Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Лабораторная работа №2 по курсу**

**«Операционные системы»**

Группа: М8О-214Б-23

Студент: Миронов Д. А.

Преподаватель: Бахарев В. Д.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 28.10.24

Москва, 2024

**Постановка задачи**

**Вариант 20.**

**Составить программу на языке Си, обрабатывающую данные в многопоточном режиме. При обработки 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. Дан массив координат (x, y, z). Необходимо найти три точки, которые образуют треугольник максимальной площади.**

**Общий метод и алгоритм решения**

Использованные системные вызовы:

* pthread\_t threads[num\_threads]; - объявляет массив потоков.
* pthread\_create(&threads[i], NULL, find\_max\_area, &thread\_data[i]); - создаёт новый поток.
* pthread\_join(threads[i], NULL); - ожидание завершения конкретного потока.

Решение:

1. Объявляю функцию, которая будет выполняться внутри потока.
2. Считываю количество точек и количество потоков.
3. Заполняю массив точек случайным образом, используя функцию rand().
4. Инициализирую массив потоков, массив данных потоков и переменную для максимальной площади.
5. В цикле заполняю данные потока и запускаю поток.
6. В цикле жду завершения каждого потока и сравнивыаю его ответ с текущим.
7. Вывожу ответ.

**Код программы**

**server.c**

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <math.h>

**int** toInt(**const** **char**\* argv) {

**int** res = 0, i = 0;

**while** (argv[i] != '\0') {

res = res \* 10 + argv[i] - '0';

i++;

}

**return** res;

}

**typedef** **struct** {

**double** x, y, z;

} Point;

**typedef** **struct** {

Point \*points;

**int** numThreads;

**int** end;

**int** ind;

**double** max\_area;

} ThreadData;

**double** triangle\_area(**const** Point a, **const** Point b, **const** Point c) {

**double** ABx = b.x - a.x;

**double** ABy = b.y - a.y;

**double** ABz = b.z - a.z;

// Вектор AC

**double** ACx = c.x - a.x;

**double** ACy = c.y - a.y;

**double** ACz = c.z - a.z;

// Векторное произведение AB и AC

**double** cross\_x = ABy \* ACz - ABz \* ACy;

**double** cross\_y = ABz \* ACx - ABx \* ACz;

**double** cross\_z = ABx \* ACy - ABy \* ACx;

// Площадь треугольника равна половине длины вектора, полученного от векторного произведения

**return** 0.5 \* sqrt(cross\_x \* cross\_x + cross\_y \* cross\_y + cross\_z \* cross\_z);

}

**void**\* find\_max\_area(**void**\* arg) {

ThreadData \*data = (ThreadData\*)arg;

**double** max\_area = 0.0;

**for** (**int** i = data->ind; i < data->end; i += data->numThreads) {

**for** (**int** j = i + 1; j < data->end; j++) {

**for** (**int** k = j + 1; k < data->end; k++) {

**double** area = triangle\_area(data->points[i], data->points[j], data->points[k]);

//printf("i: %d | j: %d | k: %d ||| area: %lf\n", i, j, k, area);

**if** (area > max\_area) {

max\_area = area;

}

}

}

}

data->max\_area = max\_area;

**return** NULL;

}

**int** main(**int** argc, **char** \*argv[]) {

**if** (argc < 3) {

printf("Usage: <number of points> <number of threads>\n");

**return** -1;

}

**const** **int** num\_points = toInt(argv[1]);

**const** **int** num\_threads = toInt(argv[2]);

Point \*points = malloc(num\_points \* **sizeof**(Point));

**for** (**int** i = 0; i < num\_points; i++) {

points[i].x = rand() % 100;

points[i].y = rand() % 100;

points[i].z = rand() % 100;

}

pthread\_t threads[num\_threads];

ThreadData thread\_data[num\_threads];

**double** max\_area = 0.0;

// Creating threads

**for** (**int** i = 0; i < num\_threads; i++) {

thread\_data[i].points = points;

thread\_data[i].numThreads = num\_threads;

thread\_data[i].end = num\_points;

thread\_data[i].ind = i;

pthread\_create(&threads[i], NULL, find\_max\_area, &thread\_data[i]);

//printf("Thread %d\n", i + 1);

}

**for** (**int** i = 0; i < num\_threads; i++) {

pthread\_join(threads[i], NULL);

**if** (thread\_data[i].max\_area > max\_area) {

max\_area = thread\_data[i].max\_area;

}

}

printf("Max area: %f\n", max\_area);

free(points);

**return** 0;

}

**Протокол работы программы**

**Тестирование:**

$ cc -o test main.c -lm -pthread

$ ./test 256 2

Max area: 6735.965892

Num\_Points = 256

|  |  |  |  |
| --- | --- | --- | --- |
| **Число потоков** | **Время исполнения (мс)** | **Ускорение** | **Эффективность** |
| 1 | 23 | 1 | 1 |
| 2 | 13 | 1,76 | 0,88 |
| 3 | 11 | 2,09 | 0,69 |
| 4 | 10 | 2,3 | 0,57 |
| 5 | 7 | 3,28 | 0,65 |
| 6 | 6 | 3,83 | 0,63 |

Num\_Points = 1024

|  |  |  |  |
| --- | --- | --- | --- |
| **Число потоков** | **Время исполнения (мс)** | **Ускорение** | **Эффективность** |
| 1 | 1321 | 1 | 1 |
| 2 | 715 | 1,84 | 0,92 |
| 3 | 447 | 2,95 | 0,98 |
| 4 | 361 | 3,65 | 0,91 |
| 5 | 339 | 3,89 | 0,77 |
| 6 | 283 | 4,66 | 0,77 |

Num\_Points = 3000

|  |  |  |  |
| --- | --- | --- | --- |
| **Число потоков** | **Время исполнения (мс)** | **Ускорение** | **Эффективность** |
| 1 | 36126 | 1 | 1 |
| 2 | 20482 | 1,76 | 0,88 |
| 3 | 15362 | 2,35 | 0,78 |
| 4 | 9294 | 3,88 | 0,97 |
| 5 | 8413 | 4,29 | 0,85 |
| 6 | 6905 | 5,23 | 0,87 |

**Strace:**

$ strace ./test 1024 24

execve("./test", ["./test", "1024", "24"], 0x7fff65ffebf0 /\* 28 vars \*/) = 0

brk(NULL) = 0x556ab99b8000

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f0417aa9000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=20335, ...}) = 0

mmap(NULL, 20335, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f0417aa4000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libm.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=952616, ...}) = 0

mmap(NULL, 950296, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f04179bb000

mmap(0x7f04179cb000, 520192, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x10000) = 0x7f04179cb000

mmap(0x7f0417a4a000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8f000) = 0x7f0417a4a000

mmap(0x7f0417aa2000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe7000) = 0x7f0417aa2000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2125328, ...}) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

mmap(NULL, 2170256, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f04177a9000

mmap(0x7f04177d1000, 1605632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f04177d1000

mmap(0x7f0417959000, 323584, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1b0000) = 0x7f0417959000

mmap(0x7f04179a8000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1fe000) = 0x7f04179a8000

mmap(0x7f04179ae000, 52624, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f04179ae000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f04177a6000

arch\_prctl(ARCH\_SET\_FS, 0x7f04177a6740) = 0

set\_tid\_address(0x7f04177a6a10) = 1580

set\_robust\_list(0x7f04177a6a20, 24) = 0

rseq(0x7f04177a7060, 0x20, 0, 0x53053053) = 0

mprotect(0x7f04179a8000, 16384, PROT\_READ) = 0

mprotect(0x7f0417aa2000, 4096, PROT\_READ) = 0

mprotect(0x556aa83e7000, 4096, PROT\_READ) = 0

mprotect(0x7f0417ae1000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f0417aa4000, 20335) = 0

getrandom("\x9d\x77\x11\xaf\x53\xd8\x6e\x83", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x556ab99b8000

brk(0x556ab99d9000) = 0x556ab99d9000

rt\_sigaction(SIGRT\_1, {sa\_handler=0x7f0417842520, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_ONSTACK|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7f04177ee320}, NULL, 8) = 0

rt\_sigprocmask(SIG\_UNBLOCK, [RTMIN RT\_1], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0416fa5000

mprotect(0x7f0416fa6000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f04177a5990, parent\_tid=0x7f04177a5990, exit\_signal=0, stack=0x7f0416fa5000, stack\_size=0x7fff80, tls=0x7f04177a56c0} => {parent\_tid=[1581]}, 88) = 1581

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f04167a4000

mprotect(0x7f04167a5000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0416fa4990, parent\_tid=0x7f0416fa4990, exit\_signal=0, stack=0x7f04167a4000, stack\_size=0x7fff80, tls=0x7f0416fa46c0} => {parent\_tid=[1582]}, 88) = 1582

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0415fa3000

mprotect(0x7f0415fa4000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f04167a3990, parent\_tid=0x7f04167a3990, exit\_signal=0, stack=0x7f0415fa3000, stack\_size=0x7fff80, tls=0x7f04167a36c0} => {parent\_tid=[1583]}, 88) = 1583

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f04157a2000

mprotect(0x7f04157a3000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0415fa2990, parent\_tid=0x7f0415fa2990, exit\_signal=0, stack=0x7f04157a2000, stack\_size=0x7fff80, tls=0x7f0415fa26c0} => {parent\_tid=[1584]}, 88) = 1584

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0414fa1000

mprotect(0x7f0414fa2000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f04157a1990, parent\_tid=0x7f04157a1990, exit\_signal=0, stack=0x7f0414fa1000, stack\_size=0x7fff80, tls=0x7f04157a16c0} => {parent\_tid=[1585]}, 88) = 1585

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f04147a0000

mprotect(0x7f04147a1000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0414fa0990, parent\_tid=0x7f0414fa0990, exit\_signal=0, stack=0x7f04147a0000, stack\_size=0x7fff80, tls=0x7f0414fa06c0} => {parent\_tid=[1586]}, 88) = 1586

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0413f9f000

mprotect(0x7f0413fa0000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f041479f990, parent\_tid=0x7f041479f990, exit\_signal=0, stack=0x7f0413f9f000, stack\_size=0x7fff80, tls=0x7f041479f6c0} => {parent\_tid=[1587]}, 88) = 1587

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f041379e000

mprotect(0x7f041379f000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0413f9e990, parent\_tid=0x7f0413f9e990, exit\_signal=0, stack=0x7f041379e000, stack\_size=0x7fff80, tls=0x7f0413f9e6c0} => {parent\_tid=[1588]}, 88) = 1588

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0412f9d000

mprotect(0x7f0412f9e000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f041379d990, parent\_tid=0x7f041379d990, exit\_signal=0, stack=0x7f0412f9d000, stack\_size=0x7fff80, tls=0x7f041379d6c0} => {parent\_tid=[1589]}, 88) = 1589

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f041279c000

mprotect(0x7f041279d000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0412f9c990, parent\_tid=0x7f0412f9c990, exit\_signal=0, stack=0x7f041279c000, stack\_size=0x7fff80, tls=0x7f0412f9c6c0} => {parent\_tid=[1590]}, 88) = 1590

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0411f9b000

mprotect(0x7f0411f9c000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f041279b990, parent\_tid=0x7f041279b990, exit\_signal=0, stack=0x7f0411f9b000, stack\_size=0x7fff80, tls=0x7f041279b6c0} => {parent\_tid=[1591]}, 88) = 1591

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f041179a000

mprotect(0x7f041179b000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0411f9a990, parent\_tid=0x7f0411f9a990, exit\_signal=0, stack=0x7f041179a000, stack\_size=0x7fff80, tls=0x7f0411f9a6c0} => {parent\_tid=[1592]}, 88) = 1592

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0410f99000

mprotect(0x7f0410f9a000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0411799990, parent\_tid=0x7f0411799990, exit\_signal=0, stack=0x7f0410f99000, stack\_size=0x7fff80, tls=0x7f04117996c0} => {parent\_tid=[1593]}, 88) = 1593

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f0410798000

mprotect(0x7f0410799000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0410f98990, parent\_tid=0x7f0410f98990, exit\_signal=0, stack=0x7f0410798000, stack\_size=0x7fff80, tls=0x7f0410f986c0} => {parent\_tid=[1594]}, 88) = 1594

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040ff97000

mprotect(0x7f040ff98000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f0410797990, parent\_tid=0x7f0410797990, exit\_signal=0, stack=0x7f040ff97000, stack\_size=0x7fff80, tls=0x7f04107976c0} => {parent\_tid=[1595]}, 88) = 1595

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040f796000

mprotect(0x7f040f797000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040ff96990, parent\_tid=0x7f040ff96990, exit\_signal=0, stack=0x7f040f796000, stack\_size=0x7fff80, tls=0x7f040ff966c0} => {parent\_tid=[1596]}, 88) = 1596

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040ef95000

mprotect(0x7f040ef96000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040f795990, parent\_tid=0x7f040f795990, exit\_signal=0, stack=0x7f040ef95000, stack\_size=0x7fff80, tls=0x7f040f7956c0} => {parent\_tid=[1597]}, 88) = 1597

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040e794000

mprotect(0x7f040e795000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040ef94990, parent\_tid=0x7f040ef94990, exit\_signal=0, stack=0x7f040e794000, stack\_size=0x7fff80, tls=0x7f040ef946c0} => {parent\_tid=[1598]}, 88) = 1598

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040df93000

mprotect(0x7f040df94000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040e793990, parent\_tid=0x7f040e793990, exit\_signal=0, stack=0x7f040df93000, stack\_size=0x7fff80, tls=0x7f040e7936c0} => {parent\_tid=[1599]}, 88) = 1599

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040d792000

mprotect(0x7f040d793000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040df92990, parent\_tid=0x7f040df92990, exit\_signal=0, stack=0x7f040d792000, stack\_size=0x7fff80, tls=0x7f040df926c0} => {parent\_tid=[1600]}, 88) = 1600

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040cf91000

mprotect(0x7f040cf92000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040d791990, parent\_tid=0x7f040d791990, exit\_signal=0, stack=0x7f040cf91000, stack\_size=0x7fff80, tls=0x7f040d7916c0} => {parent\_tid=[1601]}, 88) = 1601

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040c790000

mprotect(0x7f040c791000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040cf90990, parent\_tid=0x7f040cf90990, exit\_signal=0, stack=0x7f040c790000, stack\_size=0x7fff80, tls=0x7f040cf906c0} => {parent\_tid=[1602]}, 88) = 1602

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040bf8f000

mprotect(0x7f040bf90000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040c78f990, parent\_tid=0x7f040c78f990, exit\_signal=0, stack=0x7f040bf8f000, stack\_size=0x7fff80, tls=0x7f040c78f6c0} => {parent\_tid=[1603]}, 88) = 1603

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7f040b78e000

mprotect(0x7f040b78f000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7f040bf8e990, parent\_tid=0x7f040bf8e990, exit\_signal=0, stack=0x7f040b78e000, stack\_size=0x7fff80, tls=0x7f040bf8e6c0} => {parent\_tid=[1604]}, 88) = 1604

rt\_sigprocmask(SIG\_SETMASK, [], NULL, 8) = 0

futex(0x7f04177a5990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 1581, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0

futex(0x7f0415fa2990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 1584, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0

munmap(0x7f0416fa5000, 8392704) = 0

futex(0x7f0414fa0990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 1586, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0

munmap(0x7f04167a4000, 8392704) = 0

futex(0x7f041479f990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 1587, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0

munmap(0x7f0415fa3000, 8392704) = 0

munmap(0x7f04157a2000, 8392704) = 0

munmap(0x7f0414fa1000, 8392704) = 0

munmap(0x7f04147a0000, 8392704) = 0

futex(0x7f041279b990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 1591, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0

munmap(0x7f0413f9f000, 8392704) = 0

munmap(0x7f041379e000, 8392704) = 0

munmap(0x7f0412f9d000, 8392704) = 0

munmap(0x7f041279c000, 8392704) = 0

munmap(0x7f0411f9b000, 8392704) = 0

munmap(0x7f041179a000, 8392704) = 0

munmap(0x7f0410f99000, 8392704) = 0

munmap(0x7f0410798000, 8392704) = 0

munmap(0x7f040ff97000, 8392704) = 0

munmap(0x7f040f796000, 8392704) = 0

munmap(0x7f040ef95000, 8392704) = 0

munmap(0x7f040e794000, 8392704) = 0

munmap(0x7f040df93000, 8392704) = 0

futex(0x7f040bf8e990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 1604, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0

munmap(0x7f040d792000, 8392704) = 0

fstat(1, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}) = 0

write(1, "Max area: 7211.060914\n", 22Max area: 7211.060914

) = 22

exit\_group(0) = ?

+++ exited with 0 +++

**Вывод**

В результате выполнения лабораторной работы удалось познакомиться с многопоточным программированием. Программа успешно реализует задачу обработки данных в многопоточном режиме с использованием стандартных средств операционной системы. Были изучены базовые системные вызовы для создания и управления потоками.