Министерство образования и науки Российской Федерации

ФГБОУ ВО «Ульяновский государственный технический университет»

Кафедра «Вычислительная техника»

Лабораторная работа №4

«Графы»

Выполнил студент

группы ИВТАПбд-11

Мухаметзянов Т. А.

Ульяновск, 2022

**Оглавление**

[**Постановка задачи 2**](#_Toc103805388)

[**Реализация 3**](#_Toc103805389)

[**Блок – схема 4**](#_Toc103805390)

[**Интерфейс работы 5**](#_Toc103805391)

[**Листинг кода 6**](#_Toc103805392)

[**Выводы 9**](#_Toc103805393)

[**Литература 10**](#_Toc103805394)

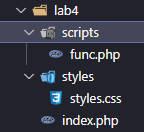
# Постановка задачи

Разработать программу на php нахождения кратчайшего пути в графе между двумя вершинами. Вид графа определяется заданием. Алгоритм определяется самостоятельно.

# Реализация

Структура

1. PHP страница для пользователя
2. PHP файл с основным скриптом для корректной работы страницы
3. CSS файл со всеми стилями



Лабораторная работа реализована в 3 файла (php страница, css, php). На сайте пользователь вводит начальную вершину, конечную, а также сами элементы графа через пробел.

1. Если ввод не проходит валидацию, тогда пользователю выводит ошибку, которую необходим исправить
2. Если ввод корректен, тогда пользователь получает матрицу конечных путей и кратчайший путь от начальной до конечной вершины.

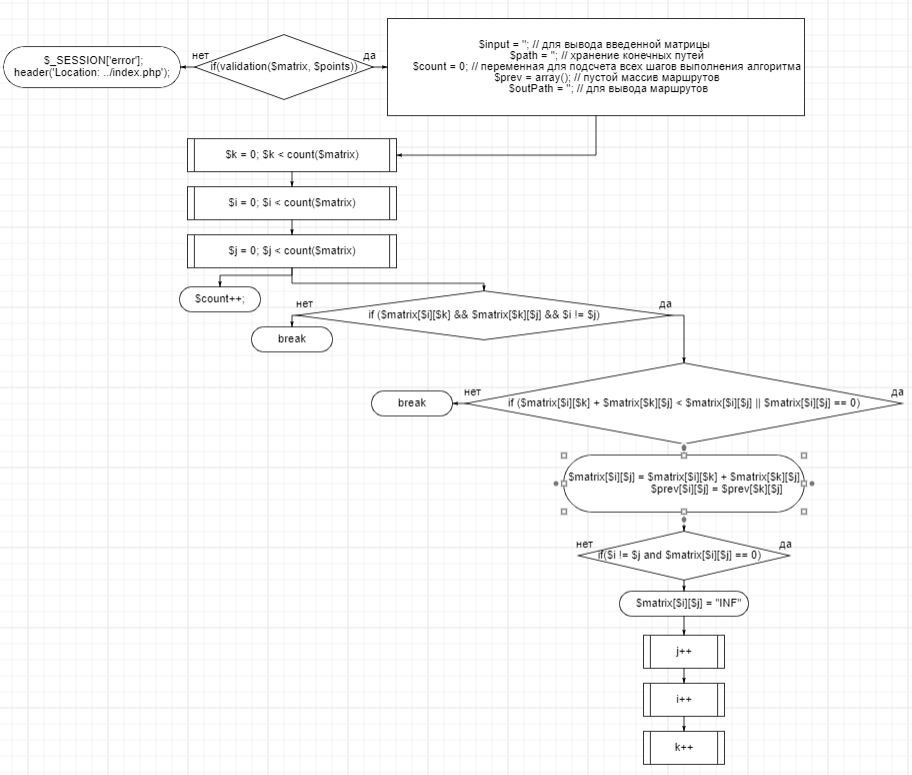
Для отображения матрицы конечных путей и матрицы маршрутов использован алгоритм Флойда-Уоршелла.

Если путь отсутствует значение 0 меняется на INF. Также алгоритм записывает количество всех шагов его выполнения.

Сайт состоит из номера л\р, а также варианта, полей ввода количества вершин графа и самого графа, кнопки подтверждения (связанная с основной функцией на PHP) и абзацев для вывода результатов.

# Блок – схема

Алгоритм Флойда-Уоршелла



# Интерфейс работы

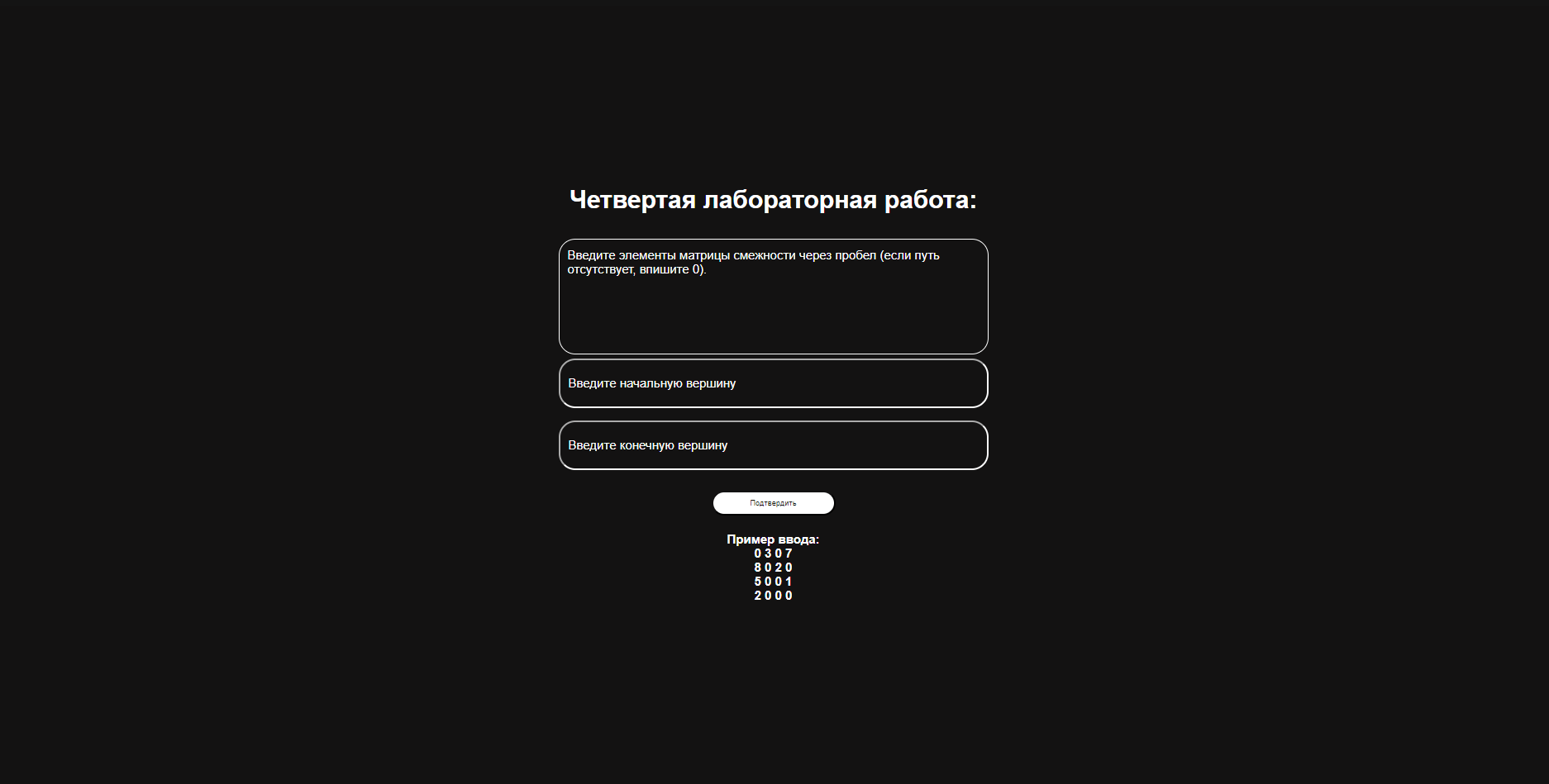


Рис 1 – Вид страницы



Рис 2 – Ошибка валидации



Рис 3 – Результаты

# Листинг кода

PHP

<?php

    session\_start();

    // считывание матрицы && преобразование строки в массив

    $matrix = preg\_split('/[\r\n]+/', $\_POST['array']);

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

        $matrix[$i] = trim($matrix[$i]);

        $matrix[$i] = explode(" ", $matrix[$i]);

    }

    // считывание стартовой и конечных вершин

    $start = (*int*)$\_POST['first'];

    $last = (*int*)$\_POST['last'];

    //функция валидации

    function validation($matrix, $start, $last) {

        $\_SESSION['textError'] = "";

        if(empty($start) or empty($last) or empty($matrix)) {

            $\_SESSION['text'] = "Не должно быть ни одного пустого поля!";

            return false;

        }

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

            if (count($matrix) != count($matrix[$i])) {

                $\_SESSION['text'] = "Матрица должна быть квадратной!";

                return false;

            }

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

                if($i == $j and $matrix[$i][$j] != 0) {

                    $\_SESSION['text'] = "На главной диагонали должны быть нули!";

                    return false;

                }

                if(!is\_numeric($matrix[$i][$j])) {

                    $\_SESSION['text'] = "Матрица должна состоять из цифр!";

                    return false;

                }

            }

        }

        if ($start > count($matrix) or $last > count($matrix)) {

            $\_SESSION['text'] = "Номер вершины не совпадает с количеством вершин";

            return false;

        }

        if ($start == 0 or $last == 0 or $start < 0 or $last < 0) {

            $\_SESSION['text'] = "Номер вершины не совпадает с количеством вершин";

            return false;

        }

        if (!is\_numeric($start)) {

            $\_SESSION['text'] = "Номер вершины должен быть цифрой!";

            return false;

        }

        if (!is\_numeric($last)) {

            $\_SESSION['text'] = "Номер вершины должен быть цифрой!";

            return false;

        }

        return true;

    }

    if(validation($matrix, $start, $last)) {

        $start = (*int*)$\_POST['first'] - 1; // стартовая вершина

        $last = (*int*)$\_POST['last'] - 1; // конечная вершина

        $matrixOut = ''; // для вывода матрицы смежности

        $path = array(); // массив маршрутных вершин

        $count = 0; // переменная для подсчета всех шагов выполнения алгоритма

        // инициализация массива

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

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

                if($i == $j) {

                    $R[$i][$j] = -1;

                } else {

                    $R[$i][$j] = $j;

                }

            }

        }

        //алгорит Флойда-Уоршелла для нахождения минимального пути между каждой парой элементов

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

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

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

                    $count++;

                    if ($matrix[$i][$k] && $matrix[$k][$j] && $i != $j) {

                        if ($matrix[$i][$k] + $matrix[$k][$j] < $matrix[$i][$j] || $matrix[$i][$j] == 0) {

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

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

                        }

                    }

                }

            }

        }

        // вычисление кратчайшего маршрута от start до last

        array\_push($path, $start + 1);

        if ($R != -1) {                 // если есть путь, то заносим в массив

            while ($start != $last) {

                $start = $R[$start][$last];

                array\_push($path, $start + 1);

            }

        }

        // замена всех 0 не находящихся на главной диагонали на бесконечность (INF)

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

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

                if($i == $j) {

                    $matrix[$i][$j] = "INF";

                }

                $matrixOut = $matrixOut.$matrix[$i][$j]." ";

            }

            $matrixOut = $matrixOut."<br>";

        }

        //вывод пути

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

            $\_SESSION['path'] = "Маршрут из " . $path[0]. " вершины до " . $path[count($path) - 1] . ":<br>" . implode(' - ', $path) . "";

        }

        $\_SESSION['outMatrix'] = "Матрица конечных путей :<br>" . $matrixOut . "<br>Количество шагов выполнения алгоритма: <br>" . $count . "<br>";

        header('Location: ../index.php');

    }

    else {

        header('Location: ../index.php');

    }

?>

# Выводы

С помощью данной лабораторной работы были изучены построение матрицы кратчайших путей между парами всех вершин и матрицы маршрутов всех пар вершин по алгоритму Флойда-Уоршелла.

# Литература

1. Р. Хаггарти. Дискретная математика для программистов.
2. https://myslide.ru/presentation/skachatalgoritmizaciyaiprogrammirovanieyazykCcelochislennyealgoritmy3845
3. https://kvodo.ru/algoritm-floyda-uorshella.html