=begin

ПРОГРАММА АНАЛИЗА ВВЕДЕННОГО ТЕКСТА.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

I раздел. Ruby.

Задачи:

- обработать ошибки ввода:

- пустой ввод;

- не английский язык;

- вывести исходный текст;

- посчитать количество знаков;

- посчитать количество пробелов;

- посчитать количество цифр;

- вывести введенные цифры строкой;

- вывести только четные цифры;

- вывести только нечетные цифры;

- сравнить цифры и вывести наибольшую;

- для наибольшей цифры посчитать факториал;

- посчитать частоту цифр;

- посчитать сумму использованных цифр;

- посчитать произведение использованных цифр;

- посчитать количество символов (не букв и цифр);

- вывести символы (не буквы и цифры) сплошной строкой;

- вывести символы (не буквы и цифры) строкой через точки (без пробелов);

- посчитать большие буквы;

- вывести большие буквы;

- посчитать маленькие буквы;

- вывести маленькие буквы;

- убрать из исходного текста цифры и символы (не буквы и цифры), сделать все буквы маленькими и вывести все буквы;

- инвертировать порядок букв и вывести;

- посчитать частоту букв и вывести на экран;

- вывести на экран частоту только упоминаемых букв;

- создать структуру каталогов из Ruby;

- создать файл и сохранять (дополнять) в него результаты по мере выполнения заданий;

- создать файл с чеклистом и по мере выполнения заданий (прохождения тестов) отмечать в нем выполнение;

- все задания должны иметь тест;

- использовать github для хранения кода на всех этапах выполнения заданий;

- задания должны быть branch и при выполнении сливаться с master;

- код должен быть документирован.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

II раздел. Ruby on Rails.

Задачи:

Создать для проекта веб-интерфейс.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

III раздел. Условия выполнения.

Требования по использованию ruby:

- модули;

- классы и суперклассы;

- структуры и OStruct;

- блоки кода;

- наследование;

- для размещения классов использовать отдельные файлы;

- примеси;

- методы:

- доступа (геттеры, сеттеры, асессоры);

- to\_s;

- super;

- initialize;

- Math;

- Time;

- Proc;

- lambda;

- диапазоны;

- массивы;

- хэши и символы;

- вложенный хэш;

- вложенный масссив;

- иттератор;

- условные операторы:

- if;

- unless;

- case;

- циклы:

- for;

- while;

- until;

- рекурсия;

- переменные:

-локальные;

- глобальные;

- класса;

- экземпляра;

- модификаторы доступа:

- public;

- private;

- protected.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

IV раздел. План реализации.

Реализация:

1.

2.

3.

4.

5.

6.

7.

8.

9.

10.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

V раздел. Архитектура и пространство имен.

1. Разработка иерархии классов (каркаса кода).

2. Определение имен классов и файлов.

3. Схема каталогов проекта.

4. Определение переменных и их имен по маске:

- классы/методы: имя-тип объекта (класс/метод/переменная)-файл-модификатор доступа-пояснение/назначение-суперкласс-подчиненные классы

- переменные: имя-

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

VI раздел. Код.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

=end