

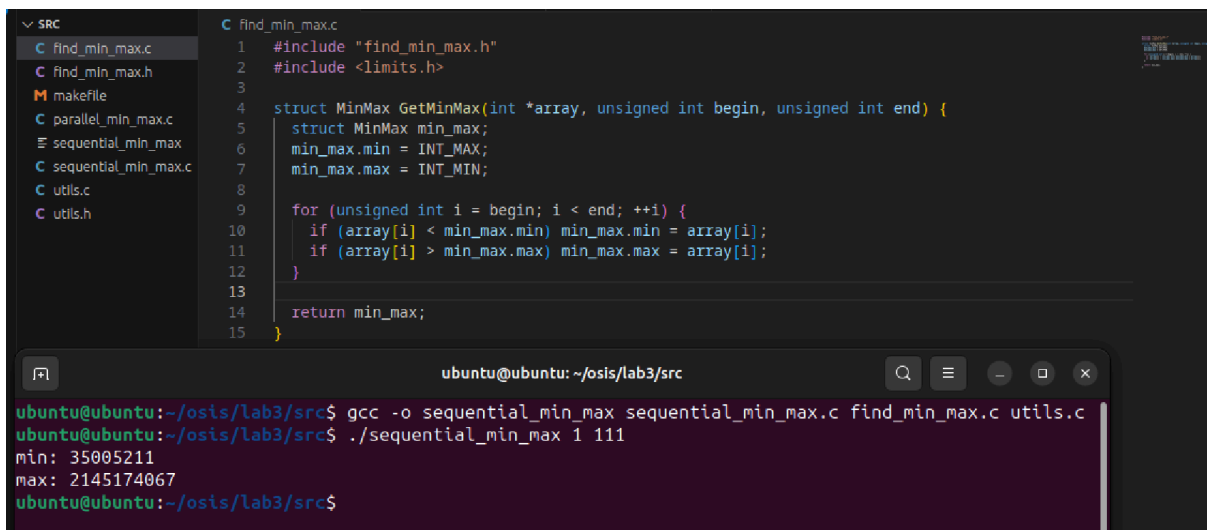
## Лабораторная работа 3

### Задание 1

#### Необходимые знания

1. Аргументы командной строки
2. Сборка с помощью gcc (clang)

Написать функцию GetMinMax в find\_max\_min.c, которая ищет минимальный и максимальный элементы массива, на заданном промежутке. Разобраться, что делает программа в sequential\_min\_max.c, скомпилировать, проверить, что написанный вами GetMinMax работает правильно.



```
▼ SRC
C find_min_max.c
C find_min_max.h
M makefile
C parallel_min_max.c
E sequential_min_max
C sequential_min_max.c
C utils.c
C utils.h

C find_min_max.c
1 #include "find_min_max.h"
2 #include <limits.h>
3
4 struct MinMax GetMinMax(int *array, unsigned int begin, unsigned int end) {
5     struct MinMax min_max;
6     min_max.min = INT_MAX;
7     min_max.max = INT_MIN;
8
9     for (unsigned int i = begin; i < end; ++i) {
10         if (array[i] < min_max.min) min_max.min = array[i];
11         if (array[i] > min_max.max) min_max.max = array[i];
12     }
13
14     return min_max;
15 }
```

```
ubuntu@ubuntu: ~/osis/lab3/src
ubuntu@ubuntu:~/osis/lab3/src$ gcc -o sequential_min_max sequential_min_max.c find_min_max.c utils.c
ubuntu@ubuntu:~/osis/lab3/src$ ./sequential_min_max 1 111
min: 35005211
max: 2145174067
ubuntu@ubuntu:~/osis/lab3/src$
```

### Задание 2 - 3

#### Необходимые знания

1. Аргументы командной строки
2. Системный вызов `fork`
3. Системный вызов `pipe`
4. Работа с файлами в Си

Завершить программу parallel\_min\_max.c, так, чтобы задача нахождения минимума и максимума в массиве решалась параллельно. Если выставлен аргумент `by_files` для синхронизации процессов использовать файлы (задание 2), в противном случае использовать `pipe` (задание 3).

```
ubuntu@ubuntu:~/osis/lab3/src$ gcc -o parallel_min_max parallel_min_max.c find_min_max.c utils.c
ubuntu@ubuntu:~/osis/lab3/src$ ./parallel_min_max --seed 123 --array_size 10000 --pnum 4
Min: 63542
Max: 2147481872
Elapsed time: 0.540000ms
ubuntu@ubuntu:~/osis/lab3/src$ ./parallel_min_max --seed 123 --array_size 10000 --pnum 4 --by_files
Min: 63542
Max: 2147481872
Elapsed time: 1.439000ms
ubuntu@ubuntu:~/osis/lab3/src$
```

## Задание 4

### Необходимые знания

1. Как работают Makefile'ы

Изучить все targets в makefile, будьте готовы объяснить, за что они отвечают. Используя `makefile`, собрать получившиеся решения. Добавьте target `all`, отвечающий за сборку всех программ.

```
ubuntu@ubuntu:~/osis/lab3/src$ make clean
rm -f sequential_min_max parallel_min_max utils.o find_min_max.o tmp_min_max_*.txt
ubuntu@ubuntu:~/osis/lab3/src$
```

## Задание 5

### Необходимые знания

1. Системный вызов `exec`

Написать программу, которая запускает в отдельном процессе ваше приложение `sequential_min_max`. Добавить его сборку в ваш makefile.

```
ubuntu@ubuntu:~/osis/lab3/src$ make clean
rm -f sequential_min_max parallel_min_max runner utils.o find_min_max.o tmp_min_max_*.txt
```