Тестовое задание

Задание №1.

Имеется корневая папка. В этой папке могут находиться текстовые файлы, а также другие папки. В других папках также могут находится текстовые файлы и папки (уровень вложенности может оказаться любым). Найти все текстовые файлы, отсортировать их по имени и склеить содержимое в один текстовый файл.

В каждом файле может быть ни одной, одна или несколько директив формата:

require ‘<путь к другому файлу от корневого каталога>’

Директива означает, что текущий файл зависит от другого указанного файла. Необходимо выявить все зависимости между файлами, построить сортированный список, для которого выполняется условие: если файл A, зависит от файла B, то файл A находится ниже файла B в списке. Осуществить конкатенацию файлов в соответствии со списком. Если такой список построить невозможно (существует циклическая зависимость), программа должна вывести соответствующее сообщение. В случае циклической зависимости вывести объяснение ошибки - указать цикл зависимостей между файлами.

☝ **Реализация должна быть написана на любом из следующих языков: C, C++, Java, Kotlin, Rust, Python**

➡️ Пример. Дана структура файлов и каталогов:

- Каталог “Folder 1”
    - Файл “File 1-1”. Содержимое:

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Suspendisse id enim euismod erat elementum cursus. In hac
habitasse platea dictumst. Etiam vitae tortor ipsum. Morbi
massa augue, lacinia sed nisl id, congue eleifend lorem.

require ‘Folder 2/File 2-1’
Praesent feugiat egestas sem, id luctus lectus dignissim ac.
Donec elementum rhoncus quam, vitae viverra massa euismod a.
Morbi dictum sapien sed porta tristique. Donec varius
convallis quam in fringilla.
```

- Каталог “Folder 2”
    - Файл “File 2-1”. Содержимое:

```
Phasellus eget tellus ac risus iaculis feugiat nec in eros. Aenean in luctus ante. In lacinia lectus tempus, rutrum ipsum quis, gravida nunc. Fusce tempor eleifend libero at pharetra. Nulla lacinia ante ac felis malesuada auctor. Vestibulum eget congue sapien, ac euismod elit. Fusce nisl ante, consequat et imperdiet vel, semper et neque.
```

    - Файл “File 2-2”. Содержимое:

```
require 'Folder 1/File 1-1'

require 'Folder 2/File 2-1'

In pretium dictum lacinia. In rutrum, neque a dignissim maximus, dolor mi pretium ante, nec volutpat justo dolor non nulla. Vivamus nec suscipit nisl, ornare luctus erat. Aliquam eget est orci. Proin orci urna, elementum a nunc ac, fermentum varius eros. Mauris id massa elit.
```

💡 Для указанной структуры каталогов и файлов должен быть построен список:

```
Folder 2/File 2-1

Folder 1/File 1-1

Folder 2/File 2-2
```

Задание №2.

Напишите Todo List в соответствии с макетом (макет приблизительный, вы вольны придумать свое дизайнерское решение).

☝ Для выполнения задания, вам потребуется обращаться к серверу посредством API. Ознакомьтесь с документацией.

☝ **В ходе выполнения помимо JavaScript из сторонних библиотек можно использовать только jQuery и jQuery UI.**

☝ **Возможно, понадобится написать серверную часть для обхода CORS-ограничений. Эту часть решения можно писать на любом ЯП.**

☝ Что должно быть реализовано:

1. Поиск задач по названию.
2. Календарь с возможностью выбора даты.
3. Кнопка для вывода задач на сегодняшнюю (текущую) дату.
4. Кнопка для вывода задач на текущую неделю.
5. Возможность сортировать список задач по дате.
6. Возможность вывода только невыполненных задач.
7. Возможность открывать полное описание задачи (например, в модальном окне).
8. В календаре можно выбрать диапазон дат.
9. Поиск с выпадающим списком найденных задач, по нажатии на элементы которого открывается полное описание задачи.

💡**Перед выполнением заданий, представленных ниже, необходимо выполнить следующие шаги:**

0. Ознакомиться с курсом вебинаров по Z8, как минимум с 1-м:
https://youtu.be/YML9VJwtkFs

1. Склонировать репозиторий фреймворка Z8 (https://github.com/zenframework/z8) и примера приложения на Z8 - Z8 Template (https://github.com/zenframework/z8-template).

2. С помощью документации
- https://github.com/zenframework/z8/wiki
- https://github.com/zenframework/z8/wiki/2.-Импорт-Z8-Template
собрать и запустить проект Z8 Template.

3. Разобраться, каким образом осуществляется расширение стандартных JavaScript-компонентов, см. в проекте Z8 Template:
- org/zenframework/z8/template/view/Documents.bl
- src/main/js/main/js/ui/control/EMail.js

4. Изучить исходные коды JavaScript-части фреймворка (подпроект org.zenframework.z8.js), разобраться, как устроены стандартные JavaScript-компоненты Z8.

Задание №3

В языке BL, помимо StringField (см. org/zenframework/z8/template/view/Document.bl) с ограничением длины значения, предусмотрено поле TextField для хранения текста неограниченной длины.
В классе Document.bl добавить поле "xml" типа TextField для хранения XML-текста документа, добавить поле "xml" в массив controls в классе Documents.bl.
Реализовать JavaScript-расширение стандартного компонента Z8.form.field.TextArea (js/form/field/TextArea.js) таким образом, чтобы он подсвечивал синтаксис XML-текста документа в поле "xml".
(В Documents.bl использовать атрибут [ui], аналогично тому, как это сделано для поля "email".)
Допускается использование JavaScript-библиотек с открытым исходным кодом, реализующих подсветку синтаксиса, например, highlight.js, CodeMirror или аналогов.

Задание №4

В классе Document.bl есть поле doc типа FileField, хранящее ссылку на файл. Предполагая, что в это поле загружается аудиофайл (любой формат по выбору соискателя), реализовать JavaScript-расширение стандартного компонента Z8.form.field.Text (js/form/field/Text.js) таким образом, чтобы он встраивал в форму проигрыватель аудиофайлов, ссылки на которые хранятся в поле "doc".
(В Documents.bl использовать атрибут [ui], аналогично тому, как это сделано для поля "email".)

💡Оценка выполнения

Мы будем обращать внимание на полноту решения задания (1), чистоту кода (2), понимание принципов ООП (3) и работу с GitHub (4).

Решение будем ожидать на Github с README файлом, в котором будет краткое описание хода выполнения каждого из заданий, перечисление используемых сторонних библиотек (если есть) и пр.