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 и синхронизация

Студент: Семенов Илья Михайлович
Группа: М80 – 206Б-18
Вариант: 21
Преподаватель: Соколов Андрей Алексеевич
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2019

Постановка задачи

Составить и отладить программу на языке Си, обрабатывающую данные в многопоточном режиме. Использовать стандартные средства создания потоков операционной системы Unix. Предусмотреть возможность ограничить максимальное количество потоков, используемых в программе.

Вариант задания: 21. Произвести поиск кратчайшего пути в графе поиском в ширину. Граф задается матрицей смежности, где элементы этой матрицы указывают расстояние между вершинами.

Общие сведения о программе

Программа состоит из файла **main.c, adjacency_list.c, adjacency_list.h**, **queue.c, queue.h.**

В **main.c** реализован многопоточный алгоритм поиска в ширину в графе.

В **adjacency_list.c** и **adjacency_list.h** реализована структура, называемая списком смежности, необходимая в этой программе для более удобного представления графа.

В **queue.c** и **queue.h** реализована очередь, без которой невозможно реализовать любой обход в ширину.

В программе используются заголовочные файлы stdio.h, pthread.h, stdbool.h, ctype.h.

Используются следующие системные вызовы

1. **pthread_create** — создает новый поток выполнения в программе.
2. **pthread_mutex_init —** инициализирует мьютекс.
3. **pthread_mutex_lock —** блокирует мьютекс.
4. **pthread_mutex_unlock —** разблокирует мьютекс.
5. **pthread_join —** дожидается завершения переданного потока, после чего получает его выходное значение и позволяет программе продолжить работу.

Общий метод и алгоритм решения

- Получить из входного потока матрицу смежности графа, номер вершины для начала поиска в ширину и максимальное возможное количество потоков, обработать возможные ошибки ввода.
- Преобразовать матрицу смежности в список смежности, представленный в файлах **adjacency_list**.

- Создать и инициализировать два мьютекса, для синхронизации доступа, а также создать структуру с параметрами, используемыми в функции, вызываемой в отдельных потоках
- Запустить функцию обхода в ширину многопоточно. Функция принимает в параметрах номер вершины, из которой происходит обход. Сначала подсчитывается расстояние до вершин, смежных с данной, если в некоторых вершинах удалось его уменьшить, они помещаются в вектор вершин для пересчета.
- Проверить, сколько еще можно создать потоков, запустить функцию в отдельном потоке для максимально возможного количества вершин из ранее полученного вектора. Для оставшихся вершин запустить стандартный поиск в ширину с использованием очереди. При поиске в ширину стартовые вершины добавляются в очередь, далее, пока очередь не пуста, из нее извлекается первый элемент, пересчитываются расстояние до соседних с этим элементом вершин. Вершины, для которых оно изменилось снова добавляются в очередь. К массиву с результатом и переменной, в которой хранится количество потоков, которые можно создать, осуществляется синхронизированный доступ.
- После завершения обхода в ширину, необходимо подождать завершения дочерних потоков, и освободить использованную ранее память в куче.
- После завершения работы самого первого потока, вывести результат.

Код программы

main.c:

```
int main(int argc, char** argv) {
    if (argc < 3) {
        printf("Not enough arguments\n");
        exit(1);
    }
    FILE* file = fopen(argv[1], "r");
    if (!file) {
        printf("Wrong filename\n");
        exit(1);
    }
    int *max_thread_count = malloc(sizeof(int));
    *max_thread_count = parse_count(argv[2]);
    if (*max_thread_count <= 0) {
        printf("Incorrect thread count\n");
        exit(1);
    }
    int n;
    fscanf(file, "%d", &n);
    int matrix[n][n];
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
```

```
            fscanf(file, "%d", &matrix[i][j]);
        }
    }
    fclose(file);
    adjacency_list *list = malloc(sizeof(adjacency_list));
    a_init(list);
    a_resize(list, n);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (matrix[i][j] != 0) {
                v_push(&list->vecs[i], (pair) {j, matrix[i][j]});
            }
        }
    }
    size_t vertex_number;
    int *result = malloc(sizeof(int) * n);
    for (int i = 0; i < n; ++i) {
        result[i] = -1;
    }
    pthread_mutex_t *mutex = malloc(sizeof(pthread_mutex_t) * n);
    pthread_mutex_init(mutex, NULL);
    pthread_mutex_t* thread_count_mutex = malloc(sizeof(pthread_mutex_t));
    pthread_mutex_init(thread_count_mutex, NULL);
    printf("Enter vertex to start search\n");
    scanf("%ld", &vertex_number);
    while (vertex_number > n || vertex_number == 0) {
        printf("Vertex number must be < %d\n and > 0\n", n);
        printf("Enter vertex to start search\n");
        scanf("%ld", &vertex_number);
    }
    vertex_number--;
    result[vertex_number] = 0;
    (*max_thread_count)--;
    thread_params parameters = (thread_params) {.max_thread_count =
max_thread_count,
            .result = result,
            .current_vertex = vertex_number,
            .list = list,
            .mutex = mutex,
            .thread_count_mutex = thread_count_mutex};
    pthread_t* first_thread = malloc(sizeof(pthread_t));
    pthread_create(first_thread, NULL, bfs, &parameters);
    pthread_join(*first_thread, NULL);
    for (int i = 0; i < n; ++i) {
        printf("%d:%d ", i + 1, result[i]);
    }
    printf("\n");
    return 0;
}
```

adjacency_list.h

```
#ifndef OS_LAB_3_ADJACENCY_LIST_H
#define OS_LAB_3_ADJACENCY_LIST_H
#include <stdlib.h>
#include <stdbool.h>
```

```
typedef struct Pair pair;
struct Pair {
    int vertex_number;
    int length;
};
typedef struct Vector vector;
struct Vector {
    size_t size;
    size_t capacity;
    pair* data;
};
void v_init(vector* v);
bool v_empty(vector* v);
size_t v_size(vector* v);
pair v_get(vector* v, size_t index);
void v_put(vector* v, size_t index, pair elem);
void v_resize(vector* v, size_t new_size);
void v_push(vector* v, pair elem);
void v_destroy(vector* v);
typedef struct Adjacency_list adjacency_list;
struct Adjacency_list {
    size_t size;
    vector* vecs;
};
void a_init(adjacency_list* list);
void a_resize(adjacency_list* list, size_t new_size);
void a_destroy(adjacency_list* list);
#endif //OS_LAB_3_ADJACENCY_LIST_H
```

adjacency_list.c

```
#include "adjacency_list.h"
bool v_empty(vector* v) {
    return v->size == 0;
}
size_t v_size(vector* v) {
    return v->size;
}
void v_init(vector* v) {
    v->data = NULL;
    v->size = 0;
    v->capacity = 0;
}
pair v_get(vector* v, size_t index) {
    if (index < v->size) {
        return v->data[index];
    }
}
void v_put(vector* v, size_t index, pair elem) {
    if (index < v->size) {
        v->data[index] = elem;
    }
}
void v_resize(vector* v, size_t new_size) {
    if (new_size == 0) {
        v->capacity = 0;
        v->size = 0;
```

```
        free(v->data);
        v->data = NULL;
    }  else {
        v->data = realloc(v->data, new_size * sizeof(pair));
        v->size = new_size;
        v->capacity = new_size;
    }
}
void v_push(vector* v, pair elem) {
    if (v->size == v->capacity) {
        v->capacity = v->capacity == 0 ? 1 : v->capacity * 2;
        v->data = realloc(v->data, v->capacity * sizeof(pair));
    }
    v->data[v->size] = elem;
    v->size++;
}
void v_destroy(vector* v) {
    free(v->data);
    v->data = NULL;
    v->size = 0;
    v->capacity = 0;
}
void a_init(adjacency_list* list) {
    list->size = 0;
    list->vecs = NULL;
}
void a_resize(adjacency_list* list, size_t new_size) {
    size_t old_size = list->size;
    if (old_size == new_size) {
        return;
    }
    if (old_size > new_size) {
        for (int i = new_size; i < old_size; ++i) {
            v_destroy(&list->vecs[i]);
        }
        list->vecs = realloc(list->vecs, new_size * sizeof(vector));
    } else {
        list->vecs = realloc(list->vecs, new_size * sizeof(vector));
        for (int i = old_size; i < new_size; ++i) {
            v_init(&list->vecs[i]);
        }
    }
    list->size = new_size;
}
void a_destroy(adjacency_list* list) {
    for (int i = 0; i < list->size; ++i) {
        v_destroy(&list->vecs[i]);
    }
    free(list->vecs);
    list->size = 0;
    list->vecs = NULL;
}
```

queue.h

```
#ifndef OS_LAB_3_QUEUE_H
#define OS_LAB_3_QUEUE_H
```

```h
#include <stdbool.h>
#include <stdlib.h>
typedef struct QueueItem queue_item;
struct QueueItem {
    struct QueueItem* next;
    struct QueueItem* prev;
    int value;
};
typedef struct Queue queue;
struct Queue {
    queue_item* head;
    queue_item* tail;
    size_t size;
};
void q_init(queue* q);
int q_top(queue* q);
int q_pop(queue* q);
size_t q_size(queue* q);
void q_push(queue* q, int elem);
bool q_empty(queue* q);
void q_destroy(queue* q);
#endif //OS_LAB_3_QUEUE_H
```

queue.c

```c
#include "queue.h"
void q_init(queue* q) {
    q->head = NULL;
    q->tail = NULL;
    q->size = 0;
}
int q_top(queue* q) {
    return q->head->value;
}
int q_pop(queue* q) {
    int temp = q_top(q);
    queue_item* ptr_to_free = q->head;
    q->head = q->head->next;
    if (q->head == NULL) {
        q->tail = NULL;
    }
    free(ptr_to_free);
    q->size--;
    return temp;
}
void q_push(queue* q, int elem) {
    queue_item* new_elem = malloc(sizeof(queue_item));
    new_elem->value = elem;
    new_elem->next = 0;
    if (q->head == NULL) {
        q->head = new_elem;
        q->tail = new_elem;
        new_elem->prev = 0;
    } else {
        q->tail->next = new_elem;
        new_elem->prev = q->tail;
        q->tail = new_elem;
```

```
    }
    q->size++;
}
bool q_empty(queue* q) {
    return q->head == NULL;
}
size_t q_size(queue* q) {
    return q->size;
}
void q_destroy(queue* q) {
    queue_item* start = q->head;
    while (start != NULL) {
        queue_item* next = start->next;
        free(start);
        start = next;
    }
    q->head = NULL;
    q->tail = NULL;
    q->size = 0;
}
```

Демонстрация работы программы

ilya@ilya-lenovo:~/CLionProjects/OS_lab_3/src/cmake-build-debug$ cat matrix_1

6

0 20 0 10 0 2

0 0 0 1 0 0

0 2 0 0 0 0

0 6 0 0 4 0

0 0 2 0 0 0

0 0 6 0 2 0

ilya@ilya-lenovo:~/CLionProjects/OS_lab_3/src/cmake-build-debug$ cat matrix_2

5

0 1 3 0 2

1 0 5 6 0

3 5 0 1 7

0 6 1 0 8

2 0 7 8 0

ilya@ilya-lenovo:~/CLionProjects/OS_lab_3/src/cmake-build-debug$ ./OS_lab_3 matrix_0 10

Wrong filename

ilya@ilya-lenovo:~/CLionProjects/OS_lab_3/src/cmake-build-debug$ ./OS_lab_3 matrix_1 -5

Incorrect thread count

ilya@ilya-lenovo:~/CLionProjects/OS_lab_3/src/cmake-build-debug$ ./OS_lab_3 matrix_1 1

Enter vertex to start search

1

1:0 2:8 3:6 4:9 5:4 6:2

ilya@ilya-lenovo:~/CLionProjects/OS_lab_3/src/cmake-build-debug$ ./OS_lab_3 matrix_1 5

Enter vertex to start search

7

Vertex number must be < 6

and > 0

Enter vertex to start search

5

1:-1 2:4 3:2 4:5 5:0 6:-1

ilya@ilya-lenovo:~/CLionProjects/OS_lab_3/src/cmake-build-debug$ strace -f ./OS_lab_3 matrix_1 5

execve("./OS_lab_3", ["./OS_lab_3", "matrix_1", "5"], 0x7ffe609d24a8 /* 52 vars */) = 0

brk(NULL) = 0x56051ca8b000

access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No such file or directory)

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=144444, ...}) = 0

mmap(NULL, 144444, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f516f462000

close(3) = 0

access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libpthread.so.0", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0000b\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st_mode=S_IFREG|0755, st_size=144976, ...}) = 0

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f516f460000

mmap(NULL, 2221184, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f516f040000

mprotect(0x7f516f05a000, 2093056, PROT_NONE) = 0

mmap(0x7f516f259000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19000) = 0x7f516f259000

mmap(0x7f516f25b000, 13440, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f516f25b000

close(3) = 0

access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No such file or directory)

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\260\34\2\0\0\0\0\0"..., 832) = 832

fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0

mmap(NULL, 4131552, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f516ec4f000

mprotect(0x7f516ee36000, 2097152, PROT_NONE) = 0

mmap(0x7f516f036000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f516f036000

mmap(0x7f516f03c000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f516f03c000

close(3) = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f516f45d000

arch_prctl(ARCH_SET_FS, 0x7f516f45d740) = 0

mprotect(0x7f516f036000, 16384, PROT_READ) = 0

mprotect(0x7f516f259000, 4096, PROT_READ) = 0

mprotect(0x56051b0ea000, 4096, PROT_READ) = 0

mprotect(0x7f516f486000, 4096, PROT_READ) = 0

munmap(0x7f516f462000, 144444) = 0

set_tid_address(0x7f516f45da10) = 5631

set_robust_list(0x7f516f45da20, 24) = 0

rt_sigaction(SIGRTMIN, {sa_handler=0x7f516f045cb0, sa_mask=[], sa_flags=SA_RESTORER|SA_SIGINFO, sa_restorer=0x7f516f052890}, NULL, 8) = 0

```
rt_sigaction(SIGRT_1, {sa_handler=0x7f516f045d50, sa_mask=[],
sa_flags=SA_RESTORER|SA_RESTART|SA_SIGINFO,
sa_restorer=0x7f516f052890}, NULL, 8) = 0

rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0

prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024,
rlim_max=RLIM64_INFINITY}) = 0

brk(NULL)                               = 0x56051ca8b000

brk(0x56051caac000)                     = 0x56051caac000

openat(AT_FDCWD, "matrix_1", O_RDONLY)  = 3

fstat(3, {st_mode=S_IFREG|0644, st_size=77, ...}) = 0

read(3, "6\n0 20 0 10 0 2\n0 0 0 1 0 0\n0 2 "..., 4096) = 77

close(3)                                = 0

fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0

write(1, "Enter vertex to start search\n", 29Enter vertex to start search

) = 29

fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0

read(0, 1

"1\n", 1024)                    = 2

mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|
MAP_STACK, -1, 0) = 0x7f516e44e000

mprotect(0x7f516e44f000, 8388608, PROT_READ|PROT_WRITE) = 0

clone(strace: Process 5632 attached

child_stack=0x7f516ec4dfb0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|
CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|
CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID,
parent_tidptr=0x7f516ec4e9d0, tls=0x7f516ec4e700,
child_tidptr=0x7f516ec4e9d0) = 5632

[pid  5632] set_robust_list(0x7f516ec4e9e0, 24 <unfinished ...>

[pid  5631] futex(0x7f516ec4e9d0, FUTEX_WAIT, 5632, NULL <unfinished ...>

[pid  5632] <... set_robust_list resumed> ) = 0
```

[pid 5632] mmap(NULL, 134217728, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_NORESERVE, -1, 0) = 0x7f516644e000

[pid 5632] munmap(0x7f516644e000, 29040640) = 0

[pid 5632] munmap(0x7f516c000000, 38068224) = 0

[pid 5632] mprotect(0x7f5168000000, 135168, PROT_READ|PROT_WRITE) = 0

[pid 5632] mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x7f516dc4d000

[pid 5632] mprotect(0x7f516dc4e000, 8388608, PROT_READ|PROT_WRITE) = 0

[pid 5632] clone(strace: Process 5633 attached

child_stack=0x7f516e44cfb0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tidptr=0x7f516e44d9d0, tls=0x7f516e44d700, child_tidptr=0x7f516e44d9d0) = 5633

[pid 5633] set_robust_list(0x7f516e44d9e0, 24 <unfinished ...>

[pid 5632] mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>

[pid 5633] <... set_robust_list resumed>) = 0

[pid 5632] <... mmap resumed>)   = 0x7f516d44c000

[pid 5633] mmap(NULL, 134217728, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_NORESERVE, -1, 0 <unfinished ...>

[pid 5632] mprotect(0x7f516d44d000, 8388608, PROT_READ|PROT_WRITE <unfinished ...>

[pid 5633] <... mmap resumed>)   = 0x7f5160000000

[pid 5632] <... mprotect resumed>)   = 0

[pid 5633] munmap(0x7f5164000000, 67108864) = 0

[pid 5632] clone(<unfinished ...>

[pid 5633] mprotect(0x7f5160000000, 135168, PROT_READ|PROT_WRITE) = 0

strace: Process 5634 attached

[pid 5634] set_robust_list(0x7f516dc4c9e0, 24 <unfinished ...>

[pid 5633] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 5634] <... set_robust_list resumed>) = 0

[pid 5633] <... openat resumed>)   = 3

[pid 5632] <... clone resumed> child_stack=0x7f516dc4bfb0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tidptr=0x7f516dc4c9d0, tls=0x7f516dc4c700, child_tidptr=0x7f516dc4c9d0) = 5634

[pid 5633] fstat(3, <unfinished ...>

[pid 5634] mmap(0x7f5164000000, 67108864, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_NORESERVE, -1, 0 <unfinished ...>

[pid 5633] <... fstat resumed> {st_mode=S_IFREG|0644, st_size=144444, ...}) = 0

[pid 5634] <... mmap resumed>)   = 0x7f5164000000

[pid 5633] mmap(NULL, 144444, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>

[pid 5634] mprotect(0x7f5164000000, 135168, PROT_READ|PROT_WRITE <unfinished ...>

[pid 5633] <... mmap resumed>)   = 0x7f516f462000

[pid 5634] <... mprotect resumed>)   = 0

[pid 5633] close(3 <unfinished ...>

[pid 5634] mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>

[pid 5632] mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>

[pid 5634] <... mmap resumed>)   = 0x7f516cc4b000

[pid 5632] <... mmap resumed>)   = 0x7f516c44a000

[pid 5634] mprotect(0x7f516cc4c000, 8388608, PROT_READ|PROT_WRITE <unfinished ...>

[pid 5632] mprotect(0x7f516c44b000, 8388608, PROT_READ|PROT_WRITE <unfinished ...>

[pid 5634] <... mprotect resumed>)   = 0

[pid 5632] <... mprotect resumed>)   = 0

[pid 5634] clone(<unfinished ...>

[pid 5633] <... close resumed>)   = 0

strace: Process 5635 attached

[pid 5633] access("/etc/ld.so.nohwcap", F_OK <unfinished ...>

[pid 5635] set_robust_list(0x7f516d44b9e0, 24 <unfinished ...>

[pid 5633] <... access resumed>)   = -1 ENOENT (No such file or directory)

[pid 5635] <... set_robust_list resumed>) = 0

[pid 5634] <... clone resumed> child_stack=0x7f516d44afb0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tidptr=0x7f516d44b9d0, tls=0x7f516d44b700, child_tidptr=0x7f516d44b9d0) = 5635

[pid 5635] mmap(NULL, 134217728, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_NORESERVE, -1, 0 <unfinished ...>

[pid 5634] futex(0x7f516d44b9d0, FUTEX_WAIT, 5635, NULL <unfinished ...>

[pid 5635] <... mmap resumed>)   = 0x7f5158000000

[pid 5633] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 5635] munmap(0x7f515c000000, 67108864 <unfinished ...>

[pid 5632] clone(<unfinished ...>

[pid 5635] <... munmap resumed>)   = 0

[pid 5633] <... openat resumed>)   = 3

strace: Process 5636 attached

[pid 5635] mprotect(0x7f5158000000, 135168, PROT_READ|PROT_WRITE <unfinished ...>

[pid 5636] set_robust_list(0x7f516cc4a9e0, 24 <unfinished ...>

[pid 5635] <... mprotect resumed>)   = 0

[pid 5636] <... set_robust_list resumed>) = 0

[pid 5635] futex(0x7f516f487968, FUTEX_WAIT_PRIVATE, 2, NULL <unfinished ...>

[pid 5633] read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\300*\0\0\0\0\0\0"..., 832) = 832

[pid 5632] <... clone resumed> child_stack=0x7f516cc49fb0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tidptr=0x7f516cc4a9d0, tls=0x7f516cc4a700, child_tidptr=0x7f516cc4a9d0) = 5636

[pid 5633] fstat(3, <unfinished ...>

[pid 5632] futex(0x7f516e44d9d0, FUTEX_WAIT, 5633, NULL <unfinished ...>

[pid 5633] <... fstat resumed> {st_mode=S_IFREG|0644, st_size=96616, ...}) = 0

[pid 5636] mmap(0x7f515c000000, 67108864, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_NORESERVE, -1, 0 <unfinished ...>

[pid 5633] mmap(NULL, 2192432, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>

[pid 5636] <... mmap resumed>)   = 0x7f515c000000

[pid 5633] <... mmap resumed>)   = 0x7f516c232000

[pid 5636] mprotect(0x7f515c000000, 135168, PROT_READ|PROT_WRITE <unfinished ...>

[pid 5633] mprotect(0x7f516c249000, 2093056, PROT_NONE <unfinished ...>

[pid 5636] <... mprotect resumed>)   = 0

[pid 5633] <... mprotect resumed>)   = 0

[pid 5636] futex(0x7f516f487968, FUTEX_WAIT_PRIVATE, 2, NULL <unfinished ...>

[pid 5633] mmap(0x7f516c448000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x16000) = 0x7f516c448000

[pid 5633] close(3)   = 0

[pid 5633] mprotect(0x7f516c448000, 4096, PROT_READ) = 0

[pid 5633] munmap(0x7f516f462000, 144444) = 0

[pid 5633] futex(0x7f516f487968, FUTEX_WAKE_PRIVATE, 1 <unfinished ...>

[pid 5635] <... futex resumed>)   = 0

[pid 5633] <... futex resumed>)   = 1

[pid 5635] futex(0x7f516f487968, FUTEX_WAKE_PRIVATE, 1 <unfinished ...>

[pid 5633] futex(0x7f516c4491a0, FUTEX_WAKE_PRIVATE, 2147483647 <unfinished ...>

[pid 5636] <... futex resumed>)   = 0

[pid 5635] <... futex resumed>)   = 1

[pid 5636] futex(0x7f516f487968, FUTEX_WAKE_PRIVATE, 1 <unfinished ...>

[pid 5633] <... futex resumed>)   = 0

[pid 5636] <... futex resumed>)   = 0

[pid 5635] madvise(0x7f516cc4b000, 8368128, MADV_DONTNEED <unfinished ...>

[pid 5636] madvise(0x7f516c44a000, 8368128, MADV_DONTNEED <unfinished ...>

[pid 5635] <... madvise resumed>)   = 0

[pid 5636] <... madvise resumed>)   = 0

[pid 5635] exit(0 <unfinished ...>

[pid 5636] exit(0)   = ?

[pid 5635] <... exit resumed>)   = ?

[pid 5633] madvise(0x7f516dc4d000, 8368128, MADV_DONTNEED <unfinished ...>

[pid 5636] +++ exited with 0 +++

[pid 5635] +++ exited with 0 +++

[pid 5634] <... futex resumed>)   = 0

[pid 5633] <... madvise resumed>)   = 0

[pid 5634] madvise(0x7f516d44c000, 8368128, MADV_DONTNEED <unfinished ...>

[pid 5633] exit(0 <unfinished ...>

[pid 5634] <... madvise resumed>)   = 0

[pid 5633] <... exit resumed>)   = ?

[pid 5634] exit(0 <unfinished ...>

[pid 5633] +++ exited with 0 +++

[pid 5634] <... exit resumed>)   = ?

[pid 5632] <... futex resumed>)   = 0

[pid 5634] +++ exited with 0 +++

[pid 5632] madvise(0x7f516e44e000, 8368128, MADV_DONTNEED) = 0

[pid 5632] exit(0)   = ?

[pid 5631] <... futex resumed>)   = 0

[pid 5632] +++ exited with 0 +++

munmap(0x7f516cc4b000, 8392704)   = 0

write(1, "1:0 2:8 3:6 4:9 5:4 6:2 \n", 251:0 2:8 3:6 4:9 5:4 6:2

) = 25

lseek(0, -1, SEEK_CUR)   = -1 ESPIPE (Illegal seek)

exit_group(0)   = ?

+++ exited with 0 +++

Вывод

В результате данной лабораторной работы мной была изучены тонкости создания многопоточных приложений. Также я получил дополнительные навыки синхронизации потоков и работы с утилитой strace.