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: М80-206Б-20

Студент: Кочев Д.О.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 01.12.2023

Москва, 2023

Постановка задачи

Вариант 9.

Рассчитать детерминант матрицы (используя определение детерминанта)

Общий метод и алгоритм решения

Использованные системные вызовы:

- int pthread_create(pthread_t *restrict thread, const pthread_attr_t *restrict attr, void *(*start_routine)(void *), void *restrict arg) - создает поток.
- pthread_join(pthread_t thread, void **retval) - блокирует вызывающий поток, пока указанный поток не завершится.

Сначала программа считывает порядок матрицы. Так как детерминант мы находим по определению, каждый процесс берет на себя вычисление некоторого количества алгебраических дополнений элементов первой строки матрицы. Следовательно, максимальное количество используемых потоков в программе всегда равно порядку вводимой матрицы. Если количество потоков меньше, то программа старается распределить количество алгебраических дополнений поровну между всеми потоками. Далее в каждом процессе находятся алгебраические дополнения элементов матрицы с помощью рекурсивной функции. После завершения работы всех процессов их результаты складываются и выводится ответ.

Код программы

lab2.c

```
#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <sys/time.h>

typedef struct {

   int parami;

   int param_numbers;

   int number_of_thread;

} ThreadParams;
```

```
int** arr;
int n;
int* determinant;

void freeMemory() {
    for (int i = 0; i < n; ++i) {
        free(arr[i]);
    }
    free(arr);
}

void findMinor(int minor[n][n], int row, int col, int size) {
    int minor_plus[size][size];
    for (int i = 0; i < size; i++)
        for (int j = 0; j < size; j++)
            minor_plus[i][j] = minor[i][j];

    int minorRow = 0, minorCol = 0;

    for (int i = 0; i < size; ++i) {
        if (i != row) {
            for (int j = 0; j < size; ++j) {
                if (j != col) {
                    minor[minorRow][minorCol] = minor_plus[i][j];
                    minorCol++;
                }
            }
```

```
            minorCol = 0;
            minorRow++;
        }
    }
}

// Функция для нахождения определителя матрицы
int findDeterminant(int matrix[n][n], int size) {
    if (size == 1) {
        return matrix[0][0];
    } else if (size == 2) {
        return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0];
    } else {
        int det = 0;
        int sign = 1;

        for (int i = 0; i < size; i++) {
            int minor[n][n]; // ОБЯЗАТЕЛЬНО РАЗМЕРНОСТЬ ТАКАЯ ЖЕ, КАК В ОБЪЯВЛЕНИИ ФУНКЦИИ

             for (int u = 0; u < size; u++){
                for (int j = 0; j < size; j++)
                    minor[u][j] = matrix[u][j];
            }

            // for (int u = 0; u < size; u++){
            //     for (int j = 0; j < size; j++)
```

```
            //         printf("%d ", minor[u][j]);
            //     printf("\n");
            // }

            findMinor(minor, 0, i, size);

            // for (int u = 0; u < size - 1; u ++){
            //     for (int j = 0; j < size - 1; j++)
            //         printf("%d ", minor[u][j]);
            //     printf("\n");
            // }
            // printf("\n");

            det += sign * matrix[0][i] * findDeterminant(minor, size - 1);
            sign = -sign;
        }

        return det;
    }
}

int help_algComplement(int row, int column){
    int arr_copy[n][n];
    for (int i = 0; i < n; i ++)
        for (int j = 0; j < n; j++)
            arr_copy[i][j] = arr[i][j];
    findMinor(arr_copy, row, column, n);
```

```
    return arr[row][column] * findDeterminant(arr_copy, n - 1);
}



void thread_create(pthread_t* thread, const pthread_attr_t* attr, void *(*start)(void
*), void* arg) {
    if (pthread_create(thread, attr, start, arg) != 0) {
        perror("create thread\n");
        exit(-1);
    }
}


void* threadDistributor(void* arg){
    ThreadParams* paramsss = (ThreadParams*)arg;
    int i = paramsss->parami;
    int numbers = paramsss->param_numbers;
    // printf("i in Distributor: %d \n", paramsss->parami);
    // printf("numbers: %d \n", paramsss->param_numbers);
    printf("номер потока: %d \n", paramsss->number_of_thread);
    int s = 0;
    for (int j = i; j < i + numbers; j++){
        // printf("determinant before: %d \n", determinant[j]);
        determinant[j] = help_algComplement(0, j);
        // printf("determinant after: %d \n", determinant[j]);
        // s += algComplement(0, j);
    }
```

```
    pthread_exit((void*)&s);
    // pthread_exit(NULL);
}


int main(int argc, char* argv[]){
    if (argc != 2){
        printf("Incorrect input of arguments\n");
        return 1;
    }
    int max_threads = atoi(argv[1]);
    if (max_threads <= 0){
        printf("Number of threads must be more then 0\n");
        return 1;
    }
    printf("Please enter the matrix dimension: ");
    scanf("%d", &n);

    arr = (int**)malloc(n * sizeof(int*));
    determinant = (int*)malloc(n * sizeof(int));

    for (int i = 0; i < n; i++){
        arr[i] = (int*)malloc(n * sizeof(int));
        for (int j = 0; j < n; j++)
            scanf("%d", &arr[i][j]);
    }
```

```
// for (int i = 0; i < n; i++){
//     for (int j = 0; j < n; j++)
//         printf("%d ", arr[i][j]);
// printf("\n");
// }



double numbers_in_threads_doub = (double)n / max_threads;
int numbers_in_threads = n / max_threads;
if (numbers_in_threads_doub != (double)numbers_in_threads)
    numbers_in_threads += 1;
// printf("numbers_in_threads: %d \n", numbers_in_threads);



pthread_t* threads = (pthread_t*)malloc(max_threads * sizeof(pthread_t));
// int* thread_args = (int*)malloc(max_threads * sizeof(int));

ThreadParams params[n];
int k = 0;

struct timeval start_time, end_time;
gettimeofday(&start_time, NULL); // начальное время

for (int i = 0; i < n; i += numbers_in_threads){
    params[k].param_numbers = numbers_in_threads;
    params[k].parami = i;
```

```
        params[k].number_of_thread = k + 1;
        // printf("i in main: %d \n", i);
        thread_create(&threads[k], NULL, threadDistributor, (void*)&params[k]);
        k++;
    }

    for (int i = 0; i < k; i += 1){
        pthread_join(threads[i], NULL);
    }

    int det_final = 0;
    int check = 1;
    for (int i = 0; i< n; i++){
        det_final += determinant[i] * check;
        // printf("%d ", determinant[i]);
        check *= -1;
    }

    gettimeofday(&end_time, NULL); // конечное время
    long long start_ms = start_time.tv_sec * 1000LL + start_time.tv_usec / 1000; // преобразование из микросекунд в миллисекунды
    long long end_ms = end_time.tv_sec * 1000LL + end_time.tv_usec / 1000; // преобразование из микросекунд в миллисекунды
    double search_time = (end_ms - start_ms) / 1000.0; // искомо

    printf("\nОтвет: %d \n", det_final);

    printf("\nвремя выполения: %.4f", search_time);
```

```
    // printf("\nstart time: %f", start_ms / 1000.0);
    // printf("\nend_time: %f\n ", end_ms / 1000.0);

    freeMemory();
    free(determinant);
    free(threads);


}
```

Протокол работы программы

```
$ ./a.out 3
Please enter the matrix dimension: 4
1 2 3 1
4 5 6 2
7 8 10 6
8 3 9 7
номер потока: 1
номер потока: 2
Ответ: 76
$ ./a.out 2
Please enter the matrix dimension: 4
1 2 3 1
4 5 6 2
7 8 10 6
8 3 9 7
номер потока: 1
номер потока: 2
```

Ответ: 76

$ ./a.out 4

Please enter the matrix dimension: 4

19 2 3 1

4 5 6 2

7 8 10 6

8 3 9 18

номер потока: 1

номер потока: 2

номер потока: 3

номер потока: 4

Ответ: -713

$ ./a.out 2

Please enter the matrix dimension: 5

19 2 3 1 1

4 5 6 2 2

7 8 10 6 3

8 3 9 18 4

9 8 7 6 5

номер потока: 1

номер потока: 2

Ответ: -4370

$ ./a.out 7

Please enter the matrix dimension: 5

19 2 3 1 1

4 5 6 2 2

7 8 10 6 3

8 3 9 18 4

9 8 7 6 5

номер потока: 1

номер потока: 3

номер потока: 2

номер потока: 4

номер потока: 5

Ответ: -4370

$ ./a.out 7

Please enter the matrix dimension: 7

19 2 3 1 1 2 9

4 5 6 2 2 10 8

7 8 10 6 3 3 1

8 3 9 18 4 2 2

9 8 7 6 5 -1 3

0 1 11 23 1 2 4

1 24 2 7 9 7 64

номер потока: 1

номер потока: 2

номер потока: 7

номер потока: 3

номер потока: 4

номер потока: 5

номер потока: 6

Ответ: 5179890

$ ./a.out 8

Please enter the matrix dimension: 7

19 2 3 1 1 2 9

4 5 6 2 2 10 8

7 8 10 6 3 3 1

8 3 9 18 4 2 2

9 8 7 6 5 -1 3

0 1 11 23 1 2 4

1 24 2 7 9 7 64

номер потока: 1

номер потока: 2

номер потока: 3

номер потока: 5

номер потока: 4

номер потока: 7

номер потока: 6

Ответ: 5179890

$ ./a.out 4

Please enter the matrix dimension: 7

19 2 3 1 1 2 9

4 5 6 2 2 10 8

7 8 10 6 3 3 1

8 3 9 18 4 2 2

9 8 7 6 5 -1 3

0 1 11 23 1 2 4

1 24 2 7 9 7 64

номер потока: 1

номер потока: 3

номер потока: 2

номер потока: 4

Ответ: 5179890

Таблица зависимости времени выполнения от исходных данных и количества потоков:

Порядок матрицы	**Число потоков**	**Время исполнения (с)**	**Ускорение**	**Эффективность**
11	1	5.0220	1	1
11	2	3.1640	1.59	0.795
11	4	1.6830	2.98	0.745
11	6	1.2810	3.92	0.653
11	11	1.1290	4.45	0.404
9	1	0.0620	1	1
9	9	0.0180	3.44	0.382
7	1	0.0060	1	1
7	7	0.0010	6	0.85

Strace:

```
$ strace -f ./a.out 2
execve("./a.out", ["./a.out", "2"], 0x7ffd268697c0 /* 36 vars */) = 0
brk(NULL)                               = 0x559786f32000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffd8c3b0c00) = -1 EINVAL (Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x7fb6e42c4000
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
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=18151, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 18151, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fb6e42bf000
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\0P\237\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
pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48
```

```
pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\244;\374\204(\337f#\315I\214\234\f\256\271\32"..., 68, 896)
= 68
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2216304, ...}, AT_EMPTY_PATH) = 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, 2260560, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7fb6e4097000
mmap(0x7fb6e40bf000, 1658880, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7fb6e40bf000
mmap(0x7fb6e4254000, 360448, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7fb6e4254000
mmap(0x7fb6e42ac000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x214000) = 0x7fb6e42ac000
mmap(0x7fb6e42b2000, 52816, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fb6e42b2000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS,
-1, 0) = 0x7fb6e4094000
arch_prctl(ARCH_SET_FS, 0x7fb6e4094740) = 0
set_tid_address(0x7fb6e4094a10)         = 3863
set_robust_list(0x7fb6e4094a20, 24)     = 0
rseq(0x7fb6e40950e0, 0x20, 0, 0x53053053) = 0
mprotect(0x7fb6e42ac000, 16384, PROT_READ) = 0
mprotect(0x559785386000, 4096, PROT_READ) = 0
mprotect(0x7fb6e42fe000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) =
0
munmap(0x7fb6e42bf000, 18151)           = 0
newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x4), ...},
AT_EMPTY_PATH) = 0
getrandom("\xeb\xfe\x18\x76\xab\x28\xa4\x68", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x559786f32000
brk(0x559786f53000)                     = 0x559786f53000
newfstatat(0, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x4), ...},
AT_EMPTY_PATH) = 0
write(1, "Please enter the matrix dimensio"..., 35Please enter the matrix dimension: ) = 35
read(0, 4
"4\n", 1024)                    = 2
read(0, 1 2 3 4
"1 2 3 4\n", 1024)              = 8
read(0, 67 34 0 9
"67 34 0 9\n", 1024)            = 10
read(0, 12 3 4 1
"12 3 4 1\n", 1024)             = 9
read(0, 2 45 60
"2 45 60\n", 1024)              = 8
read(0, 89
"89\n", 1024)                   = 3
```

```
rt_sigaction(SIGRT_1, {sa_handler=0x7fb6e4128870, sa_mask=[],
sa_flags=SA_RESTORER|SA_ONSTACK|SA_RESTART|SA_SIGINFO,
sa_restorer=0x7fb6e40d9520}, 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) = 0x7fb6e3893000
mprotect(0x7fb6e3894000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREA
D|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEA
RTID, child_tid=0x7fb6e4093910, parent_tid=0x7fb6e4093910, exit_signal=0,
stack=0x7fb6e3893000, stack_size=0x7fff00, tls=0x7fb6e4093640}strace: Process 3992
attached
 => {parent_tid=[3992]}, 88) = 3992
[pid  3863] rt_sigprocmask(SIG_SETMASK, [],  <unfinished ...>
[pid  3992] rseq(0x7fb6e4093fe0, 0x20, 0, 0x53053053 <unfinished ...>
[pid  3863] <... rt_sigprocmask resumed>NULL, 8) = 0
[pid  3992] <... rseq resumed>)         = 0
[pid  3863] mmap(NULL, 8392704, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>
[pid  3992] set_robust_list(0x7fb6e4093920, 24 <unfinished ...>
[pid  3863] <... mmap resumed>)         = 0x7fb6e3092000
[pid  3992] <... set_robust_list resumed>) = 0
[pid  3863] mprotect(0x7fb6e3093000, 8388608, PROT_READ|PROT_WRITE <unfinished
...>
[pid  3992] rt_sigprocmask(SIG_SETMASK, [],  <unfinished ...>
[pid  3863] <... mprotect resumed>)     = 0
[pid  3992] <... rt_sigprocmask resumed>NULL, 8) = 0
[pid  3863] rt_sigprocmask(SIG_BLOCK, ~[],  <unfinished ...>
[pid  3992] write(1, "\320\275\320\276\320\274\320\265\321\200
\320\277\320\276\321\202\320\276\320\272\320\260: 1 \n", 28номер потока: 1
 <unfinished ...>
[pid  3863] <... rt_sigprocmask resumed>[], 8) = 0
[pid  3992] <... write resumed>)        = 28
[pid  3863]
clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREA
D|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEA
RTID, child_tid=0x7fb6e3892910, parent_tid=0x7fb6e3892910, exit_signal=0,
stack=0x7fb6e3092000, stack_size=0x7fff00, tls=0x7fb6e3892640} <unfinished ...>
[pid  3992] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXECstrace:
Process 3993 attached
) = 3
[pid  3863] <... clone3 resumed> => {parent_tid=[3993]}, 88) = 3993
[pid  3993] rseq(0x7fb6e3892fe0, 0x20, 0, 0x53053053 <unfinished ...>
[pid  3863] rt_sigprocmask(SIG_SETMASK, [],  <unfinished ...>
[pid  3992] newfstatat(3, "",  <unfinished ...>
[pid  3863] <... rt_sigprocmask resumed>NULL, 8) = 0
[pid  3993] <... rseq resumed>)         = 0
```

```
[pid  3863] futex(0x7fb6e4093910, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME,
3992, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
[pid  3992] <... newfstatat resumed>{st_mode=S_IFREG|0644, st_size=18151, ...},
AT_EMPTY_PATH) = 0
[pid  3993] set_robust_list(0x7fb6e3892920, 24 <unfinished ...>
[pid  3992] mmap(NULL, 18151, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>
[pid  3993] <... set_robust_list resumed>) = 0
[pid  3992] <... mmap resumed>)         = 0x7fb6e42bf000
[pid  3993] rt_sigprocmask(SIG_SETMASK, [],  <unfinished ...>
[pid  3992] close(3 <unfinished ...>
[pid  3993] <... rt_sigprocmask resumed>NULL, 8) = 0
[pid  3992] <... close resumed>)        = 0
[pid  3993] write(1, "\320\275\320\276\320\274\320\265\321\200
\320\277\320\276\321\202\320\276\320\272\320\260: 2 \n", 28 <unfinished ...>
[pid  3992] mmap(NULL, 134217728, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_NORESERVE, -1, 0номер потока: 2
 <unfinished ...>
[pid  3993] <... write resumed>)        = 28
[pid  3992] <... mmap resumed>)         = 0x7fb6db092000
[pid  3993] futex(0x7fb6e4300a48, FUTEX_WAIT_PRIVATE, 2, NULL <unfinished ...>
[pid  3992] munmap(0x7fb6db092000, 16179200) = 0
[pid  3992] munmap(0x7fb6e0000000, 50929664) = 0
[pid  3992] mprotect(0x7fb6dc000000, 135168, PROT_READ|PROT_WRITE) = 0
[pid  3992] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1",
O_RDONLY|O_CLOEXEC) = 3
[pid  3992] read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0"..., 832) =
832
[pid  3992] newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=125488, ...},
AT_EMPTY_PATH) = 0
[pid  3992] mmap(NULL, 127720, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7fb6e3072000
[pid  3992] mmap(0x7fb6e3075000, 94208, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000) = 0x7fb6e3075000
[pid  3992] mmap(0x7fb6e308c000, 16384, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1a000) = 0x7fb6e308c000
[pid  3992] mmap(0x7fb6e3090000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1d000) = 0x7fb6e3090000
[pid  3992] close(3)                    = 0
[pid  3992] mprotect(0x7fb6e3090000, 4096, PROT_READ) = 0
[pid  3992] munmap(0x7fb6e42bf000, 18151) = 0
[pid  3992] futex(0x7fb6e4300a48, FUTEX_WAKE_PRIVATE, 1 <unfinished ...>
[pid  3993] <... futex resumed>)        = 0
[pid  3992] <... futex resumed>)        = 1
[pid  3993] futex(0x7fb6e4300a48, FUTEX_WAKE_PRIVATE, 1 <unfinished ...>
[pid  3992] futex(0x7fb6e3091210, FUTEX_WAKE_PRIVATE, 2147483647 <unfinished ...>
[pid  3993] <... futex resumed>)        = 0
[pid  3992] <... futex resumed>)        = 0
[pid  3993] rt_sigprocmask(SIG_BLOCK, ~[RT_1],  <unfinished ...>
```

```
[pid  3992] rt_sigprocmask(SIG_BLOCK, ~[RT_1],  <unfinished ...>
[pid  3993] <... rt_sigprocmask resumed>NULL, 8) = 0
[pid  3992] <... rt_sigprocmask resumed>NULL, 8) = 0
[pid  3993] madvise(0x7fb6e3092000, 8368128, MADV_DONTNEED <unfinished ...>
[pid  3992] madvise(0x7fb6e3893000, 8368128, MADV_DONTNEED <unfinished ...>
[pid  3993] <... madvise resumed>)      = 0
[pid  3992] <... madvise resumed>)      = 0
[pid  3993] exit(0 <unfinished ...>
[pid  3992] exit(0 <unfinished ...>
[pid  3993] <... exit resumed>)         = ?
[pid  3992] <... exit resumed>)         = ?
[pid  3993] +++ exited with 0 +++
[pid  3863] <... futex resumed>)        = 0
[pid  3992] +++ exited with 0 +++
write(1, "\320\236\321\202\320\262\320\265\321\202: 4724 \n", 18Ответ: 4724
) = 18
lseek(0, -1, SEEK_CUR)                  = -1 ESPIPE (Illegal seek)
exit_group(0)                           = ?
+++ exited with 0 +++
```

Вывод

У меня получилось реализовать программу с использованием многопоточности и доказать, что таким я сокращаю время работы своего кода. Стало очевидно, что в задачах с большими данными многопоточность - незаменимый инструмент, который может сократить время выполнения в несколько раз. Было очень интересно продумывать работу каждого потока и организовывать логику их выполнения, следя за тем, чтобы не происходили “Data Race”. Уверен, эти знания пригодятся мне в будущем.