**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 4](#_Toc135921158)

[1 Техническое задание 4](#_Toc135921159)

[1.1 Введение 4](#_Toc135921160)

[1.2 Основание для разработки 4](#_Toc135921161)

[1.3 Назначение 4](#_Toc135921162)

[1.4 Требования к программе или программному изделию 4](#_Toc135921163)

[1.4.1 Требования к программе или программному изделию 4](#_Toc135921164)

[1.4.2 Требования к надёжности 4](#_Toc135921165)

[1.4.3 Требования к составу и параметрам технических средств 4](#_Toc135921166)

[1.4.4 Требования к информационной программной совместимости 4](#_Toc135921167)

[1.4.5 Требования к графическому дизайну 4](#_Toc135921168)

[1.5 Стадии и этапы разработки 4](#_Toc135921169)

[1.6 Порядок контроля и приёмки 4](#_Toc135921170)

[2 Алгоритм задачи 4](#_Toc135921171)

ВВЕДЕНИЕ

Решение широкого круга задач требует умения разрабатывать и создавать веб проекты, работать в команде, разрабатывать прикладные программы решения задач математического моделирования, создавать документацию и тесты, соответствующие проекту.

Задачей учебной практики по ПМ 02 «Осуществление интеграции программных модулей» является закрепление теоретического материала междисциплинарных курсов ПМ 02 и получение практических навыков.

# Техническое задание

## Введение

Настоящее техническое задание распространяется на разработку веб-сайта по решению задач, связанных с алгоритмами обхода графов. Предполагается, что сайт будет использоваться студентами и преподавателями во время прохождения соответствующих учебных тем, а также для людей интересующихся данной темой.

Система поможет проверить правильность нахождения остовных деревьев с помощью различных алгоритмов:

* Алгоритм поиска в ширину;
* Алгоритм поиска в глубину;
* Алгоритм Краскала для нахождения кратчайшего остовного дерева.

Также необходимо реализовать главную страницу, на которой будет описание функциональности сайта, с навигацией на решение определённым алгоритмом и страницу «Об авторах», с фотографиями разработчиков, ФИО и кратким пояснением о личном вкладе в проект приложения.

Реализация системы на базе фреймворка Bottle и языка программирования Python. Данный выбор обусловлен большим опытом работы команды разработки с ним, хорошей документацией и множеством встроенных инструментов, таких как удобная система шаблонов и обработка форм, ускоряющих и упрощающих работу с веб частью разработки и задающих общую архитектуру приложения.

## Основание для разработки

Задание на учебную практику по «Интеграции программных модулей».

## Назначение

Система предназначена для решения задач построения остовных деревьев, с помощью алгоритмов обходов графов в ширину, глубину и алгоритмом Краскала.

## Требования к программе или программному изделию

### Требования к программе или программному изделию

Система должна обеспечивать следующий функционал:

* Генерация случайных входных данных;
* Сохранение файла с решением;
* Выполнение алгоритма Краскала для нахождения кратчайшего остовного дерева;
* Удобный ввод и демонстрация введённых данных.

Исходные данные:

* матрица схема рёбер.

Выходные данные:

* матрицы смежности полученных деревьев.

### Требования к надёжности

* Контроль ввода данных пользователем;
* Обработка некорректных действий со стороны пользователя.

### Требования к составу и параметрам технических средств

* Система должна работать на персональных компьютерах и мобильных устройствах.

### Требования к информационной программной совместимости

* Веб-сайт должен работать в современных браузерах, таких как Chrome, Opera, Yandex Browser (основанные на Chromium выше 100 версии) и Firefox (выше 70 версии).

### Требования к графическому дизайну

* Соответствие предметной области;
* Лаконичность;
* Понятность.

## Стадии и этапы разработки

* Анализ требований – 25 мая 2023 года;
* Составление ТЗ – 25 мая 2023 года;
* Проектирование – 25 мая - 27 мая 2023 года;
* Программирование – 27 мая - 7 июня 2023 года;
* Отладка и тестирование – 27 мая - 7 июня 2023 года.

## Порядок контроля и приёмки

На этапе испытаний нужно проверить следующие функции:

* Проверить наличие защиты от необдуманных и случайных действий пользователя;
* Проверить корректность тестов, что все подсчеты в программе производятся правильно;
* Проверить выполнение целевой задачи;
* Прием программы должна осуществляться руководителем учебной практики.

# Алгоритм задачи

Формулировка задачи:

Требуется найти кратчайший остов данного графа с помощью алгоритма Краскала. Матрица весов рёбер дана по условию.

Словесное описание алгоритма:

Остовное дерево (остов) — это подграф данного графа, содержащий все его вершины и являющийся деревом.

Алгоритм Краскала, также алгоритм Крускала — эффективный алгоритм построения минимального остовного дерева взвешенного связного неориентированного графа.

Алгоритм Краскала:

* Создать пустое минимальное остовное дерево.
* Отсортировать все ребра графа по возрастанию весов.
* Для каждого ребра (u, v) из отсортированного списка:
  + Если добавление ребра (u, v) не создаст цикл в минимальном остовном дереве, то добавить его в минимальное остовное дерево.
  + Иначе, проигнорировать ребро и перейти к следующему.
* Вернуть минимальное остовное дерево.

В результате получится минимально остновное дерево.

Пример:

Найти для представленного на рисунке 1 графа, кратчайший остов.

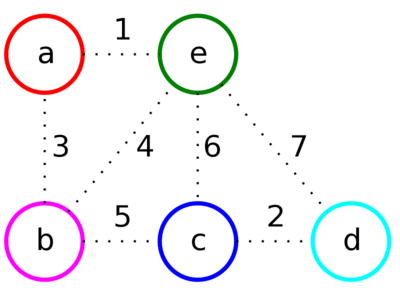


Рисунок 1 - Исходный граф

Шаг один:

Первое ребро, которое будет рассмотрено — ae, так как его вес минимальный.

Добавим его к ответу, так как его концы соединяют вершины из разных множеств (a — красное и e — зелёное).

Объединим красное и зелёное множество в одно (красное), так как теперь они соединены ребром.

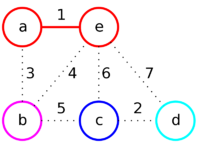


Рисунок 2 – После первого шага

Шаг два:

Рассмотрим следующие ребро — cd.

Добавим его к ответу, так как его концы соединяют вершины из разных множеств (c — синее и d — голубое).

Объединим синее и голубое множество в одно (синее), так как теперь они соединены ребром.

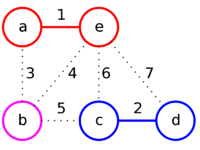


Рисунок 3 – После второго шага

Шаг три:

Дальше рассмотрим ребро ab.

Добавим его к ответу, так как его концы соединяют вершины из разных множеств (a — красное и b — розовое).

Объединим красное и розовое множество в одно (красное), так как теперь они соединены ребром.

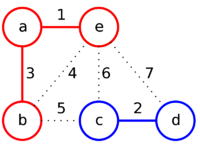


Рисунок 4 – После третьего шага

Шаг 4:

Рассмотрим следующие ребро — be.

Оно соединяет вершины из одного множества, поэтому перейдём к следующему ребру bc

Добавим его к ответу, так как его концы соединяют вершины из разных множеств (b — красное и c — синее).

Объединим красное и синее множество в одно (красное), так как теперь они соединены ребром.

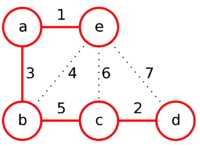


Рисунок 5 – После 4 шага

Как мы можем увидеть на рисунке 5, все вершины соединены между собой и не создают цикл, это значит, что мы нашли кратчайшее остовное дерево: e-a-b-c-d.

На рисунке 6 представлена блок-схема алгоритма Краскала.



Рисунок – Блок схема алгоритма

Комментарий к рисунку 6:

\* - Вершины уже есть в пустом остовном дереве?

# Разработка UML-диаграммы Use Case и компонентов приложения

Диаграмма вариантов использования – диаграмма, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей. Диаграмма use-case для проекта представлена на рисунке 7, жёлтым цветом отмеч

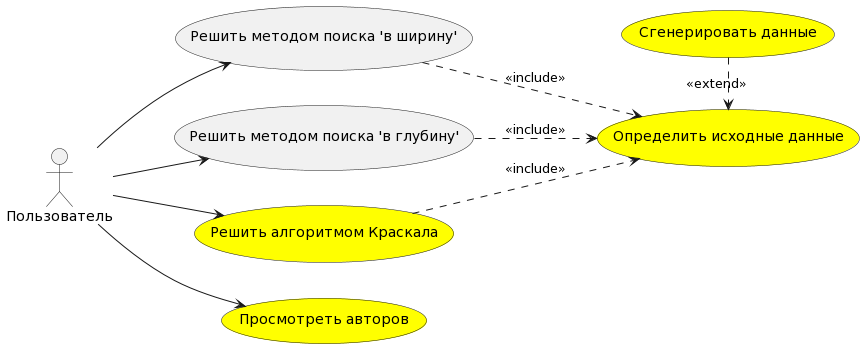


Рисунок 7 – use-case диаграмма

Диагра́мма компоне́нтов — элемент языка моделирования UML, статическая структурная диаграмма, которая показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п. Диаграмма компонентов для проекта представлена на рисунке 8.

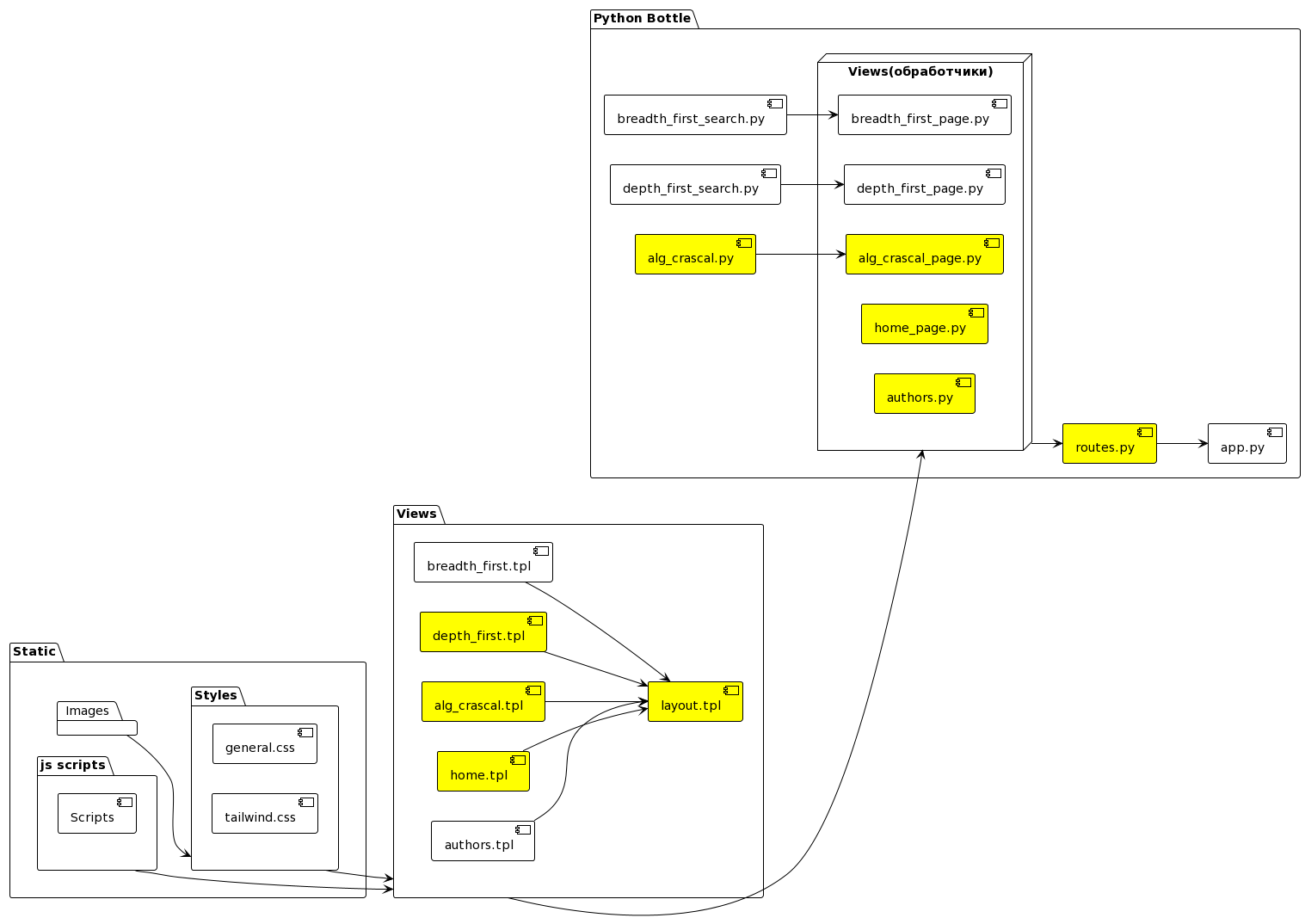


Рисунок – Диаграмма компонентов