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 операционной системе «UNIX».**

Студент: Селивёрстов Д. С. Преподаватель: Миронов Е. С.

Группа: М8О-201Б-21

Вариант 15

Дата: Оценка: Подпись:

**Москва, 2023**

# Условие

Составить программу на языке Си, обрабатывающую данные в многопоточном режи- ме. При обработки использовать стандартные средства создания потоков операционной системы (Windows/Unix). Ограничение потоков должно быть задано ключом запуска вашей программы. Так же необходимо уметь продемонстрировать количество потоков, используемое вашей программой с помощью стандартных средств операционной систе- мы. В отчете привести исследование зависимости ускорения и эффективности алгорит- ма от входящих данных и количества потоков. Получившиеся результаты необходимо объяснить.

# Задание

Отсортировать массив целых чисел при помощи TimSort.

# Метод решения

Составленный алгоритм соответсвует принципу *completelylockless*, заключающийся в том, что каждый поток изменяет только те данные, которые не изменяют другие пото- ки.

# Код программы

## main.cpp

**#include** <b i t s / s tdc++.h>

**#include** " u t i l s . hpp"

**#include** "body . hpp"

**int** main ()

{

std : : vector <**int**> a r r ;

**for** ( **int** i = 0 ; i < 1 6 ; ++i ){

a r r . push\_back ( GetRandomInt ( 0 , 3 2 ) ) ;

}

**for** ( **int** elem : a r r ){

std : : cout << elem << "␣" ;

}

std : : cout << ’ \n ’ ;

TimSort ( arr , 2 ) ;

**for** ( **int** elem : a r r ){

std : : cout << elem << "␣" ;

}

std : : cout << ’ \n ’ ;

**return** 0 ;

}

## utils.hpp

**#ifndef** UTILS\_HPP

**#define** UTILS\_HPP

**#include** <random>

**int** GetRandomInt ( **int** min , **int** max ) ;

## #endif body.hpp

**#ifndef** BODY\_HPP

**#define** BODY\_HPP

**#include** <vector > **#include** <fstream> **#include** <pthread . h>

**#include** <iostream>

**void** TimSort ( std : : vector <**int**> &arr , **int** numOfThreads ) ;

## #endif utils.cpp

**#include** " u t i l s . hpp"

**int** GetRandomInt ( **const int** min , **const int** max)

{

std : : random\_device rd ; std : : mt19937 mt( rd ( ) ) ;

std : : uniform\_int\_distribution <**int**> d i s t ( min , max ) ;

**return** d i s t (mt ) ;

}

# Выводы

В процессе выполнения лабораторной работы были приобретены навыки практического применения создания, обработки и отслеживания состояния потоков. Для выполнения данного варианта задания использование примитивов синхронизации не требуется.