# Клещенко В

# ЛР #4: Параллельные алгоритмы

## Цель

Познакомить студента с инструментами, направленными на решение задач, использующих технологии распараллеливания.

## Задача

Моделирование нагрузки пользователей, которые обращаются почти одновременно на сервер (запросы выстраиваются в очередь).

**Дано**:

* Действие 1: Регистрация на портале (время обработки T1, нагрузка на процессор ~25%)
* Действие 2: Получение главной страницы (время обработки T2, нагрузка на процессор ~15%)
* Действие 3: Просмотр перечня зарегистрированных пользователей (время обработки T3, нагрузка на процессор ~1%)
* U1 – количество пользователей, которые отправляют запрос на действие 1
* U2 – количество пользователей, которые отправляют запрос на действие 2
* U3 – количество пользователей, которые отправляют запрос на действие 3

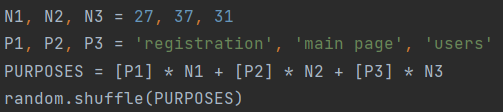
**Требуется**:

Составить программу, на которой смоделировать поведение обработки запросов пользователей. Рассмотреть вариант последовательного решения задачи (1) и параллельных вариантах (рассмотреть каждый), subprocess (2), threads (3), asyncio (4)

## Решение

Для моделирования нагрузки пользователей была использована клиент серверная архитектура, взаимодействие между частями которой осуществлялось посредством сокетов.

Изначально задавались следующие цели у клиентов – как в «дано»



«Практические одновременное» подключение пользователей к серверу осуществлялось за счет распараллеливание по потокам, для этого они подходили.

Изображение выглядит как текст

Автоматически созданное описание

На стороне сервера открывался слушающий сокет, принимающий все поступившие запросы и отправляющий их на выполнение либо последовательно, либо с использованием одного из способов распараллеливания.

Для эмулирования нагрузки на сервер, занимающей определенное время использовалась задержка по времени:

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

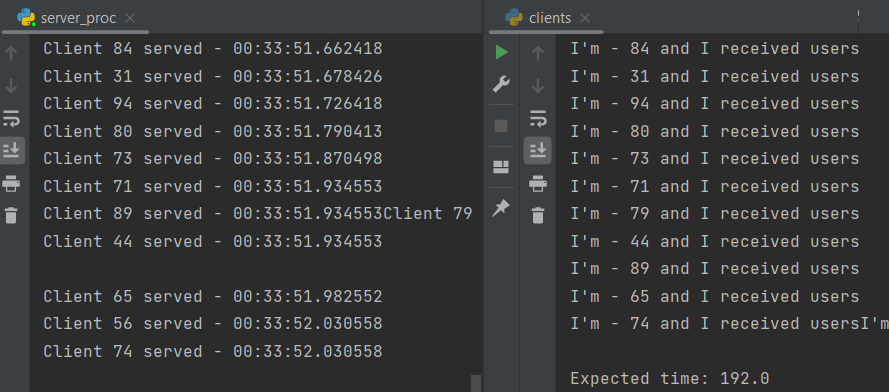
Автоматически созданное описание

Непосредственная реализация выполнена в программе, пример для последовательной работы:

Изображение выглядит как текст

Автоматически созданное описание

И для параллельной:



## Вывод

Распараллеливание дает очевидное преимущество перед последовательным выполнением: например, даже для 100 клиентов: 7 секунд против 192.

Потоки, асинхронный: 2.5 – 3

Процессы: 7 секунд

С точки зрения выбора способа распараллеливания – на таком масштабе это не ощутимо, но можно отметить, что при большом количестве клиентов потоки будут тратить больше времени на переключение, а количество процессов ограничено количеством ядер, но на создание процесса требуется дополнительное время, в то время как асинхронный способ помогает выполнять действия параллельно в цикле событий.