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-206Б-22

Студент: Коломытцева Е.А.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 01.03.2024

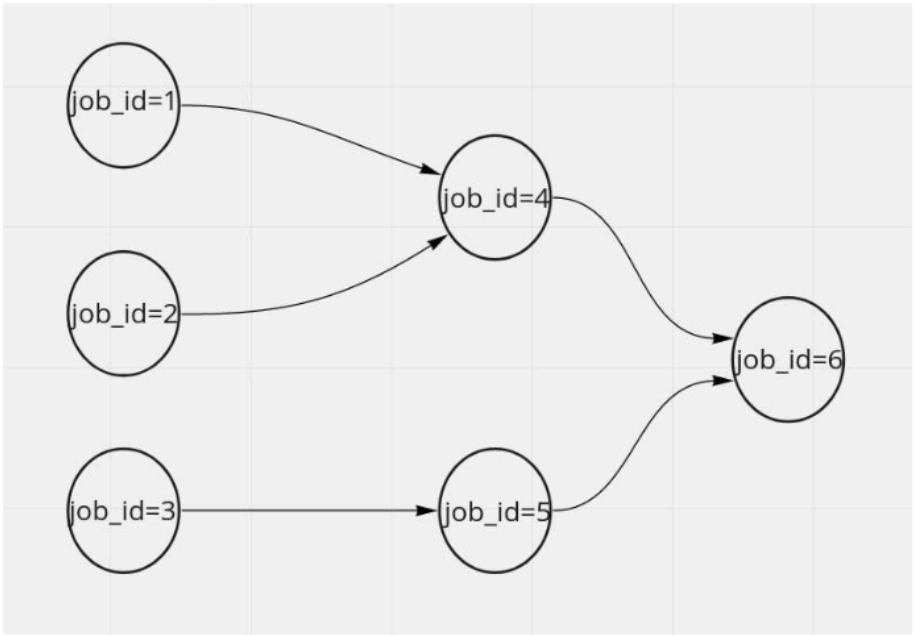
Москва, 2024

Постановка задачи

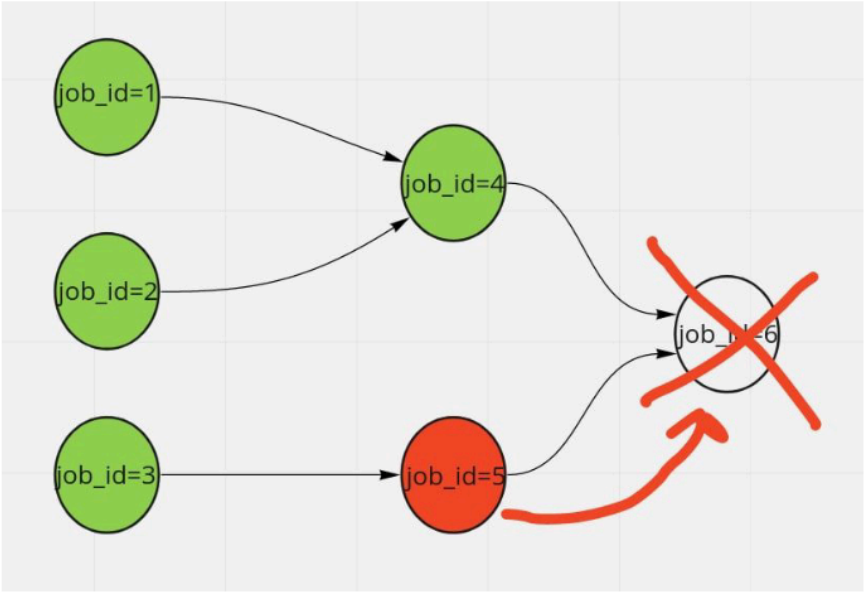
Вариант 2 на 3.

На языке C\C++ написать программу, которая:

По конфигурационному файлу в формате yaml, json или ini принимает спроектированный DAG джобов и проверяет на корректность: отсутствие циклов, наличие только одной компоненты связанности, наличие стартовых и завершающих джоб. Структура описания джоб и их связей произвольная.



При завершении джобы с ошибкой, необходимо прервать выполнение всего DAG’а и всех запущенных джоб.



Код программы

main.cpp:

```
#include <stdio.h>
#include <sys/wait.h>
#include <unistd.h>

#include <iostream>
#include <map>
#include <string>
#include <vector>

#include "graph.h"
#include "parser.h"

using namespace std;

map<int, string> CreateDictionary(vector<Configuration> configs) {
 map<int, string> paths_dictionary;  // словарь <id joba>:<исполняемый файл>
 for (auto config : configs) {
   paths_dictionary[config.id] = config.path;
 }
 return paths_dictionary;
}

int main(int argc, char* argv[]) {
 if (argc < 2) {
   cout << "Error: Not input ini file!\n";
   exit(1);
 }

 vector<Configuration> configs;

 configs = IniParser(argv[1]);

 Graph dag_jobs = CreateGraph(configs);
 if (dag_jobs.isEmpty()) {
   cout << "The graph is empty!\n";
   exit(2);
 }

 // dag_jobs.print();

 vector<int> visited(dag_jobs.getSize(), 0);

 for (auto end_job : dag_jobs.end_jobs_id) {
```

```
  if (IsCycle(end_job, dag_jobs, visited)) {
    cout << "The graph has a cycle!\n";
    exit(3);
  }
}

vector<int> traveled(dag_jobs.getSize(), 0);
vector<int> path = BFS(dag_jobs, traveled);

map<int, string> job_paths = CreateDictionary(configs);

int string_size = path.size() + 1;

char state[string_size];
for (int i = 0; i < string_size; ++i) {
  state[i] = '0';
}

for (int id = path.size() - 1; id >= 0; --id) {
  int pipe_parent[2];
  int pipe_child[2];
  if (pipe(pipe_parent) == -1) {
    fprintf(stderr, "Pipe Failed");
    exit(1);
  }

  if (pipe(pipe_child) == -1) {
    fprintf(stderr, "Pipe Failed");
    exit(1);
  }
  pid_t process_id = fork();

  if (process_id < 0) {
    printf("Error: Fork\n");
    exit(1);
  }

  if (process_id > 0) {
    close(pipe_parent[0]);

    write(pipe_parent[1], state, sizeof(char) * (string_size + 1));
    close(pipe_parent[1]);

    wait(NULL);

    close(pipe_child[1]);

    read(pipe_child[0], state, sizeof(char) * (string_size + 1));
```

```
    close(pipe_child[0]);

  } else {
    close(pipe_parent[1]);

    if (dup2(pipe_child[1], 2) == -1) {
      printf("Error: Dup2\n");
      exit(2);
    }

    char received[string_size];

    read(pipe_parent[0], received, sizeof(char) * (string_size + 1));

    close(pipe_parent[0]);

    close(pipe_child[0]);

    string adjacency;
    for (int i = 0; i < dag_jobs.adjacency[path[id]].size(); ++i) {
      adjacency.push_back(dag_jobs.adjacency[path[id]][i] + '0');
    }
    char ID[2];
    sprintf(ID, "%d", path[id]);

    execl(job_paths[path[id]].c_str(), job_paths[path[id]].c_str(), ID,
          received, adjacency.c_str(), NULL);
  }
 }

 for (int id = 1; id < dag_jobs.getSize(); ++id) {
   cout << "Result for job_id_" << id << " is " << state[id] << '\n';
 }

 return 0;
}
```

parser.cpp:

```
#include "parser.h"

#include <fstream>
#include <iostream>
#include <sstream>
#include <string>
#include <vector>
```

```
using namespace std;

vector<Configuration> IniParser(string nameFile) {
 ifstream iniFile;
 iniFile.open(nameFile);
 if (!iniFile.is_open()) {
   cout << "The file was not open!" << '\n';
   exit(1);
 }
 vector<Configuration> configs;
 string line;
 Configuration currentConfig;
 while (getline(iniFile, line)) {
   if (line[0] == '[') {
     continue;
   } else if (line.substr(0, 2) == "id") {
     currentConfig.id = stoi(line.substr(5, line.size() - 5));
   } else if (line.substr(0, 6) == "parent") {
     currentConfig.parents.push_back(stoi(line.substr(9, line.size() - 9)));
   } else if (line.substr(0, 4) == "path") {
     currentConfig.path = line.substr(8, line.size() - 10);
   } else {
     configs.push_back(currentConfig);
     currentConfig.parents.clear();
   }
 }
 configs.push_back(currentConfig);

 iniFile.close();
 return configs;
}
```

parser.h:

```
#ifndef __PARSER_H__
#define __PARSER_H__

#include <iostream>
#include <string>
#include <vector>

class Configuration {
public:
 int id;
 std::vector<int> parents;
 std::string path;
```

```
 void printConfiguration() {
   for (int i = 0; i < parents.size(); ++i)
     std::cout << id << '\n' << parents[i] << '\n' << path << "\n\n";
 }
};

std::vector<Configuration> IniParser(std::string);
#endif
```

graph.cpp:

```
#include "graph.h"

#include <queue>
#include <vector>

#include "parser.h"

using namespace std;

Graph CreateGraph(vector<Configuration> configs) {
 Graph graph(configs.size() + 1);
 for (auto config : configs) {
   for (auto parent : config.parents) {
     if (parent == 0) {
       graph.end_jobs_id.push_back(config.id);
     }
     graph.addNode(parent, config.id);
   }
 }

 if (graph.end_jobs_id.empty()) {
   cout << "Graph doesn't have End Jobs\n";
   exit(1);
 }
 return graph;
}

int IsCycle(int vertex, Graph &graph, vector<int> &visited) {
 visited[vertex] = 1;
 int result = 0;
 for (auto neighbor : graph.adjacency[vertex]) {
   if (visited[neighbor] == 0) {
     result = IsCycle(neighbor, graph, visited);
     if (result) {
       break;
```

```
    }
   } else if (visited[neighbor] == 1) {
    result = 1;
    break;
   }
  }
  visited[vertex] = 2;
  return result;
}

vector<int> BFS(Graph &graph, vector<int> &visited) {
 vector<int> path;
 queue<int> line;
 int current_vertex;
 for (auto id : graph.end_jobs_id) {
  line.push(id);
 }
 while (!line.empty()) {
  current_vertex = line.front();
  line.pop();
  path.push_back(current_vertex);
  for (int neighbor : graph.adjacency[current_vertex]) {
   if (visited[neighbor] == 0) {
    line.push(neighbor);
    visited[neighbor] = 1;
   }
  }
 }
 return path;
}
```

graph.h:

```
#ifndef __GRAPH_H__
#define __GRAPH_H__

#include <queue>
#include <vector>
#include "parser.h"

class Graph {
public:
  int graph_size = 0;
  std::vector<int> end_jobs_id;
  std::vector<std::vector<int>> adjacency;

  Graph(int size) {
```

```
        graph_size = size;
        adjacency.resize(size);
    }
    void addNode(int startNode, int endNode) {
        adjacency[startNode].push_back(endNode);
    }

    int getSize() {
        return graph_size;
    }

    int isEmpty() {
        return graph_size == 0;
    }

    void print() {
        for (int startNode = 1; startNode < graph_size; ++startNode) {
            for (int vertex: adjacency[startNode]) {
                std::cout << startNode << ' ' << vertex << "\n";
            }
        }
    }
};

Graph CreateGraph(std::vector<Configuration>);
std::vector<int> BFS(Graph &, std::vector<int> &);
int IsCycle(int, Graph &, std::vector<int> &);

#endif
```

adder.c:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

int main(int argc, char* argv[]) {
 if (argc != 4) {
   printf("Invalid Arguments\n");
   exit(2);
 }

 int id = atoi(argv[1]);
 char* state = argv[2];
 char* adjacency = argv[3];
 // printf("I receive id - %d, state - %s, adjacency - %s\n", id, state,
```

```
//        adjacency);

 int current_sum = 0;
 for (int i = 0; i < strlen(adjacency); ++i) {
   current_sum += state[adjacency[i] - '0'] - '0';
 }
 ++current_sum;

 state[id] = current_sum + '0';
 printf("State now %s\n", state);
 write(2, state, sizeof(char*));
 return 0;
}
```

Протокол работы программы

```
❯ ./main ./configs/example.ini
State now 0001000
State now 0011000
State now 0111000
State now 0111020
State now 0111320
State now 0111326
Result for job_id_1 is 1
Result for job_id_2 is 1
Result for job_id_3 is 1
Result for job_id_4 is 3
Result for job_id_5 is 2
Result for job_id_6 is 6
❯ ./main ./configs/cycle.ini
The graph has a cycle!
❯ ./main ./configs/no-end-jobs.ini
Graph doesn't have End Jobs
```

strace

```
❯ strace ./main ./configs/example.ini
execve("./main", ["./main", "./configs/example.ini"], 0x7ffe942398a8 /* 47 vars */) = 0
brk(NULL)                               = 0x557846f8d000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffc25cb50b0) = -1 EINVAL (Недопустимый
аргумент)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x7f6587c40000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (Нет такого файла или каталога)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=121311, ...}, AT_EMPTY_PATH) = 0
```

```
mmap(NULL, 121311, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f6587c22000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libstdc++.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"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2522552, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 2535872, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f6587800000
mmap(0x7f658789c000, 1249280, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9c000) = 0x7f658789c000
mmap(0x7f65879cd000, 577536, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1cd000) = 0x7f65879cd000
mmap(0x7f6587a5a000, 57344, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x25a000) = 0x7f6587a5a000
mmap(0x7f6587a68000, 12736, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f6587a68000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", 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\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=141872, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 144232, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f6587bfe000
mmap(0x7f6587c01000, 110592, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000) = 0x7f6587c01000
mmap(0x7f6587c1c000, 16384, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e000) = 0x7f6587c1c000
mmap(0x7f6587c20000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x21000) = 0x7f6587c20000
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<\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
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2072888, ...}, 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, 2117488, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f6587400000
mmap(0x7f6587422000, 1540096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x22000) = 0x7f6587422000
mmap(0x7f658759a000, 360448, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19a000) = 0x7f658759a000
mmap(0x7f65875f2000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1f1000) = 0x7f65875f2000
mmap(0x7f65875f8000, 53104, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f65875f8000
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"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=948816, ...}, AT_EMPTY_PATH) = 0
```

```
mmap(NULL, 950520, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f6587b15000
mmap(0x7f6587b23000, 516096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe000) = 0x7f6587b23000
mmap(0x7f6587ba1000, 372736, PROT_READ,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x8c000) = 0x7f6587ba1000
mmap(0x7f6587bfc000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe6000) = 0x7f6587bfc000
close(3)                                = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x7f6587b13000
arch_prctl(ARCH_SET_FS, 0x7f6587b14440) = 0
set_tid_address(0x7f6587b14710)         = 131109
set_robust_list(0x7f6587b14720, 24)     = 0
rseq(0x7f6587b14d60, 0x20, 0, 0x53053053) = 0
mprotect(0x7f65875f2000, 16384, PROT_READ) = 0
mprotect(0x7f6587bfc000, 4096, PROT_READ) = 0
mprotect(0x7f6587c20000, 4096, PROT_READ) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x7f6587b11000
mprotect(0x7f6587a5a000, 45056, PROT_READ) = 0
mprotect(0x55784503e000, 4096, PROT_READ) = 0
mprotect(0x7f6587c75000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) =
0
munmap(0x7f6587c22000, 121311)          = 0
futex(0x7f6587a687fc, FUTEX_WAKE_PRIVATE, 2147483647) = 0
getrandom("\x5f\x52\xdf\x29\x65\x26\xbd\x18", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x557846f8d000
brk(0x557846fae000)                     = 0x557846fae000
openat(AT_FDCWD, "./configs/example.ini", O_RDONLY) = 3
read(3, "[job_id_1]\r\nid = 1\r\nparent = 4\r\n"..., 8191) = 311
read(3, "", 8191)                       = 0
close(3)                                = 0
pipe2([3, 4], 0)                        = 0
pipe2([5, 6], 0)                        = 0
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f6587b14710) = 131110
close(3)                                = 0
write(4, "0000000\0", 8)                = 8
close(4)                                = 0
wait4(-1, State now 0001000
NULL, 0, NULL)                = 131110
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=131110, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
close(6)                                = 0
read(5, "0001000\0", 8)                 = 8
```

```
close(5)                        = 0
pipe2([3, 4], 0)                  = 0
pipe2([5, 6], 0)                  = 0
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f6587b14710) = 131111
close(3)                        = 0
write(4, "0001000\0", 8)              = 8
close(4)                        = 0
wait4(-1, State now 0011000
NULL, 0, NULL)              = 131111
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=131111, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
close(6)                        = 0
read(5, "0011000\0", 8)               = 8
close(5)                        = 0
pipe2([3, 4], 0)                  = 0
pipe2([5, 6], 0)                  = 0
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f6587b14710) = 131112
close(3)                        = 0
write(4, "0011000\0", 8)              = 8
close(4)                        = 0
wait4(-1, State now 0111000
NULL, 0, NULL)              = 131112
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=131112, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
close(6)                        = 0
read(5, "0111000\0", 8)               = 8
close(5)                        = 0
pipe2([3, 4], 0)                  = 0
pipe2([5, 6], 0)                  = 0
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f6587b14710) = 131113
close(3)                        = 0
write(4, "0111000\0", 8)              = 8
close(4)                        = 0
wait4(-1, State now 0111020
NULL, 0, NULL)              = 131113
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=131113, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
close(6)                        = 0
read(5, "0111020\0", 8)               = 8
close(5)                        = 0
pipe2([3, 4], 0)                  = 0
pipe2([5, 6], 0)                  = 0
```

```
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f6587b14710) = 131114
close(3)                        = 0
write(4, "0111020\0", 8)              = 8
close(4)                        = 0
wait4(-1, State now 0111320
NULL, 0, NULL)              = 131114
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=131114, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
close(6)                        = 0
read(5, "0111320\0", 8)               = 8
close(5)                        = 0
pipe2([3, 4], 0)                  = 0
pipe2([5, 6], 0)                  = 0
clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f6587b14710) = 131115
close(3)                        = 0
write(4, "0111320\0", 8)              = 8
close(4)                        = 0
wait4(-1, State now 0111326
NULL, 0, NULL)              = 131115
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=131115, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
close(6)                        = 0
read(5, "0111326\0", 8)               = 8
close(5)                        = 0
newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...},
AT_EMPTY_PATH) = 0
write(1, "Result for job_id_1 is 1\n", 25Result for job_id_1 is 1
) = 25
write(1, "Result for job_id_2 is 1\n", 25Result for job_id_2 is 1
) = 25
write(1, "Result for job_id_3 is 1\n", 25Result for job_id_3 is 1
) = 25
write(1, "Result for job_id_4 is 3\n", 25Result for job_id_4 is 3
) = 25
write(1, "Result for job_id_5 is 2\n", 25Result for job_id_5 is 2
) = 25
write(1, "Result for job_id_6 is 6\n", 25Result for job_id_6 is 6
) = 25
exit_group(0)                     = ?
+++ exited with 0 +++
```

Вывод

В результате выполнения курсового проекта мною была изучена работа с конфигурационными файлами типа .ini. Помимо всего прочего, был составлен направленный ациклический график (DAG - Directed Acyclic Graph), который взаимодействует с разными работами (job).