Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №3 по курсу**

**«Операционные системы»**

**УПРАВЛЕНИЕ ПОТОКАМИ В ОС**

Студент: Черненко Илья Денисович

Группа: М8О–208Б–19

Вариант: 3

Преподаватель: Соколов Андрей Алексеевич

Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2020

**Содержание**

1. Постановка задачи
2. Общие сведения о программе
3. Общий метод и алгоритм решения
4. Основные файлы программы
5. Примеры работы
6. Вывод

**Постановка задачи**

Составить программу, обрабатывающую данные в многопоточном режиме. При обработке использовать стандартные средства создания потоков операционной системы. При создании необходимо предусмотреть ключи, которые позволяли бы задать максимальное количество потоков, используемое программой. При возможности необходимо использовать максимальное количество возможных потоков. Ограничение потоков может быть задано или ключом запуска вашей программы, или алгоритмом.

Отсортировать массив строк при помощи параллельной сортировки слиянием.

**Общие сведения о программе**

Программа компилируется из файла main.cpp. Также используется заголовочные файлы: iostream, string, pthread.h, math.h. В программе используются следующие системные вызовы:

1. **CreateThread** – создает новый поток
2. **pthread\_join** – ожидает завершения переданного потока, получает его выходное значение
3. **pthread\_mutex\_init** – инициализация mutex.
4. **pthread\_mutex\_destroy –** уничтожение mutex.
5. **pthread\_mutex\_lock –** блокировка части кода определенным потоком.
6. **pthread\_mutex\_unlock –** разблокировка части кода определенным потоком.

**Общий метод и алгоритм решения**.

Для реализации поставленной задачи необходимо:

1. Ввести размер массива строк
2. Аргументом исполняемого файла задать кол-во потоков (которое приведется к степени двойки (кроме 1).
3. Используя системный вызов fork создать дочерний процесс.
4. Исходный массив сортируется параллельной сортировкой слиянием.
5. Отсортированный массив выводится на стандартный поток вывода.

**Основные файлы программы**

**main.cpp:**

**Примеры работы**

**test1:**

**test2:**

**test3:**

**test4:**

**test5:**

**Вывод**

В результате работы были приобретены навыки работы с многопоточностью, использовал mutex для синхронизации потоков. Получены результаты измерений времени работы программы при повторяющихся тестовых данных. Причем использование стандартной библиотеки time.h не привело к результату, так как на выход метод clock давал всегда 0. Причиной этому является малый техпроцесс у процессоров Intel Core i7 1067 10th – 13hm