

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Лабораторная работа 2.25

Управление процессами в Python

Выполнил студент группы ИВТ-б-о-20-1

Симанский М.Ю « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

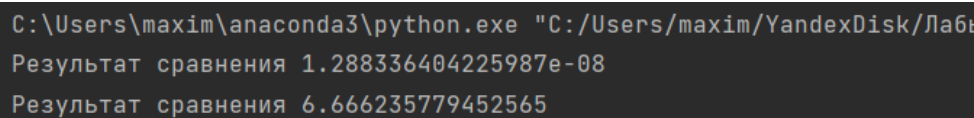
Проверил Воронкин Р.А. _____

(подпись)

Цель работы: приобретение навыков написания многозадачных приложений на языке программирования Python версии 3.x.

Задание 1

Для своего индивидуального задания лабораторной работы 2.23 необходимо реализовать вычисление значений в двух функций в отдельных процессах. Результат выполнения отображен на рисунке 1.



```
C:\Users\maxim\anaconda3\python.exe "C:/Users/maxim/YandexDisk/Лаб...
Результат сравнения 1.288336404225987e-08
Результат сравнения 6.666235779452565
```

Рисунок 1 – Результат выполнения работы

Ответы на контрольные вопросы

1. Lock-объект может находиться в двух состояниях: захваченное (заблокированное) и не захваченное (не заблокированное, свободное). После создания он находится в свободном состоянии. Для работы с Lock-объектом используются методы `acquire()` и `release()`. Если Lock свободен, то вызов метода `acquire()` переводит его в заблокированное состояние. Повторный вызов `acquire()` приведет к блокировке инициировавшего это действие потока до тех пор, пока Lock не будет разблокирован каким-то другим потоком с помощью метода `release()`. Вывоз метода `release()` на свободном Lock-объекте приведет к вы抛су исключения `RuntimeError`.

2. В отличии от рассмотренного выше Lock-объекта `RLock` может освободить только тот поток, который его захватил. Повторный захват потоком уже захваченного `RLock`-объекта не блокирует его. `RLock`-объекты поддерживают возможность вложенного захвата, при этом освобождение

происходит только после того, как был выполнен `release()` для внешнего `acquire()`. Сигнатуры и назначение методов `release()` и `acquire()` `RLock`-объектов совпадают с приведенными для `Lock`, но в отличие от него у `RLock` нет метода `locked()`. `RLock`-объекты поддерживают протокол менеджера контекста. С помощью команды закрытия `close()`.

3. Порядок работы с условными переменными выглядит так:

- На стороне `Consumer`'а: проверить доступен ли ресурс, если нет, то перейти в режим ожидания с помощью метода `wait()`, и ожидать оповещение от `Producer`'а о том, что ресурс готов и с ним можно работать. Метод `wait()` может быть вызван с таймаутом, по истечении которого поток выйдет из состояния блокировки и продолжит работу.

- На стороне `Producer`'а: произвести работы по подготовке ресурса, после того, как ресурс готов оповестить об этом ожидающие потоки с помощью методов `notify()` или `notify_all()`. Разница между ними в том, что `notify()` разблокирует только один поток (если он вызван без параметров), а `notify_all()` все потоки, которые находятся в режиме ожидания. Чтобы обновить данные в таблице, просто создайте соединение, затем создайте объект курсора с помощью соединения и, наконец, используйте оператор `UPDATE`.

4. При создании объекта `Condition` вы можете передать в конструктор объект `Lock` или `RLock`, с которым хотите работать. Перечислим методы объекта `Condition` с кратким описанием: `acquire(*args)` – захват объекта-блокировки. `release()` – освобождение объекта-блокировки. `wait(timeout=None)` – блокировка выполнения потока до оповещения о снятии блокировки. Через параметр `timeout` можно задать время ожидания оповещения о снятии блокировки. Если вызвать `wait()` на Условной переменной, у которой предварительно не был вызван `acquire()`, то будет выброшено исключение `RuntimeError`.

5. Чтобы перечислить все таблицы в базе данных `SQLite3`, вы должны запросить данные из таблицы `sqlite_master`, а затем использовать `fetchall()` для получения результатов из инструкции `SELECT`

6. При создании таблицы мы должны убедиться, что она еще не существует. Аналогично, при удалении/удалении таблицы она должна существовать. Чтобы проверить, не существует ли таблица уже, мы используем IF NOT EXISTS с оператором CREATE TABLE следующим образом.

7. Метод executemany можно использовать для вставки нескольких строк одновременно.

8. В базе данных Python SQLite3 мы можем легко хранить дату или время, импортируя модуль datetime . Следующие форматы являются наиболее часто используемыми форматами для datetime::