**Лабораторная работа 3**

2 балла

Дан пример в виде файла lab1.go, который измеряет время доступа к кэшу и оперативной памяти, в случайном порядке итерируясь по элементам массивов разных размеров. Пример можно найти здесь: https://github.com/MikeBasanets/latency-test

Пример при работе выводит результат в виде двух столбцов, например:

5000 1.95686561

6000 1.94325336

7200 1.94303685

8640 1.94442978

10368 1.97553434

12441 1.97510052

14929 1.99337019

17915 2.28430303

21499 3.02394246

25798 3.23472403

Левый столбец– это размер массива в байтах, правый — время доступа в наносекундах.

Нужно запустить программу из примера (при помощи команды go run lab1.go) и сохранить результат. Затем построить график (при помощи программы plot.py из примера, она считывает результат из файла data.txt и строит график при помощи matplotlib).  
Пример можно запустить скриптом run.sh.

3 балла

Проделать все те же действия, но написать программу для итераций по массиву самостоятельно на любом языке (кроме JavaScript, Python), скрипт для графика использовать из примера. Генерацию случайных чисел для каждой итерации использовать не стоит, это может быть медленно.

2 балл

Скачайте утилиту CPU-Z (она для Windows, если у вас Мак/Линукс, можете запускать свою программу и утилиту на местных компьютерах). Найдите там информацию об уровнях кэша, сравните со своим графиком. Объясните полученные результаты.

3 балла

Беседа по следующим вопросам:

1. Что такое кэш и зачем он нужен?
2. Иерархия памяти.
3. Ассоциативность кэша (полностью ассоциативный, множественно-ассоциативный, с прямым отображением).
4. Политики записи кэш-памяти (write back, write-through).
5. Алгоритмы вытеснения ( LRU, LFU и др.).