – 2022





**Код программы Крускала:**

|  |
| --- |
| #include<iostream>  #include<vector>  #include<algorithm>  using namespace std;  struct Node {  int start;  int end;  int length;  Node(int start, int end, int length) :  start(start),  end(end),  length(length) {}  };  bool compare(Node a, Node b) {  return a.length < b.length;  }  void Kruskal(vector<Node>& arr, vector<bool>& visited) {  int M, N;  M = visited.size();  N = arr.size();  sort(arr.begin(), arr.end(), compare);  int weight = 0;  for (int i = 0; i < N; i++)  if (!visited[arr[i].start] || !visited[arr[i].end]) {  weight += arr[i].length;  visited[arr[i].start] = true;  visited[arr[i].end] = true;  }  cout << "Kruskal: min weight = " << weight << endl;  }  int main() {  vector<Node> arr = {  Node(1, 2, 5),  Node(1, 3, 2),  Node(1, 4, 3),  Node(2, 4, 3),  Node(3, 4, 2),  Node(3, 5, 6),  Node(4, 5, 7),  Node(4, 6, 5),  Node(5, 6, 4)  };  vector<bool> visited = { 0, 0, 0, 0, 0, 0, 0, 0, 0 };  Kruskal(arr, visited);  } |

**Вывод программы:**

|  |  |
| --- | --- |
|  |  |

**Код программы Прима:**

|  |
| --- |
| #include<iostream>  #include<vector>  #include<algorithm>  using namespace std;  int main()  {  int n = 6;  vector<vector<int>> matrix = {  // a b c d e f  /\*a\*/{0,5,2,3,0,0},  /\*b\*/{5,0,0,3,0,0},  /\*c\*/{2,0,0,2,6,0},  /\*d\*/{3,3,2,0,7,5},  /\*e\*/{0,0,6,7,0,4},  /\*f\*/{0,0,0,5,4,0}  };  const int INF = 1000000000;  // алгоритм  vector<bool> used(n);  vector<int> min\_e(n, INF), sel\_e(n, -1);  min\_e[0] = 0;  int weight = 0;  for (int i = 0; i < n; ++i) {  int v = -1;  for (int j = 0; j < n; ++j)  if (!used[j] && (v == -1 || min\_e[j] < min\_e[v]))  v = j;  if (min\_e[v] == INF) {  cout << "No MST!";  return 0;  }  used[v] = true;  if (sel\_e[v] != -1)  weight += matrix[i][v];  for (int to = 0; to < n; ++to)  if (matrix[v][to] < min\_e[to]) {  min\_e[to] = matrix[v][to];  sel\_e[to] = v;  }  }  cout << "Prim: min weight = " << weight << endl;  } |

**Минимальное островное дерево:**

|  |
| --- |
|  |