Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ульяновский государственный технический университет»

Кафедра: «Вычислительная техника»

Дисциплина: «Математическая логика и дискретная математика»

Лабораторная работа №4

«Графы. Определение кратчайшего пути»

Выполнил студент

группы ИВТАСбд-12

Давыдов Д. Д.

Проверил

Игонин А. Г.

Ульяновск, 2022

# Оглавление

[Оглавление 2](#_Toc103938042)

[Цель работы 3](#_Toc103938043)

[Описание метода выполнения 3](#_Toc103938044)

[Вывод 7](#_Toc103938045)

# Цель работы

Требуется написать программу, которая находит кратчайший путь в орграфе. Пользователь вводит матрицу весов, начальную и конечную точку.

# Описание метода выполнения

Граф - это множество упорядоченных пар G := (V, E), где V – множество вершин графа, а E – множество ребер графа. Оба множества непустые. Графы бывают ориентированные и неориентированные. Их отличие заключается в том, что в первом случае проход по ребру ориентированного графа (дуга) возможен только в одном направлении, а во втором – в обоих. Матрица смежности – квадратная матрица представления отношения вершин и ребер, где значение элемента говорит о весе ребра из i точки в j. Вес ребра —значение, поставленное в соответствие данному ребру. Кратчайший путь в графе – последовательность вершин, соединенных ребрами или дугами, у которых сумма весов этих ребер или дуг наименьшая.

Для того чтобы пользователь мог вводить данные необходимо создать html страницу, состоящую из одной мультистрочной формы и двух форм для ввода начальной и конечной вершины и кнопок взаимодействия. Код алгоритма был написан на PHP. PHP — это язык программирования, специально разработанный для написания web-приложений (сценариев), исполняющихся на Web-сервере. PHP взаимодействует с html.

Пользователь вводит матрицу весов. После нажатия кнопки «Анализ», через JS скрипт происходит передача данных в PHP скрипт.

Первом делом происходит запись данных в матрицу весов, объявление начального и конечного индекса, а также проверка данных на ошибки.

$graph = $\_POST["weights"];

$startIndex = $\_POST["startIndex"]-1;

$endIndex = $\_POST["endIndex"]-1;

$rows = explode("\n", $graph);

for ($i = 0; $i < count($rows); $i++){

$els = preg\_replace("/\s+/", " ", $rows[$i]);

$els = explode(" ", $els);

$row = [];

for($j = 0; $j < count($els); $j++){

if ($els[$j] == ""){

unset($els[$j]);

$j--;

}

}

if (count($els) != count($rows)){

echo "<p>Неверный ввод</p>";

die();

}

for($j = 0; $j < count($els); $j++){

$t = $els[$j];

if ($t == "-"){

array\_push($row, INF);

}

else {

array\_push($row, (int)$t);

}

}

array\_push($weightMatrix, $row);

}

analize();

Дальше запускается алгоритм анализа, который первым делом запускает алгоритм изначального заполнения матрицы путей.

function initPathMatrix() {

global $weightMatrix, $pathMatrix;

for ($i = 0; $i < count($weightMatrix); $i++) {

for ($j = 0; $j < count($weightMatrix[$i]); $j++) {

$pathMatrix[$i][$j] = $i;

}

$pathMatrix[$i][$i] = -1;

}

}

Дальше происходит постройка матрицы дистанций с помощью алгоритма Флойда – Уоршелла.

global $weightMatrix, $distanceMatrix, $pathMatrix;

for ($k = 0; $k < count($weightMatrix); $k++) {

for ($i = 0; $i < count($weightMatrix); $i++) {

for ($j = 0; $j < count($weightMatrix); $j++) {

if ($distanceMatrix[$i][$k]+$distanceMatrix[$k][$j] < $distanceMatrix[$i][$j]) {

$distanceMatrix[$i][$j] = $distanceMatrix[$i][$k]+$distanceMatrix[$k][$j];

$pathMatrix[$i][$j] = $pathMatrix[$k][$j];

}

}

}

}

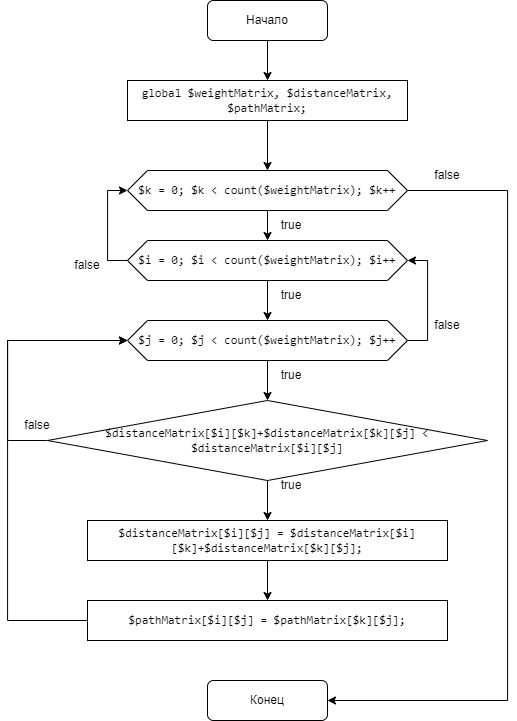


Рисунок 1. Блок-схема поиска кратчайшего пути.

После этого происходит поиск кратчайшего пути и строиться массив индексов для вывода пути на экран.

global $distanceMatrix, $pathMatrix, $shortestPath, $startIndex, $endIndex;

$nodeEnd = $endIndex;

if ($distanceMatrix[$startIndex][$endIndex] == INF) {

echo "<p>Путь не найден.</p>";

die();

}

while ($nodeEnd != -1) {

array\_push($shortestPath, $nodeEnd+1);

$nodeEnd = $pathMatrix[$startIndex][$nodeEnd];

}

$shortestPath = array\_reverse($shortestPath);

После полученная информация выводится на экран.

echo "<p>Путь до вершины: ";

for ($i = 0; $i < count($shortestPath); $i++){

echo $shortestPath[$i]." ";

}

echo "</p>";

echo "Расстояние до вершины: ".$distanceMatrix[$startIndex][$endIndex];

Графический интерфейс сайта выглядит так:

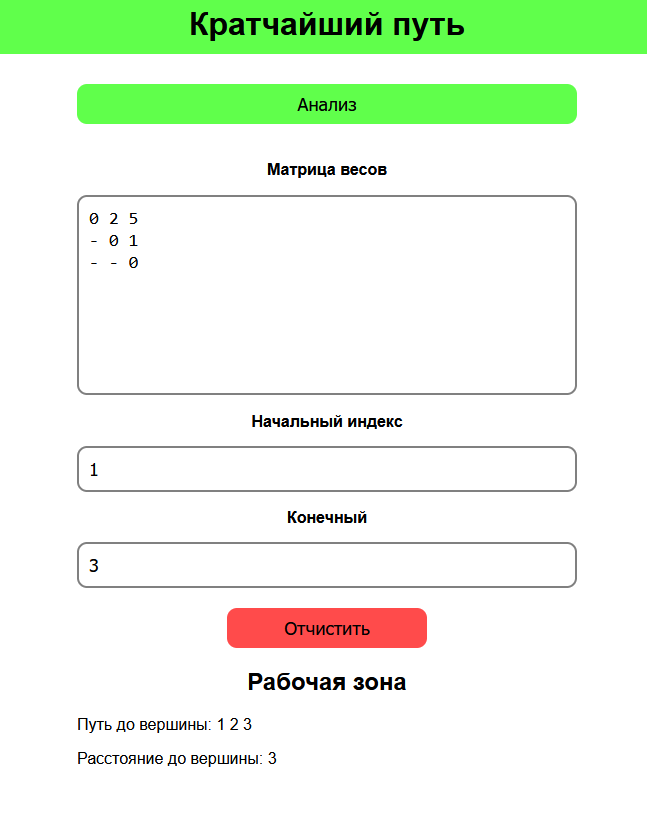
****

Рисунок 2. Интерфейс сайта.

# Вывод

Выполнив данную лабораторную работу, я создал сайт, который может находить кратчайший путь в графе и выводить его стоимость, а также проверять ввод на ошибки.