

Московский авиационный институт
(национальный исследовательский университет)
Институт № 8 «Информационные технологии и прикладная математика»

Лабораторная работа № 2

Тема: Потоки

Выполнил студент группы М8О-208Б-23

Никольский Константин Германович

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: _____

Москва, 2024

- **Цель работы**

Целью является приобретение практических навыков в:

- Управление потоками в ОС
- Обеспечение синхронизации между потоками

- **Задание**

Составить программу на языке Си, обрабатывающую данные в многопоточном режиме. При обработке использовать стандартные средства создания потоков операционной системы (Windows/Unix). Ограничение максимального количества потоков, работающих в один момент времени, должно быть задано ключом запуска вашей программы.

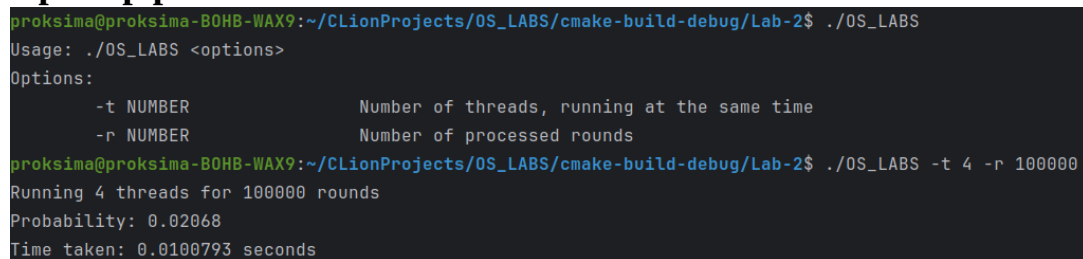
Так же необходимо уметь продемонстрировать количество потоков, используемое вашей программой с помощью стандартных средств операционной системы.

В отчете привести исследование зависимости ускорения и эффективности алгоритма от входных данных и количества потоков. Получившиеся результаты необходимо объяснить.

- **Вариант**

15 вариант - Есть колода из 52 карт, рассчитать экспериментально (метод Монте-Карло) вероятность того, что сверху лежат две одинаковых карты. Количество раундов задаётся ключом программы

- **Пример работы**



```
proksima@proksima-B0HB-WAX9:~/CLionProjects/OS_LABS/cmake-build-debug/Lab-2$ ./OS_LABS
Usage: ./OS_LABS <options>
Options:
  -t NUMBER          Number of threads, running at the same time
  -r NUMBER          Number of processed rounds
proksima@proksima-B0HB-WAX9:~/CLionProjects/OS_LABS/cmake-build-debug/Lab-2$ ./OS_LABS -t 4 -r 100000
Running 4 threads for 100000 rounds
Probability: 0.02068
Time taken: 0.0100793 seconds
```

- **Вывод**

В ходе выполнения лабораторной работы были успешно приобретены практические навыки в управлении потоками в операционной системе и обеспечении синхронизации между ними. Были изучены основные механизмы создания и управления потоками с использованием библиотек, таких как pthread, а также рассмотрены методы синхронизации, включая мьютексы, семафоры и условные переменные. Это позволило понять, как организовать параллельное выполнение задач и избежать проблем, связанных с состоянием гонки. Полученные знания имеют важное практическое значение для разработки многопоточных приложений,

обеспечивающих эффективное использование ресурсов и корректное взаимодействие между потоками.