

Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

Курсовой проект по курсу

«Операционные системы»

Группа: М8О-209БВ-24

Студент: Крысанов А. Ю.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 20.12.25

Москва, 2025

Постановка задачи

Вариант 37.

1. По конфигурационному файлу в формате yaml, json или ini принимает спроектированный DAG джобов и проверяет на корректность: отсутствие циклов, наличие только одной компоненты связности, наличие стартовых и завершающих джоб. Структура описания джоб и их связей произвольная.
2. При завершении джобы с ошибкой, необходимо прервать выполнение всего DAG'а и всех запущенных джоб.
3. json\Mutex

Общий метод и алгоритм решения

Использованные системные вызовы:

pthread_create() — создание потока для каждой задачи

pthread_join() — ожидание завершения потока

pthread_mutex_lock()/pthread_mutex_unlock() — блокировка/разблокировка мьютексов

pthread_cond_wait()/pthread_cond_broadcast() — ожидание и уведомление о завершении зависимостей

sleep() — имитация выполнения работы задачи

printf() — вывод информации о старте и завершении задач

json-с функции (json_object_from_file, json_object_array_get_idx, ...) — чтение и парсинг JSON

Алгоритм работы:

Загружается JSON-файл с задачами и мьютексами, строится DAG.

Проверяется DAG: отсутствие циклов, связность, наличие стартовых и конечных задач.

Создаются потоки для каждой задачи (pthread_create).

Поток ждет, пока выполнены все зависимости (pthread_cond_wait).

Захватывает мьютекс, если требуется, выполняет работу (sleep) и выводит старт/конец.

Обновляет счётчики зависимостей для дочерних задач и уведомляет другие потоки (pthread_cond_broadcast).

Главный поток ждёт завершения всех потоков (pthread_join) и выводит сообщение об успешном завершении всех задач.

Код программы

main.c

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <pthread.h>

#include <json-c/json.h>


#define MAX_JOBS 32

#define MAX_MUTEX 16

#define MAX_DEPS 10


typedef struct Job {
    char id[32];

    char mutex_name[32];

    char deps[MAX_DEPS][32];

    int deps_count;

    int completed_deps;

    int work_time;

    int done;

    pthread_t thread;
} Job;


typedef struct {
    char name[32];

    pthread_mutex_t mutex;
```

```
} NamedMutex;
```

```
Job jobs[MAX_JOBS];
```

```
NamedMutex mutexes[MAX_MUTEX];
```

```
int job_count = 0;
```

```
int mutex_count = 0;
```

```
pthread_mutex_t global_lock = PTHREAD_MUTEX_INITIALIZER;
```

```
pthread_cond_t cond = PTHREAD_COND_INITIALIZER;
```

```
pthread_mutex_t* find_mutex(const char* name) {
```

```
    for (int i = 0; i < mutex_count; i++) {
```

```
        if (strcmp(mutexes[i].name, name) == 0)
```

```
            return &mutexes[i].mutex;
```

```
    }
```

```
    return NULL;
```

```
}
```

```
int deps_done(Job* job) {
```

```
    return job->completed_deps == job->deps_count;
```

```
}
```

```
int check_no_cycles() {
```

```
    int in_degree[MAX_JOBS] = {0};
```

```
    for (int i = 0; i < job_count; i++) {
```

```
        for (int d = 0; d < jobs[i].deps_count; d++) {
```

```
            int found = 0;
```

```
            for (int j = 0; j < job_count; j++) {
```

```

        if (strcmp(jobs[j].id, jobs[i].deps[d]) == 0) {

            in_degree[i]++;

            found = 1;

        }

    }

    if (!found) {

        printf("ОШИБКА: задача %s зависит от неизвестной задачи %s\n",

            jobs[i].id, jobs[i].deps[d]);

        return 0;

    }

}

```

```

int queue[MAX_JOBS];

```

```

int qh = 0, qt = 0;

```

```

for (int i = 0; i < job_count; i++)

```

```

    if (in_degree[i] == 0)

```

```

        queue[qt++] = i;

```

```

int processed = 0;

```

```

while (qh < qt) {

```

```

    int u = queue[qh++];

```

```

    processed++;

```

```

    for (int i = 0; i < job_count; i++) {

```

```

        for (int d = 0; d < jobs[i].deps_count; d++) {

```

```

            if (strcmp(jobs[i].deps[d], jobs[u].id) == 0) {

```

```

                if (--in_degree[i] == 0)

```

```

        queue[qt++] = i;

    }

}

}

}

if (processed != job_count) {

    printf("ОШИБКА: обнаружен цикл в DAG\n");

    return 0;

}

return 1;

}

void dfs(int v, int visited[]) {

    visited[v] = 1;

    for (int i = 0; i < job_count; i++) {

        for (int d = 0; d < jobs[i].deps_count; d++) {

            if (strcmp(jobs[i].deps[d], jobs[v].id) == 0 && !visited[i])

                dfs(i, visited);

        }

        for (int d = 0; d < jobs[v].deps_count; d++) {

            if (strcmp(jobs[v].deps[d], jobs[i].id) == 0 && !visited[i])

                dfs(i, visited);

        }

    }

}

```

```

int check_single_connected_component() {

```

```

int visited[MAX_JOBS] = {0};

dfs(0, visited);


for (int i = 0; i < job_count; i++) {
    if (!visited[i]) {
        printf("ОШИБКА: DAG состоит из нескольких связанных компонентов\n");
        return 0;
    }
}

return 1;
}


int check_start_and_end_jobs() {
    int start = 0, end = 0;


    for (int i = 0; i < job_count; i++)
        if (jobs[i].deps_count == 0)
            start++;


    for (int i = 0; i < job_count; i++) {
        int is_end = 1;
        for (int j = 0; j < job_count; j++)
            for (int d = 0; d < jobs[j].deps_count; d++)
                if (strcmp(jobs[j].deps[d], jobs[i].id) == 0)
                    is_end = 0;
        if (is_end)
            end++;
    }


    if (start == 0) {

```

```

        printf("ОШИБКА: отсутствуют стартовые задачи\n");

        return 0;
    }

    if (end == 0) {

        printf("ОШИБКА: отсутствуют конечные задачи\n");

        return 0;
    }

    return 1;
}

void* job_runner(void* arg) {

    Job* job = (Job*)arg;

    pthread_mutex_lock(&global_lock);

    while (!deps_done(job))

        pthread_cond_wait(&cond, &global_lock);

    pthread_mutex_unlock(&global_lock);

    pthread_mutex_t* mtx = NULL;

    if (strlen(job->mutex_name) > 0)

        mtx = find_mutex(job->mutex_name);

    if (mtx) pthread_mutex_lock(mtx);

    printf("[START] Job %s\n", job->id);

    sleep(job->work_time);

    printf("[END] Job %s\n", job->id);

```

```

if (mtx) pthread_mutex_unlock(mtx);

pthread_mutex_lock(&global_lock);

job->done = 1;

for (int i = 0; i < job_count; i++)
    for (int d = 0; d < jobs[i].deps_count; d++)
        if (strcmp(jobs[i].deps[d], job->id) == 0)
            jobs[i].completed_deps++;

pthread_cond_broadcast(&cond);
pthread_mutex_unlock(&global_lock);

return NULL;
}

void load_json(const char* filename) {
    struct json_object* root = json_object_from_file(filename);
    if (!root) {
        printf("Невозможно открыть JSON файл\n");
        exit(1);
    }

    struct json_object* jmutexes;
    json_object_object_get_ex(root, "mutexes", &jmutexes);
    mutex_count = json_object_array_length(jmutexes);

    for (int i = 0; i < mutex_count; i++) {
        strcpy(mutexes[i].name,
            json_object_get_string(json_object_array_get_idx(jmutexes, i)));
    }
}

```

```

pthread_mutex_init(&mutexes[i].mutex, NULL);
}

struct json_object* jjobs;
json_object_object_get_ex(root, "jobs", &jjobs);
job_count = json_object_array_length(jjobs);

for (int i = 0; i < job_count; i++) {
    memset(&jobs[i], 0, sizeof(Job));

    struct json_object* j = json_object_array_get_idx(jjobs, i);

    strcpy(jobs[i].id,
        json_object_get_string(json_object_object_get(j, "id")));

    struct json_object* jm;
    if (json_object_object_get_ex(j, "mutex", &jm))
        strcpy(jobs[i].mutex_name, json_object_get_string(jm));
    else
        jobs[i].mutex_name[0] = '\0';

    jobs[i].work_time =
        json_object_get_int(json_object_object_get(j, "work_time"));

    struct json_object* jdeps;
    json_object_object_get_ex(j, "deps", &jdeps);
    jobs[i].deps_count = json_object_array_length(jdeps);

    for (int d = 0; d < jobs[i].deps_count; d++) {
        strcpy(jobs[i].deps[d],

```

```

        json_object_get_string(json_object_array_get_idx(jdeps, d)));
    }
}
}

```

```

int main(int argc, char** argv) {
    if (argc < 2) {
        printf("Использование: %s jobs.json\n", argv[0]);
        return 1;
    }
}

```

```

load_json(argv[1]);

```

```

if (!check_no_cycles()) return 1;
if (!check_single_connected_component()) return 1;
if (!check_start_and_end_jobs()) return 1;

```

```

printf("Проверка DAG пройдена успешно\n");

```

```

for (int i = 0; i < job_count; i++)
    pthread_create(&jobs[i].thread, NULL, job_runner, &jobs[i]);

```

```

for (int i = 0; i < job_count; i++)
    pthread_join(jobs[i].thread, NULL);

```

```

printf("Все задачи выполнены успешно\n");
return 0;

```

```

}

```

diff_DAG.json:

```
{  
  "mutexes": ["m1", "m2"],  
  "jobs": [  
    { "id": "A", "deps": [], "mutex": "m1", "work_time": 2 },  
    { "id": "B", "deps": ["A"], "mutex": "m1", "work_time": 2 },  
    { "id": "C", "deps": ["A"], "mutex": "m2", "work_time": 2 },  
    { "id": "D", "deps": ["B", "C"], "work_time": 1 },  
    { "id": "E", "deps": ["D"], "mutex": "m2", "work_time": 2 },  
    { "id": "F", "deps": ["D"], "mutex": "m1", "work_time": 2 }  
  ]  
}
```

Протокол работы программы

root → /workspace \$./scheduler jsons/diff_DAG.json

Проверка DAG пройдена успешно

[START] Job A

[END] Job A

[START] Job B

[START] Job C

[END] Job B

[END] Job C

[START] Job D

[END] Job D

[START] Job E

[START] Job F

[END] Job E

[END] Job F

Все задачи выполнены успешно

Strace:

```
root → /workspace $ strace -f ./scheduler jsons/diff_DAG.json
execve("./scheduler", ["/workspace/scheduler", "jsons/diff_DAG.json"],
0x7ffcfc16507f0 /* 11 vars */) = 0

brk(NULL)                                = 0x5df06f11b000

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=21940, ...}) = 0

mmap(NULL, 21940, PROT_READ, MAP_PRIVATE, 3, 0) =
0x7a6ec4ed0000

close(3)                                 = 0

openat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/libjson-c.so.5",
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\0@C\0\0\0\0\0"...
, 832) = 832

fstat(3, {st_mode=S_IFREG|0644, st_size=71896, ...}) = 0

mmap(NULL, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7a6ec4ece000

mmap(NULL, 74184, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0)
= 0x7a6ec4ebb000

mprotect(0x7a6ec4ebf000, 53248, PROT_NONE) = 0

mmap(0x7a6ec4ebf000, 36864, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x4000) = 0x7a6ec4ebf000

mmap(0x7a6ec4ec8000, 12288, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xd000) = 0x7a6ec4ec8000
```

```

mmap(0x7a6ec4ecc000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x10000) = 0x7a6ec4ecc000

close(3) = 0

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\3\0>\0\1\0\0\0
1\0\0\0\0\0\0"... , 832) = 832

fstat(3, {st_mode=S_IFREG|0755, st_size=149520, ...}) = 0

mmap(NULL, 136304, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0)
= 0x7a6ec4e99000

mmap(0x7a6ec4e9f000, 65536, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x6000) = 0x7a6ec4e9f000

mmap(0x7a6ec4eaf000, 24576, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x16000) = 0x7a6ec4eaf000

mmap(0x7a6ec4eb5000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b000) = 0x7a6ec4eb5000

mmap(0x7a6ec4eb7000, 13424, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7a6ec4eb7000

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\3\0>\0\1\0\0\0\0\260>\2\0\0\0\0\0"
..., 832) = 832

fstat(3, {st_mode=S_IFREG|0755, st_size=1901536, ...}) = 0

mmap(NULL, 1914496, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3,
0) = 0x7a6ec4cc5000

mmap(0x7a6ec4ce7000, 1413120, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x22000) = 0x7a6ec4ce7000

mmap(0x7a6ec4e40000, 323584, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x17b000) =
0x7a6ec4e40000

```

```
mmap(0x7a6ec4e8f000, 24576, PROT_READ|PROT_WRITE,  
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1c9000) =  
0x7a6ec4e8f000
```

```
mmap(0x7a6ec4e95000, 13952, PROT_READ|PROT_WRITE,  
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7a6ec4e95000
```

```
close(3) = 0
```

```
mmap(NULL, 12288, PROT_READ|PROT_WRITE,  
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7a6ec4cc2000
```

```
arch_prctl(ARCH_SET_FS, 0x7a6ec4cc2740) = 0
```

```
mprotect(0x7a6ec4e8f000, 16384, PROT_READ) = 0
```

```
mprotect(0x7a6ec4eb5000, 4096, PROT_READ) = 0
```

```
mprotect(0x7a6ec4ecc000, 4096, PROT_READ) = 0
```

```
mprotect(0x5df06e64f000, 4096, PROT_READ) = 0
```

```
mprotect(0x7a6ec4f00000, 4096, PROT_READ) = 0
```

```
munmap(0x7a6ec4ed0000, 21940) = 0
```

```
set_tid_address(0x7a6ec4cc2a10) = 574
```

```
set_robust_list(0x7a6ec4cc2a20, 24) = 0
```

```
rt_sigaction(SIGRTMIN, {sa_handler=0x7a6ec4e9f690, sa_mask=[],  
sa_flags=SA_RESTORER|SA_SIGINFO, sa_restorer=0x7a6ec4eac140},  
NULL, 8) = 0
```

```
rt_sigaction(SIGRT_1, {sa_handler=0x7a6ec4e9f730, sa_mask=[],  
sa_flags=SA_RESTORER|SA_RESTART|SA_SIGINFO,  
sa_restorer=0x7a6ec4eac140}, 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
```

```
openat(AT_FDCWD, "jsons/diff_DAG.json", O_RDONLY) = 3
```

```
brk(NULL) = 0x5df06f11b000
```

```
brk(0x5df06f13c000) = 0x5df06f13c000
```

```

    read(3, "{\r\n  \"mutexes\": [\"m1\", \"m2\"],\r\n \"...\", 4096)
= 434

    read(3, "", 4096) = 0

    stat("/dev/urandom", {st_mode=S_IFCHR|0666,
st_rdev=makedev(0x1, 0x9), ...}) = 0

    openat(AT_FDCWD, "/dev/urandom", O_RDONLY) = 4

    read(4, "\314\254~\30", 4) = 4

    close(4) = 0

    close(3) = 0

    fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0),
...}) = 0

    write(1,
"\320\237\321\200\320\276\320\262\320\265\321\200\320\272\320\260
DAG \320\277\321\200\320\276\320\271\320\264\320"... , 53Проверка
DAG пройдена успешно

) = 53

    mmap(NULL, 8392704, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x7a6ec44c1000

    mprotect(0x7a6ec44c2000, 8388608, PROT_READ|PROT_WRITE) = 0

    clone(child_stack=0x7a6ec4cc0fb0,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CL
ONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID
strace: Process 575 attached

, parent_tid=[575], tls=0x7a6ec4cc1700,
child_tidptr=0x7a6ec4cc19d0) = 575

[pid 575] set_robust_list(0x7a6ec4cc19e0, 24 <unfinished
...>

[pid 574] mmap(NULL, 8392704, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>

[pid 575] <... set_robust_list resumed>) = 0

[pid 574] <... mmap resumed> = 0x7a6ec3cc0000

```

```
[pid 575] write(1, "[START] Job A\n", 14[START] Job A
<unfinished ...>
```

```
[pid 574] mprotect(0x7a6ec3cc1000, 8388608,
PROT_READ|PROT_WRITE <unfinished ...>
```

```
[pid 575] <... write resumed>) = 14
```

```
[pid 574] <... mprotect resumed>) = 0
```

```
[pid 575] clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2,
tv_nsec=0}, <unfinished ...>
```

```
[pid 574] clone(child_stack=0x7a6ec44bffb0,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CL
ONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID
strace: Process 576 attached
```

```
, parent_tid=[576], tls=0x7a6ec44c0700,
child_tidptr=0x7a6ec44c09d0) = 576
```

```
[pid 576] set_robust_list(0x7a6ec44c09e0, 24 <unfinished
...>
```

```
[pid 574] mmap(NULL, 8392704, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>
```

```
[pid 576] <... set_robust_list resumed>) = 0
```

```
[pid 574] <... mmap resumed>) = 0x7a6ec34bf000
```

```
[pid 576] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>
```

```
[pid 574] mprotect(0x7a6ec34c0000, 8388608,
PROT_READ|PROT_WRITE) = 0
```

```
[pid 574] clone(child_stack=0x7a6ec3cbefb0,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CL
ONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID
strace: Process 577 attached
```

```
, parent_tid=[577], tls=0x7a6ec3cbf700,
child_tidptr=0x7a6ec3cbf9d0) = 577
```

```
[pid 577] set_robust_list(0x7a6ec3cbf9e0, 24 <unfinished
...>
```

```

    [pid  574] mmap(NULL, 8392704, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>

    [pid  577] <... set_robust_list resumed>) = 0

    [pid  574] <... mmap resumed>                = 0x7a6ec2cbe000

    [pid  577] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  574] mprotect(0x7a6ec2cbf000, 8388608,
PROT_READ|PROT_WRITE) = 0

    [pid  574] clone(child_stack=0x7a6ec34bdfb0,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CL
ONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID
strace: Process 578 attached

    , parent_tid=[578], tls=0x7a6ec34be700,
child_tidptr=0x7a6ec34be9d0) = 578

    [pid  578] set_robust_list(0x7a6ec34be9e0, 24 <unfinished
...>

    [pid  574] mmap(NULL, 8392704, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>

    [pid  578] <... set_robust_list resumed>) = 0

    [pid  574] <... mmap resumed>                = 0x7a6ec24bd000

    [pid  578] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  574] mprotect(0x7a6ec24be000, 8388608,
PROT_READ|PROT_WRITE) = 0

    [pid  574] clone(child_stack=0x7a6ec2cbcfb0,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CL
ONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID
strace: Process 579 attached

    , parent_tid=[579], tls=0x7a6ec2cbd700,
child_tidptr=0x7a6ec2cbd9d0) = 579

    [pid  579] set_robust_list(0x7a6ec2cbd9e0, 24 <unfinished
...>

```

```

    [pid  574] mmap(NULL, 8392704, PROT_NONE,
MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>

    [pid  579] <... set_robust_list resumed>) = 0

    [pid  574] <... mmap resumed>)          = 0x7a6ec1cbc000

    [pid  579] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  574] mprotect(0x7a6ec1cbd000, 8388608,
PROT_READ|PROT_WRITE) = 0

    [pid  574] clone(child_stack=0x7a6ec24bbfb0,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CL
ONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID
strace: Process 580 attached

    , parent_tid=[580], tls=0x7a6ec24bc700,
child_tidptr=0x7a6ec24bc9d0) = 580

    [pid  580] set_robust_list(0x7a6ec24bc9e0, 24 <unfinished
...>

    [pid  574] futex(0x7a6ec4cc19d0, FUTEX_WAIT, 575, NULL
<unfinished ...>

    [pid  580] <... set_robust_list resumed>) = 0

    [pid  580] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  575] <... clock_nanosleep resumed>0x7a6ec4cc0e80) = 0

    [pid  575] write(1, "[END]  Job A\n", 14[END]  Job A
) = 14

    [pid  575] futex(0x5df06e653908, FUTEX_WAKE_PRIVATE,
2147483647 <unfinished ...>

    [pid  578] <... futex resumed>)          = 0

    [pid  577] <... futex resumed>)          = 0

    [pid  576] <... futex resumed>)          = 0

    [pid  575] <... futex resumed>)          = 5

```

```

    [pid  580] <... futex resumed>)          = 0

    [pid  579] <... futex resumed>)          = 0

    [pid  578] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid  577] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid  576] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid  580] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  579] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  578] <... futex resumed>)          = -1 EAGAIN (Resource
temporarily unavailable)

    [pid  575] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  580] <... futex resumed>)          = 0

    [pid  579] <... futex resumed>)          = 0

    [pid  577] <... futex resumed>)          = -1 EAGAIN (Resource
temporarily unavailable)

    [pid  576] <... futex resumed>)          = -1 EAGAIN (Resource
temporarily unavailable)

    [pid  580] futex(0x5df06e65390c, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  579] futex(0x5df06e65390c, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  578] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  577] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  575] <... futex resumed>)          = 0

```

```

    [pid  578] <... futex resumed>)          = 0

    [pid  576] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  577] <... futex resumed>)          = 0

    [pid  576] <... futex resumed>)          = 0

    [pid  578] futex(0x5df06e65390c, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  577] write(1, "[START] Job C\n", 14 <unfinished ...>

    [pid  576] futex(0x7a6ec4e95990, FUTEX_WAIT_PRIVATE, 2,
NULL[START] Job C
    <unfinished ...>

    [pid  577] <... write resumed>)          = 14

    [pid  575] madvise(0x7a6ec44c1000, 8368128, MADV_DONTNEED
<unfinished ...>

    [pid  577] futex(0x7a6ec4e95990, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  575] <... madvise resumed>)        = 0

    [pid  577] <... futex resumed>)          = 1

    [pid  576] <... futex resumed>)          = 0

    [pid  577] clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2,
tv_nsec=0}, <unfinished ...>

    [pid  576] write(1, "[START] Job B\n", 14 <unfinished ...>

    [pid  575] exit(0[START] Job B
)
                                = ?

    [pid  576] <... write resumed>)          = 14

    [pid  574] <... futex resumed>)          = 0

    [pid  576] futex(0x7a6ec4e95990, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  575] +++ exited with 0 +++

```

```

    [pid 574] futex(0x7a6ec44c09d0, FUTEX_WAIT, 576, NULL
<unfinished ...>

    [pid 576] <... futex resumed>)                = 0

    [pid 576] clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2,
tv_nsec=0}, <unfinished ...>

    [pid 577] <... clock_nanosleep resumed>0x7a6ec3cbee80) = 0
    [pid 576] <... clock_nanosleep resumed>0x7a6ec44bfe80) = 0
    [pid 577] write(1, "[END]   Job C\n", 14[END]   Job C
    <unfinished ...>

    [pid 576] futex(0x7a6ec4e95990, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid 577] <... write resumed>)                = 14
    [pid 577] futex(0x7a6ec4e95990, FUTEX_WAKE_PRIVATE, 1) = 1
    [pid 576] <... futex resumed>)                = 0

    [pid 577] futex(0x5df06e65390c, FUTEX_WAKE_PRIVATE,
2147483647 <unfinished ...>

    [pid 576] write(1, "[END]   Job B\n", 14 <unfinished ...>
    [pid 580] <... futex resumed>)                = 0
    [END]   Job B

    [pid 579] <... futex resumed>)                = 0
    [pid 578] <... futex resumed>)                = 0
    [pid 577] <... futex resumed>)                = 3

    [pid 580] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid 579] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid 576] <... write resumed>)                = 14

    [pid 580] <... futex resumed>)                = -1 EAGAIN (Resource
temporarily unavailable)

```

[pid 579] <... futex resumed> = -1 EAGAIN (Resource temporarily unavailable)

[pid 578] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

[pid 577] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

[pid 580] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

[pid 579] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

[pid 576] futex(0x7a6ec4e95990, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

[pid 578] <... futex resumed> = 0

[pid 577] <... futex resumed> = 0

[pid 580] <... futex resumed> = 0

[pid 579] <... futex resumed> = 0

[pid 578] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

[pid 576] <... futex resumed> = 0

[pid 580] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

[pid 579] futex(0x5df06e653908, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

[pid 577] madvise(0x7a6ec34bf000, 8368128, MADV_DONTNEED
<unfinished ...>

[pid 576] futex(0x5df06e653908, FUTEX_WAKE_PRIVATE,
2147483647 <unfinished ...>

[pid 580] <... futex resumed> = -1 EAGAIN (Resource temporarily unavailable)

[pid 579] <... futex resumed> = -1 EAGAIN (Resource temporarily unavailable)

```

    [pid  580] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid  578] <... futex resumed>)                = 0
    [pid  577] <... madvise resumed>)                = 0
    [pid  576] <... futex resumed>)                = 1
    [pid  579] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid  578] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid  577] exit(0 <unfinished ...>

    [pid  576] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  579] <... futex resumed>)                = -1 EAGAIN (Resource
temporarily unavailable)
    [pid  578] <... futex resumed>)                = -1 EAGAIN (Resource
temporarily unavailable)
    [pid  579] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  577] <... exit resumed>)                = ?
    [pid  580] <... futex resumed>)                = 0
    [pid  579] <... futex resumed>)                = 0
    [pid  578] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  577] +++ exited with 0 +++
    [pid  576] <... futex resumed>)                = 1
    [pid  579] futex(0x5df06e65390c, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>

    [pid  578] <... futex resumed>)                = 0
    [pid  576] madvise(0x7a6ec3cc0000, 8368128, MADV_DONTNEED
<unfinished ...>

```

```

    [pid  580] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  578] write(1, "[START] Job D\n", 14 <unfinished ...>
    [pid  576] <... madvise resumed>)          = 0
    [pid  580] <... futex resumed>)            = 0
    [pid  576] exit(0 <unfinished ...>
    [pid  580] futex(0x5df06e65390c, FUTEX_WAIT_PRIVATE, 0, NULL
<unfinished ...>
    [pid  576] <... exit resumed>)              = ?
    [START] Job D
    [pid  574] <... futex resumed>)              = 0
    [pid  578] <... write resumed>)              = 14
    [pid  576] +++ exited with 0 +++
    [pid  574] futex(0x7a6ec34be9d0, FUTEX_WAIT, 578, NULL
<unfinished ...>
    [pid  578] clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=1,
tv_nsec=0}, 0x7a6ec34bde80) = 0
    [pid  578] write(1, "[END]   Job D\n", 14[END]   Job D
) = 14
    [pid  578] futex(0x5df06e65390c, FUTEX_WAKE_PRIVATE,
2147483647 <unfinished ...>
    [pid  580] <... futex resumed>)              = 0
    [pid  579] <... futex resumed>)              = 0
    [pid  578] <... futex resumed>)              = 2
    [pid  580] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>
    [pid  579] futex(0x5df06e6538a0, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

```

```

    [pid  578] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  580] <... futex resumed>)                = -1 EAGAIN (Resource
temporarily unavailable)

    [pid  579] <... futex resumed>)                = -1 EAGAIN (Resource
temporarily unavailable)

    [pid  580] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  578] <... futex resumed>)                = 0

    [pid  579] futex(0x5df06e6538a0, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  580] <... futex resumed>)                = 0

    [pid  578] madvise(0x7a6ec2cbe000, 8368128, MADV_DONTNEED
<unfinished ...>

    [pid  580] write(1, "[START] Job F\n", 14 <unfinished ...>

    [pid  579] <... futex resumed>)                = 0

    [pid  578] <... madvise resumed>)              = 0

    [START] Job F

    [pid  579] futex(0x7a6ec4e95990, FUTEX_WAIT_PRIVATE, 2, NULL
<unfinished ...>

    [pid  578] exit(0 <unfinished ...>

    [pid  580] <... write resumed>)                = 14

    [pid  578] <... exit resumed>)                = ?

    [pid  580] futex(0x7a6ec4e95990, FUTEX_WAKE_PRIVATE, 1
<unfinished ...>

    [pid  574] <... futex resumed>)                = 0

    [pid  580] <... futex resumed>)                = 1

    [pid  579] <... futex resumed>)                = 0

    [pid  578] +++ exited with 0 +++

```

[pid 574] futex(0x7a6ec2cbd9d0, FUTEX_WAIT, 579, NULL
<unfinished ...>

[pid 580] clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2,
tv_nsec=0}, <unfinished ...>

[pid 579] write(1, "[START] Job E\n", 14[START] Job E
) = 14

[pid 579] futex(0x7a6ec4e95990, FUTEX_WAKE_PRIVATE, 1) = 0

[pid 579] clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2,
tv_nsec=0}, <unfinished ...>

[pid 574] <... futex resumed> = ? ERESTARTSYS (To be
restarted if SA_RESTART is set)

[pid 574] --- SIGWINCH {si_signo=SIGWINCH,
si_code=SI_KERNEL} ---

[pid 574] futex(0x7a6ec2cbd9d0, FUTEX_WAIT, 579, NULL) = ?
ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 574] --- SIGWINCH {si_signo=SIGWINCH,
si_code=SI_KERNEL} ---

[pid 574] futex(0x7a6ec2cbd9d0, FUTEX_WAIT, 579, NULL
<unfinished ...>

[pid 580] <... clock_nanosleep resumed>0x7a6ec24bbe80) = 0

[pid 580] write(1, "[END] Job F\n", 14[END] Job F
) = 14

[pid 580] madvise(0x7a6ec1cbc000, 8368128, MADV_DONTNEED) =
0

[pid 580] exit(0) = ?

[pid 580] +++ exited with 0 +++

[pid 579] <... clock_nanosleep resumed>0x7a6ec2cbce80) = 0

[pid 579] write(1, "[END] Job E\n", 14[END] Job E
) = 14

```

0      [pid  579] madvise(0x7a6ec24bd000, 8368128, MADV_DONTNEED) =
0
      [pid  579] exit(0)                                = ?
      [pid  574] <... futex resumed>                    = 0
      [pid  579] +++ exited with 0 +++
      munmap(0x7a6ec44c1000, 8392704)                   = 0
      munmap(0x7a6ec3cc0000, 8392704)                   = 0
      write(1, "\320\222\321\201\320\265
\320\267\320\260\320\264\320\260\321\207\320\270
\320\262\321\213\320\277\320\276\320\273\320\275"... , 54Все
задачи выполнены успешно
) = 54
      exit_group(0)                                     = ?
      +++ exited with 0 +++

```

Вывод:

В ходе курсового проекта был разработан планировщик выполнения задач (джобов), организованных в виде направленного ациклического графа (DAG). Программа реализована на языке C с использованием системных вызовов POSIX и синхронизации через мьютексы. Разработанный планировщик DAG полностью соответствует всем требованиям курсового проекта. Программа демонстрирует эффективное использование системных вызовов POSIX, правильную работу с многопоточностью и синхронизацией, а также надежную обработку ошибок. Программа успешно решает поставленную задачу: принимает описание DAG джобов, проверяет его корректность и выполняет с соблюдением зависимостей, немедленно прерывая выполнение при любой ошибке.