# Вариант 19 конкурсных заданий *практического* этапа Конкурса

Астронавты, оказавшиеся на космической станции, ощутили, что время замерло в безмолвии космоса. Часы, которые обычно ярко сверкали и отсчитывали бесконечные минуты, теперь погрузились во мрак и несомненно потеряли свою силу. Они стали бесформенными и безжизненными, словно холодные камни, лишенные всякого движения. Астронавты, ошеломленные этим необычайным событием, осознали, что им придется восстановить работу этих часов, чтобы вернуться к нормальной жизни и ориентироваться во времени. Вы один из астронавтов, оказавшихся на космической станции. Примите участие и помогите всем остальным наладить ход времени.

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 *astronaut\_time.txt,* который хранит в себе информацию о номере часов; номере станции; номере каюты астронавта; времени остановки часов(ЧЧ:ММ:СС); кол-во часов простоя(целое число).

Столбцы: *WatchNumber, numberStation, cabinNumber, timeStop, count*

Разделитель *«>».*

***Задача 1.***

Восстановите время в каждой каюте. Для этого создайте новый файл формата csv с названием new\_time.csv, куда добавьте такие столбцы: WatchNumber, numberStation, cabinNumber, timeStop, timeNow.

Где значения из столбца рассчитываются по формуле: timeNow = timeStop + count.

Обратите внимание, что в часах 24 часа, 60 минут и 60 секунд.

После этого выведите действительное время для своей каюты номер 98-OYE в формате:

<timeNow> - действительное время для каюты: <cabinNumber>

*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***

Для комфортного пользования файлом вы решили его отсортировать по номерам станций. Для этого воспользуйтесь сортировкой с вычислительной сложностью алгоритма — O(n2). В задаче нельзя использовать встроенные сортировки! После сортировки выведите первые три станции, которым нужна помощь в формате:

На станции <numberStation> в каюте <cabinNumber> восстановлено время. Актуальное время: <timeNow>

Где актуальное время рассчитывается по формуле: timeNow = timeStop + count.

*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 пользователей номер каюты, а на выход будет выдавать время остановки и восстановленное время, если ничего не найдено будет выводить: “В этой каюте все хорошо”. Программа должна всегда запрашивать номер каюты. Прекратить свою работу она сможет только после ввода “none”.

Формат вывода ответа:

В каюте <cabinNumber> восстановлено время (время остановки: <timeStop>). Актуальное время: <timeNow>

Где timeNow = timeStop + count.

Поиск необходимо осуществить с помощью алгоритма с асимптотической сложностью O(n) в исходном файле.

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

***Задача 4***

Вы хотите понять в какое время чаще останавливались часы в каютах. Для этого сформируйте два списка, в одном из которых будет информация о станциях, где время остановки в промежутке от 00.00 до 12.00, а во втором от 12.01 до 23.59.

На основании сформированных списков выведите следующую информацию:

<длина списка 1> станций остановилось с период с 00.00 до 12.00.

<длина списка 2> станций остановилось с период с 12.01 до 23.59.

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

***Задача 5***

Чтобы поиск по каютам и станциям был эффективнее, создайте хэш-таблицу, в которой в качестве ключа будет использоваться номер каюты, а в качестве значения вся остальная информация о каюте.

Выведите информацию о первых 10 каютах в сформированной таблице.

*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