# Вариант 6 конкурсных заданий *практического* этапа Конкурса

В виртуальном мире видеоигр произошла непредвиденная ситуация. Внезапный программный сбой нарушил работу всех игровых персонажей, сделав их поведение хаотичным и непредсказуемым. Игровое сообщество столкнулось с массой проблем, поскольку многие игры стали практически непроходимыми из-за возникающих ошибок и лагов. Ваша задача разработать программу, способная обнаруживать и исправлять подобные ошибки, возвращая игровым персонажам должное поведение и делая прохождение игр вновь комфортным и приятным.

Вы готовы приступать к задачам, но сначала необходимо **создать репозиторий** для проекта и задач, которые Вы будете выполнять. Обязательно сделайте его **PUBLIC**, а то мы не сможем проверить решение вашей последней задачи. Каждую задачу вам необходимо будет правильно оформить и залить в репозиторий. *Код, который Вы напишете необходимо задокументировать, чтобы другие программисты могли понять, что делает код и за что отвечает. Примеры документирования приведены ниже.*

**Пример документирования кода на языке С++**

/\*\*

\* Это описание функции foo

\*

\* @param str это описание аргумента str

\* @param pattern это описание аргумента pattern

\* @return это описание того, что вернет функция

\*/

int foo(std::string str, std::string& pattern)

{

...

};

**Пример документирования на языке Python**

def complex(real=0.0, imag=0.0):

"""Описание функции complex.

Описание аргументов:

real – описание аргумента

imag – описание аргумента

"""

if imag == 0.0 and real == 0.0: return complex\_zero

...

Теперь можете приступать к решению задач!

К задачам прикреплен файл *game.txt,* который хранит в себе информацию о название игры; имени персонажа, имени ошибки, дате фиксации бага.

Столбцы: *GameName, characters, nameError(буквенное значение:числовое значение), date(ГГГГ-ММ-ДД).*

Разделитель *«$».*

***Задача 1.***

Первая задача, которая стоит перед вами это найти все ошибки с содержанием числа 55, тк это особо важные и опасные ошибки. Для этого составьте отчет в формате: “У персонажа\t<characters>\tв игре\t<GameName>\tнашлась ошибка с кодом:\t <nameError>.\tДата фиксации:\t <date>”. После предоставления отчета измените значение ошибки на “Done”, а в поле дата поставьте “0000-00-00” и полученные измененные данные сохраните в файле game\_new.csv.

*В задаче запрещено использование сторонних библиотек(Pandas и др)*

*Не забудьте сделать комментарии к коду согласно стандартам документирования кода выбранного языка (для языка Python – PEP 257). После выполнения необходимо сделать локальные и удаленные изменения Вашего репозитория.*

***Задача 2***

У вас снова запросили отчет по ошибкам в играх. Теперь отчет необходимо предоставить о количестве багов в каждой игре. Для этого отсортируйте данные из файла *game.txt* по столбцу игры в алфавитном порядке с помощью быстрой сортировки. После этого предоставьте отчет в формате:  
<Игра 1> - количество багов: <count>

….

<Игра N> - количество багов: <count>

Где N - количество уникальных игр.

*Не забудьте сделать комментарии к коду согласно стандартам документирования кода выбранного языка. После выполнения необходимо сделать локальные и удаленные изменения Вашего репозитория*

***Задача 3***

**Ввод: стандартный ввод**

**Вывод: стандартный вывод**

Пришло время сделать наработку для интерфейса, который будет взаимодействовать с базой данных. Для этого Вам необходимо написать консольную программу, которая будет запрашивать у Вас имя персонажа, а на выход будет выдавать список игр, в которых встречается этот персонаж(ограничьте количество вывода элементов до 5), если ничего не найдено будет выводить: “Этого персонажа не существует”. Программа должна всегда запрашивать название. Прекратить свою работу она сможет только после ввода “game”.

Поиск необходимо осуществить с помощью двоичного алгоритма поиска**.**

**Обратите внимание, что данные в файле не отсортированы! Для сортировки воспользуйтесь встриоенными сортировками или напишите собственный алгоритм.**

**Формат ответа на запрос пользователя:**

“Персонаж <characters> встречается в играх:

<Игра 1>

…

<Игра 5>

и др.”

Строчка “и др.” должна писать только в случае, если значений больше чем пять.

Поиск необходимо осуществлять в файле game.txt

*Не забудьте сделать комментарии к коду согласно стандартам документирования кода выбранного языка. После выполнения необходимо сделать локальные и удаленные изменения Вашего репозитория*

***Задача 4***

При просмотре логов вы увидели, что в одной игре может быть несколько ошибок. Вам необходимо добавить счетчик того, сколько ошибок выявлено в игре. Реализуйте методы/функции, которые будут считать количество упоминаний какой-либо игры. Полученный список игр необходимо отсортировать в порядке возрастания. При занесении данных в csv файл, счетчик необходимо применить ко всем строкам.

“Starfield, Mila, MВКJ:537, 1233-11-19” → “ Starfield, Mila, MВКJ:537, 1233-11-19, 9”

*На вход подается файл game.txt, который необходимо записать в список, добавить счетчик для каждого элемента, после чего дополнить список счетчиками. Последним этапом полученный список записать в новый game\_counter.csv файл. Назвать новый столбец “counter”.*

*Не забудьте сделать комментарии к коду согласно стандартам документирования кода выбранного языка. После выполнения необходимо сделать локальные и удаленные изменения Вашего репозитория.*

***Задача 5***

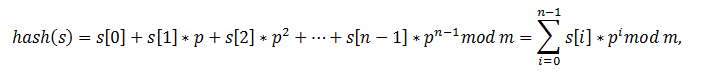
Когда мы начали исправлять ошибки, оказалось, что поиск по названию игры и персонажа неэффективен. Необходимо составить хэш-таблицу, в которой будет выстроено соответствие название игры + имя персонажа и значения их хэша. На основании этого необходимо составить хэш-таблицу, добавить полученный хэш нулевым столбом, результат записать в csv файл.

Строка для хэша генерируется следующим образом: Название игры + Имя персонажа

Примечание: в названии игры необходимо убрать все пробелы

Название игры: Starfield Персонаж: Mila → StarfieldMila

Для хэширования необходимо использовать следующий алгоритм.



где p и  m - некоторые выбранные положительные числа.

n – кол-во символов в строке

*Рекомендации по выбору чисел p и m.*

Для данного задания рекомендуется взять p = 65.

26\*2 – кол-во букв в английском алфавите строчных и прописных

10 – кол-во цифр.

2 – символы “:”,”-“, которые могут встретиться в названии игры.

1 – т.к. считаем не с 0.

m = 109+9

Для вычисления хэша строки  s, которая содержит только строчные буквы необходимо преобразовать каждый символ строки  s в целое число. Можно использовать преобразование  a →1,  b →2, … z → 26 . Преобразование  a → 0 не является хорошей идеей, поскольку тогда хэши строк  a,  aa,  aaa,  … все оцениваются как  0.

Для Заглавных буква следует использовать преобразование A → 27, B → 28 … Z → 52

Для цифр и доп символов. 1 → 53 … 0 → 63, : → 64, - → 65.

*На вход подается файл game.txt результаты необходимо записать в новый game \_with\_hash.csv файл.*

*Не забудьте сделать комментарии к коду согласно стандартам документирования кода выбранного языка. После выполнения необходимо сделать локальные и удаленные изменения Вашего репозитория.*

***Задача 6***

Ваш код будет использоваться программистом, которого возьмут на работу, поэтому он должен быть правильно оформлен и выложен на GitHub. Весь написанный код должен быть задокументирован согласно стандартам документирования кода выбранного языка.

Также необходимо оформить README.md для Вашего репозитория. Пункты, которые должны быть описаны:

1. Название проекта

2. Описание проекта

3. Оглавление (необязательно)

4. Как установить и запустить проект

5. Как использовать проект