Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

# Лабораторная работа №1 по курсу

# «Операционные системы»

Группа: М8О-213Б-23

Студент: Гига М.Я.

Преподаватель: Бахарев В.Д

Оценка: _____

Дата: 04.10.24

Москва, 2024

# Постановка задачи

**Вариант 6.**

Родительский процесс создает дочерний процесс. Первой строчкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия файла с таким именем на чтение. Стандартный поток ввода дочернего процесса переопределяется открытым файлом. Дочерний процесс читает команды из стандартного потока ввода. Стандартный поток вывода дочернего процесса перенаправляется в pipe1. Родительский процесс читает из pipe1 и прочитанное выводит в свой стандартный поток вывода. Родительский и дочерний процесс должны быть представлены разными программами. В файле записаны команды вида: «число число число<endline>». Дочерний процесс считает их сумму и выводит результат в стандартный поток вывода. Числа имеют тип int. Количество чисел может быть произвольным.

# Общий метод и алгоритм решения

**Использованные системные вызовы:**

- pid_t fork(void); – создает дочерний процесс.
- int pipe(int *fd);  – создает канал.
- int read(int fd, void* buf, size_t count); – считывает count байт из fd в buf.
- int write(int fd, void* buf, size_t count); – записывает count байт из buf в fd.
- int open(const char *pathname, int flags, ...); – открывает файловый дескриптор.
- int close(int fd);  – закрывает файловый дескриптор.
- pid_t waitpid(pid_t pid, int *_Nullable wstatus, int options); – ожидает завершения процесса.

Программа main инициализирует переменную prog_name значением из argw[1] если оно есть, или строкой "./child.out". Выполняется создание канала. Выполняется считывание имени входного файла из stdin. Открывается файловый дескриптор, соответствующий имени файла, выполняется проверка на успешность открытия. Память, выделенная на имя, освобождается. Вызывается fork. В процессе родителе создается буфер, производится последовательное чтения из канала и запись в стандартный вывод. После завершении данной операции, процесс родитель ждет завершения дочернего процесса. В дочернем процессе стандартный вход перенаправлен на открытый файл, а стандартный вывод на канал с использованием dup2. После этого файловый дескриптор и канал закрываются. С использованием execv вызывается процесс с путем prog_name.

Программа child создает буфер для ввода с переменным размером и буфер для вывода с неизменяемым размером. Используется метод read_line для построчного чтения стандартного ввода. Каждая строка разбивается на подстроки, разделенные пробелом, каждая подстрока переводится в int использую atoi. Полученные символы складываются. Результат записывается в буфер вывода методом itoa. Буфер вывода записывается в стандартный вывод.

В обоих программах используется метод read_line. В него передаются файловый дескриптор fd, буфер **buf, и размер буфера *buf_size. Этот метод производит сдвиг памяти в буфере до 0 – конца сейчас записанной в буфер строки. Оставшаяся в буфере информация – это остаток после предыдущих чтений. Длина оставшейся строки измеряется и записывается в count. В цикле do происходит проверка наполнения буфера: если он полный, то производится перевыделение памяти. Происходит чтение из fd в буфер со сдвигом в count и размером buf_size - count. Count увеличивается на количество считанных символов. Цикл завершается, когда в буфере содержится символ переноса строки. После завершения цикла первое вхождение '\n' в буфер заменяется на '\0' – символ конца строки.

# Код программы

**main.c**

```
#include <errno.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <fcntl.h>

#include "misc.h"

int main(int argc, char *argw[]) {
    char *prog_name;
    if (argc >= 2)
        prog_name = argw[1];
    else
        prog_name = "./child.out";

    int pipefd[2];
    if (pipe(pipefd) == -1) {
        print(STDERR_FILENO, "ERROR: failed to create pipe\n");
        exit(-1);
    }

    int buf_size = 64;
    char *fname = malloc(buf_size);
    *fname = 0;
    if (read_line(STDIN_FILENO, &fname, &buf_size) <= 0) {
        print(STDERR_FILENO,
              "ERROR: failed read filename from standart input\n");
        exit(-1);
    }
    int fd = open(fname, O_RDONLY);
    if (fd == -1) {
        print(STDERR_FILENO, "ERROR: failed to open file: \"");
        print(STDERR_FILENO, fname);
        print(STDERR_FILENO, "\"\n");
        print(STDERR_FILENO, strerror(errno));
        print(STDERR_FILENO, "\n");
        exit(-1);
    }
    free(fname);

    pid_t pid = fork();
    if (pid == 0) {
        dup2(fd, STDIN_FILENO);
        close(fd); // close fd, it was duplicated by dup2

        close(pipefd[0]); // close read. we only need to write to pipe
        dup2(pipefd[1], STDOUT_FILENO); // redirect output into pipe
        dup2(pipefd[1], STDERR_FILENO);
        close(pipefd[1]); // close pipe, it was duplicated

        char *argv[] = {prog_name, "", NULL};
        if (execv(prog_name, argv) == -1) {
            print(STDERR_FILENO, "ERROR: failed to launch process \"");
            print(STDERR_FILENO, prog_name);
            print(STDERR_FILENO, "\"\n");
            print(STDERR_FILENO, strerror(errno));
            print(STDERR_FILENO, "\n");
            exit(-1);
        }
    } else if (pid == -1) {
        print(STDERR_FILENO, "ERROR: failed to fork process\n");
```

```
        print(STDERR_FILENO, strerror(errno));
        print(STDERR_FILENO, "\n");
        exit(-1);
    } else {
        char buffer[128];
        close(pipefd[1]); // close write we only need to read from the pipe
        while (1) {
            int n = read(pipefd[0], buffer, sizeof(buffer));
            if (n == 0)
                break;
            write(STDOUT_FILENO, buffer, n);
        }
        waitpid(pid, 0, 0);
    }
    return 0;
}
```

**child.c**

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

#include "misc.h"

int main(void) {
    int buf_size = 2;
    char *input_buffer = malloc(buf_size);
    memset(input_buffer, 0, buf_size);
    char output_buffer[64];

    if (!input_buffer) {
        return -1;
    }
    while (1) {
        int count = read_line(STDIN_FILENO, &input_buffer, &buf_size);
        if (count <= 0)
            break;
        int res = 0;
        char *ptr = input_buffer;
        while (*ptr) {
            int f = atoi(ptr);
            res += f;
            ptr = strchr(ptr, ' ');
            if (!ptr)
                break;
            ptr++;
        }
        int n = itoa(res, output_buffer, 1);
        output_buffer[n++] = '\n';
        write(STDOUT_FILENO, output_buffer, n);
    }
    free(input_buffer);
    return 0;
}
```

**misc.h**

```
#ifndef __MISC_H__
#define __MISC_H__

#include <stddef.h>

int itoa(long n, char *res, int d);
char *strnchr(const char *buf, char c, size_t len);
```

```
int read_line(int fd, char **buf, int *buf_size);
int print(int fd, const char *s);

#endif
```

**misc.c**

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

int itoa(long n, char *res, int d) {
    int neg = 0;
    if (n < 0) {
        neg++;
        n *= -1;
        res[0] = '-';
        res++;
    }
    int i = 0;
    while (n > 0) {
        res[i++] = '0' + (n % 10);
        n /= 10;
    }
    while (i < d) {
        res[i++] = '0';
    }
    for (int j = 0; j < i / 2; j++) {
        char tmp = res[j];
        res[j] = res[i - j - 1];
        res[i - j - 1] = tmp;
    }
    res[i] = 0;
    return i + neg;
}

char *strnchr(const char *buf, char c, size_t len) {
    for (size_t i = 0; i < len; i++) {
        if (buf[i] == c)
            return (char *)(buf + i);
        if (buf[i] == 0)
            return 0;
    }
    return 0;
}

int read_line(int fd, char **buf, int *buf_size) {
    int count = strnchr(*buf, 0, *buf_size) - *buf + 1;
    for (int i = 0; i < *buf_size - count; i++) {
        (*buf)[i] = (*buf)[i + count];
    }
    count = strnchr(*buf, 0, *buf_size) - *buf;
    int read_cur;
    do {
        if (count + 1 >= *buf_size) {
            int new_size = *buf_size * 2;
            char *tmp = realloc(*buf, new_size);
            if (!tmp)
                return count;
            *buf = tmp;
            *buf_size = new_size;
        }
        read_cur = read(fd, *buf + count, *buf_size - count - 1);
        count += read_cur;
        (*buf)[count] = 0;
```

```
    } while (read_cur > 0 && !strnchr(*buf, '\n', *buf_size));
    int line_len = strnchr(*buf, '\n', *buf_size) - *buf;
    if (line_len < 0)
        return count;
    (*buf)[line_len] = 0;
    return line_len + 1;
}

int print(int fd, const char *s) {
    int n = strlen(s);
    return write(fd, s, n);
}
```

# Протокол работы программы

**Тестирование:**

```
$ cat input.txt
1
124 12 -1

2
1
-2317
-23 23 12 -11
2 4

$ ./main.out
a.txt
ERROR: failed to open file: "a.txt"
No such file or directory

$ ./main.out
input.txt
1
135
0
2
1
-2317
1
6
```

**Strace:**

```
$ sudo dtruss -f ./main.out
        PID/THRD  SYSCALL(args)                  = return
 2964/0x867b:  fork()            = 0 0
 2964/0x867b:  munmap(0x100FF4000, 0x8C000)              = 0 0
 2964/0x867b:  munmap(0x101080000, 0x8000)               = 0 0
 2964/0x867b:  munmap(0x101088000, 0x4000)               = 0 0
 2964/0x867b:  munmap(0x10108C000, 0x4000)               = 0 0
 2964/0x867b:  munmap(0x101090000, 0x50000)              = 0 0
 2964/0x867b:  crossarch_trap(0x0, 0x0, 0x0)             = -1 Err#45
 2964/0x867b:  open(".\0", 0x100000, 0x0)                = 3 0
 2964/0x867b:  fcntl(0x3, 0x32, 0x16EE2B228)             = 0 0
 2964/0x867b:  close(0x3)                = 0 0
 2964/0x867b:  fsgetpath(0x16EE2B238, 0x400, 0x16EE2B218)                = 49 0
 2964/0x867b:  fsgetpath(0x16EE2B248, 0x400, 0x16EE2B228)                = 14 0
 2964/0x867b:  csrctl(0x0, 0x16EE2B64C, 0x4)             = -1 Err#1
 2964/0x867b:  __mac_syscall(0x196271ACF, 0x2, 0x16EE2B590)              = 0 0
 2964/0x867b:  csrctl(0x0, 0x16EE2B63C, 0x4)             = -1 Err#1
 2964/0x867b:  __mac_syscall(0x19626E902, 0x5A, 0x16EE2B5D0)             = 0 0
 2964/0x867b:  sysctl([unknown, 3, 0, 0, 0, 0] (2), 0x16EE2AB58, 0x16EE2AB50,
0x196270553, 0xD)                = 0 0
```

```
 2964/0x867b:  sysctl([CTL_KERN, 140, 0, 0, 0, 0] (2), 0x16EE2AC08, 0x16EE2AC00, 0x0,
0x0)              = 0 0
 2964/0x867b:  open("/\0", 0x20100000, 0x0)              = 3 0
 2964/0x867b:  openat(0x3, "System/Cryptexes/OS\0", 0x100000, 0x0)= 4 0
 2964/0x867b:  dup(0x4, 0x0, 0x0)                = 5 0
 2964/0x867b:  fstatat64(0x4, 0x16EE2A6E1, 0x16EE2A650)          = 0 0
 2964/0x867b:  openat(0x4, "System/Library/dyld/\0", 0x100000, 0x0)= 6 0
 2964/0x867b:  fcntl(0x6, 0x32, 0x16EE2A6E0)             = 0 0
 2964/0x867b:  dup(0x6, 0x0, 0x0)                = 7 0
 2964/0x867b:  dup(0x5, 0x0, 0x0)                = 8 0
 2964/0x867b:  close(0x3)                = 0 0
 2964/0x867b:  close(0x5)                = 0 0
 2964/0x867b:  close(0x4)                = 0 0
 2964/0x867b:  close(0x6)                = 0 0
 2964/0x867b:  __mac_syscall(0x196271ACF, 0x2, 0x16EE2B0D0)              = 0 0
 2964/0x867b:  shared_region_check_np(0x16EE2ACF0, 0x0, 0x0)             = 0 0
 2964/0x867b:  fsgetpath(0x16EE2B250, 0x400, 0x16EE2B198)                = 82 0
 2964/0x867b:  fcntl(0x8, 0x32, 0x16EE2B250)             = 0 0
 2964/0x867b:  close(0x8)                = 0 0
 2964/0x867b:  close(0x7)                = 0 0
 2964/0x867b:  getfsstat64(0x0, 0x0, 0x2)                = 10 0
 2964/0x867b:  getfsstat64(0x1013E6AA0, 0x54B0, 0x2)             = 10 0
 2964/0x867b:  getattrlist("/\0", 0x16EE2B180, 0x16EE2B0F0)              = 0 0
 2964/0x867b:
stat64("/System/Volumes/Preboot/Cryptexes/OS/System/Library/dyld/dyld_shared_cache_arm6
4e\0", 0x16EE2B4E0, 0x0)          = 0 0
dtrace: error on enabled probe ID 1690 (ID 845: syscall::stat64:return): invalid
address (0x0) in action #12 at DIF offset 12
 2964/0x867b:  stat64("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/main.out\0",
0x16EE2A990, 0x0)           = 0 0
 2964/0x867b:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/main.out\0", 0x0, 0x0)
= 3 0
 2964/0x867b:  mmap(0x0, 0x88C8, 0x1, 0x40002, 0x3, 0x0)                 = 0x101428000
0
 2964/0x867b:  fcntl(0x3, 0x32, 0x16EE2AAA8)             = 0 0
 2964/0x867b:  close(0x3)                = 0 0
 2964/0x867b:  munmap(0x101428000, 0x88C8)               = 0 0
 2964/0x867b:  stat64("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/main.out\0",
0x16EE2AF00, 0x0)           = 0 0
 2964/0x867b:  stat64("/usr/lib/libSystem.B.dylib\0", 0x16EE29E40, 0x0)= -1 Err#2
 2964/0x867b:
stat64("/System/Volumes/Preboot/Cryptexes/OS/usr/lib/libSystem.B.dylib\0", 0x16EE29DF0,
0x0)             = -1 Err#2
 2964/0x867b:  stat64("/usr/lib/system/libdispatch.dylib\0", 0x16EE27A60, 0x0) = -1
Err#2
 2964/0x867b:
stat64("/System/Volumes/Preboot/Cryptexes/OS/usr/lib/system/libdispatch.dylib\0",
0x16EE27A10, 0x0)              = -1 Err#2
 2964/0x867b:  stat64("/usr/lib/system/libdispatch.dylib\0", 0x16EE27A60, 0x0) = -1
Err#2
 2964/0x867b:  open("/dev/dtracehelper\0", 0x2, 0x0)             = 3 0
 2964/0x867b:  ioctl(0x3, 0x80086804, 0x16EE29A38)               = 0 0
 2964/0x867b:  close(0x3)                = 0 0
 2964/0x867b:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/main.out\0", 0x0, 0x0)
= 3 0
 2964/0x867b:  __mac_syscall(0x196271ACF, 0x2, 0x16EE29140)              = 0 0
 2964/0x867b:  map_with_linking_np(0x16EE28FD0, 0x1, 0x16EE29000)= 0 0
 2964/0x867b:  close(0x3)                = 0 0
 2964/0x867b:  mprotect(0x100FD8000, 0x4000, 0x1)                = 0 0
 2964/0x867b:  shared_region_check_np(0xFFFFFFFFFFFFFFFF, 0x0, 0x0)= 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x1)               = 0 0
 2964/0x867b:  access("/AppleInternal/XBS/.isChrooted\0", 0x0, 0x0)= -1 Err#2
 2964/0x867b:  bsdthread_register(0x196574D2C, 0x196574D20, 0x4000)= 1073746399 0
 2964/0x867b:  getpid(0x0, 0x0, 0x0)             = 2964 0
 2964/0x867b:  shm_open(0x19640FF51, 0x0, 0x69636573)            = 3 0
```

```
 2964/0x867b:  fstat64(0x3, 0x16EE29DD0, 0x0)            = 0 0
 2964/0x867b:  mmap(0x0, 0x4000, 0x1, 0x40001, 0x3, 0x0)                 = 0x101430000
0
 2964/0x867b:  close(0x3)                = 0 0
 2964/0x867b:  ioctl(0x2, 0x4004667A, 0x16EE29E7C)               = 0 0
 2964/0x867b:  mprotect(0x10143C000, 0x4000, 0x0)                = 0 0
 2964/0x867b:  mprotect(0x101448000, 0x4000, 0x0)                = 0 0
 2964/0x867b:  mprotect(0x10144C000, 0x4000, 0x0)                = 0 0
 2964/0x867b:  mprotect(0x101458000, 0x4000, 0x0)                = 0 0
 2964/0x867b:  mprotect(0x10145C000, 0x4000, 0x0)                = 0 0
 2964/0x867b:  mprotect(0x101468000, 0x4000, 0x0)                = 0 0
 2964/0x867b:  mprotect(0x101434000, 0xA0, 0x1)          = 0 0
 2964/0x867b:  mprotect(0x101434000, 0xA0, 0x3)          = 0 0
 2964/0x867b:  mprotect(0x101434000, 0xA0, 0x1)          = 0 0
 2964/0x867b:  mprotect(0x10146C000, 0x4000, 0x1)                = 0 0
 2964/0x867b:  mprotect(0x101470000, 0xA0, 0x1)          = 0 0
 2964/0x867b:  mprotect(0x101470000, 0xA0, 0x3)          = 0 0
 2964/0x867b:  mprotect(0x101470000, 0xA0, 0x1)          = 0 0
 2964/0x867b:  mprotect(0x101434000, 0xA0, 0x3)          = 0 0
 2964/0x867b:  mprotect(0x101434000, 0xA0, 0x1)          = 0 0
 2964/0x867b:  mprotect(0x10146C000, 0x4000, 0x3)                = 0 0
 2964/0x867b:  mprotect(0x10146C000, 0x4000, 0x1)                = 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x3)               = 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x1)               = 0 0
 2964/0x867b:  objc_bp_assist_cfg_np(0x1961A1000, 0x80000018001C1048, 0x0)= -1 Err#5
 2964/0x867b:  issetugid(0x0, 0x0, 0x0)          = 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x3)               = 0 0
 2964/0x867b:  getentropy(0x16EE294E8, 0x20, 0x0)                = 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x1)               = 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x3)               = 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x1)               = 0 0
 2964/0x867b:  getattrlist("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/main.out\0",
0x16EE29D60, 0x16EE29D78)              = 0 0
 2964/0x867b:  access("/Users/maxgiga/dev/mai/MAI_OS/lab01/src\0", 0x4, 0x0)= 0 0
 2964/0x867b:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src\0", 0x0, 0x0)= 3 0
 2964/0x867b:  fstat64(0x3, 0x1296044E0, 0x0)            = 0 0
 2964/0x867b:  csrctl(0x0, 0x16EE29F8C, 0x4)             = -1 Err#1
 2964/0x867b:  fgetattrlist(0x3, 0x16EE2A030, 0x16EE29FB0)               = 0 0
 2964/0x867b:  __mac_syscall(0x1A184B652, 0x2, 0x16EE29FB0)              = 0 0
 2964/0x867b:  fcntl(0x3, 0x32, 0x16EE29C48)             = 0 0
 2964/0x867b:  close(0x3)                = 0 0
 2964/0x867b:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/Info.plist\0", 0x0, 0x0)
= -1 Err#2
 2964/0x867b:  proc_info(0x2, 0xB94, 0xD)                = 64 0
 2964/0x867b:  csops_audittoken(0xB94, 0x10, 0x16EE29FD0)                = 0 0
 2964/0x867b:  sysctl([unknown, 3, 0, 0, 0, 0] (2), 0x16EE2A328, 0x16EE2A320,
0x1999A6D3D, 0x15)               = 0 0
 2964/0x867b:  sysctl([CTL_KERN, 138, 0, 0, 0, 0] (2), 0x16EE2A3B8, 0x16EE2A3B0, 0x0,
0x0)             = 0 0
 2964/0x867b:  csops(0xB94, 0x0, 0x16EE2A45C)            = 0 0
 2964/0x867b:  mprotect(0x1013E4000, 0x40000, 0x3)               = 0 0
 2964/0x867b:  pipe(0x0, 0x0, 0x0)               = 3 0
input.txt
1
135
0
2
1
-2317
1
6
 2964/0x867b:  read(0x0, "input.txt\n\0", 0x3F)          = 10 0
 2964/0x867b:  open("input.txt\0", 0x0, 0x0)             = 5 0
 2964/0x867b:  fork()            = 2965 0
 2965/0x86ba:  fork()            = 0 0
```

```
 2965/0x86ba:  thread_selfid(0x0, 0x0, 0x0)              = 34490 0
 2965/0x86ba:  bsdthread_register(0x196574D2C, 0x196574D20, 0x4000)= -1 Err#22
 2965/0x86ba:  mprotect(0x101470000, 0xA0, 0x3)          = 0 0
 2965/0x86ba:  mprotect(0x101470000, 0xA0, 0x1)          = 0 0
 2964/0x867b:  close(0x4)                = 0 0
 2965/0x86ba:  dup2(0x5, 0x0, 0x0)                = 0 0
 2965/0x86ba:  close(0x5)                = 0 0
 2965/0x86ba:  close(0x3)                = 0 0
 2965/0x86ba:  dup2(0x4, 0x1, 0x0)                = 1 0
 2965/0x86ba:  dup2(0x4, 0x2, 0x0)                = 2 0
 2965/0x86ba:  close(0x4)                = 0 0
dtrace: error on enabled probe ID 1688 (ID 287: syscall::execve:return): invalid
address (0x100fd7ed8) in action #12 at DIF offset 12
 2965/0x86bb:  fork()            = 0 0
 2965/0x86bb:  mprotect(0x100558000, 0x8000, 0x1)                = 0 0
 2965/0x86bb:  thread_selfid(0x0, 0x0, 0x0)              = 34491 0
 2965/0x86bb:  crossarch_trap(0x0, 0x0, 0x0)             = -1 Err#45
 2965/0x86bb:  shared_region_check_np(0x16FCAF760, 0x0, 0x0)             = 0 0
 2965/0x86bb:  thread_selfid(0x0, 0x0, 0x0)              = 34491 0
 2965/0x86bb:  getpid(0x0, 0x0, 0x0)             = 2965 0
 2965/0x86bb:  proc_info(0xF, 0xB95, 0x0)                = 0 0
 2965/0x86bb:  munmap(0x1004CC000, 0x8C000)              = 0 0
 2965/0x86bb:  munmap(0x100558000, 0x8000)               = 0 0
 2965/0x86bb:  munmap(0x100560000, 0x4000)               = 0 0
 2965/0x86bb:  munmap(0x100564000, 0x4000)               = 0 0
 2965/0x86bb:  munmap(0x100568000, 0x50000)              = 0 0
 2965/0x86bb:  crossarch_trap(0x0, 0x0, 0x0)             = -1 Err#45
 2965/0x86bb:  open(".\0", 0x100000, 0x0)                = 3 0
 2965/0x86bb:  fcntl(0x3, 0x32, 0x16FCAF228)             = 0 0
 2965/0x86bb:  close(0x3)                = 0 0
 2965/0x86bb:  fsgetpath(0x16FCAF238, 0x400, 0x16FCAF218)                = 50 0
 2965/0x86bb:  fsgetpath(0x16FCAF248, 0x400, 0x16FCAF228)                = 14 0
 2965/0x86bb:  csrctl(0x0, 0x16FCAF64C, 0x4)             = -1 Err#1
 2965/0x86bb:  __mac_syscall(0x196271ACF, 0x2, 0x16FCAF590)              = 0 0
 2965/0x86bb:  csrctl(0x0, 0x16FCAF63C, 0x4)             = -1 Err#1
 2965/0x86bb:  __mac_syscall(0x19626E902, 0x5A, 0x16FCAF5D0)             = 0 0
 2965/0x86bb:  sysctl([unknown, 3, 0, 0, 0, 0] (2), 0x16FCAEB58, 0x16FCAEB50,
0x196270553, 0xD)                = 0 0
 2965/0x86bb:  sysctl([CTL_KERN, 140, 0, 0, 0, 0] (2), 0x16FCAEC08, 0x16FCAEC00, 0x0,
0x0)             = 0 0
 2965/0x86bb:  open("/\0", 0x20100000, 0x0)              = 3 0
 2965/0x86bb:  openat(0x3, "System/Cryptexes/OS\0", 0x100000, 0x0)= 4 0
 2965/0x86bb:  dup(0x4, 0x0, 0x0)                = 5 0
 2965/0x86bb:  fstatat64(0x4, 0x16FCAE6E1, 0x16FCAE650)          = 0 0
 2965/0x86bb:  openat(0x4, "System/Library/dyld/\0", 0x100000, 0x0)= 6 0
 2965/0x86bb:  fcntl(0x6, 0x32, 0x16FCAE6E0)             = 0 0
 2965/0x86bb:  dup(0x6, 0x0, 0x0)                = 7 0
 2965/0x86bb:  dup(0x5, 0x0, 0x0)                = 8 0
 2965/0x86bb:  close(0x3)                = 0 0
 2965/0x86bb:  close(0x5)                = 0 0
 2965/0x86bb:  close(0x4)                = 0 0
 2965/0x86bb:  close(0x6)                = 0 0
 2965/0x86bb:  __mac_syscall(0x196271ACF, 0x2, 0x16FCAF0D0)              = 0 0
 2965/0x86bb:  shared_region_check_np(0x16FCAECF0, 0x0, 0x0)             = 0 0
 2965/0x86bb:  fsgetpath(0x16FCAF250, 0x400, 0x16FCAF198)                = 82 0
 2965/0x86bb:  fcntl(0x8, 0x32, 0x16FCAF250)             = 0 0
 2965/0x86bb:  close(0x8)                = 0 0
 2965/0x86bb:  close(0x7)                = 0 0
 2965/0x86bb:  getfsstat64(0x0, 0x0, 0x2)                = 10 0
 2965/0x86bb:  getfsstat64(0x100562AA0, 0x54B0, 0x2)             = 10 0
 2965/0x86bb:  getattrlist("/\0", 0x16FCAF180, 0x16FCAF0F0)              = 0 0
 2965/0x86bb:
stat64("/System/Volumes/Preboot/Cryptexes/OS/System/Library/dyld/dyld_shared_cache_arm6
4e\0", 0x16FCAF4E0, 0x0)          = 0 0
```

```
dtrace: error on enabled probe ID 1690 (ID 845: syscall::stat64:return): invalid
address (0x0) in action #12 at DIF offset 12
 2965/0x86bb:  stat64("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/child.out\0",
0x16FCAE990, 0x0)          = 0 0
 2965/0x86bb:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/child.out\0", 0x0, 0x0)
= 3 0
 2965/0x86bb:  mmap(0x0, 0x87A8, 0x1, 0x40002, 0x3, 0x0)                 = 0x1005A4000
0
 2965/0x86bb:  fcntl(0x3, 0x32, 0x16FCAEAA8)             = 0 0
 2965/0x86bb:  close(0x3)                = 0 0
 2965/0x86bb:  munmap(0x1005A4000, 0x87A8)               = 0 0
 2965/0x86bb:  stat64("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/child.out\0",
0x16FCAEF00, 0x0)          = 0 0
 2965/0x86bb:  stat64("/usr/lib/libSystem.B.dylib\0", 0x16FCADE40, 0x0)= -1 Err#2
 2965/0x86bb:
stat64("/System/Volumes/Preboot/Cryptexes/OS/usr/lib/libSystem.B.dylib\0", 0x16FCADDF0,
0x0)             = -1 Err#2
 2965/0x86bb:  stat64("/usr/lib/system/libdispatch.dylib\0", 0x16FCABA60, 0x0) = -1
Err#2
 2965/0x86bb:
stat64("/System/Volumes/Preboot/Cryptexes/OS/usr/lib/system/libdispatch.dylib\0",
0x16FCABA10, 0x0)              = -1 Err#2
 2965/0x86bb:  stat64("/usr/lib/system/libdispatch.dylib\0", 0x16FCABA60, 0x0) = -1
Err#2
 2965/0x86bb:  open("/dev/dtracehelper\0", 0x2, 0x0)             = 3 0
 2965/0x86bb:  ioctl(0x3, 0x80086804, 0x16FCADA38)               = 0 0
 2965/0x86bb:  close(0x3)                = 0 0
 2965/0x86bb:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/child.out\0", 0x0, 0x0)
= 3 0
 2965/0x86bb:  __mac_syscall(0x196271ACF, 0x2, 0x16FCAD140)              = 0 0
 2965/0x86bb:  map_with_linking_np(0x16FCAD010, 0x1, 0x16FCAD040)= 0 0
 2965/0x86bb:  close(0x3)                = 0 0
 2965/0x86bb:  mprotect(0x100154000, 0x4000, 0x1)                = 0 0
 2965/0x86bb:  shared_region_check_np(0xFFFFFFFFFFFFFFFF, 0x0, 0x0)= 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x1)               = 0 0
 2965/0x86bb:  access("/AppleInternal/XBS/.isChrooted\0", 0x0, 0x0)= -1 Err#2
 2965/0x86bb:  bsdthread_register(0x196574D2C, 0x196574D20, 0x4000)= 1073746399 0
 2965/0x86bb:  getpid(0x0, 0x0, 0x0)             = 2965 0
 2965/0x86bb:  shm_open(0x19640FF51, 0x0, 0x0)           = 3 0
 2965/0x86bb:  fstat64(0x3, 0x16FCADDD0, 0x0)            = 0 0
 2965/0x86bb:  mmap(0x0, 0x4000, 0x1, 0x40001, 0x3, 0x0)                 = 0x1005AC000
0
 2965/0x86bb:  close(0x3)                = 0 0
 2965/0x86bb:  ioctl(0x2, 0x4004667A, 0x16FCADE7C)               = -1 Err#25
 2965/0x86bb:  ioctl(0x2, 0x40487413, 0x16FCADE80)               = -1 Err#25
 2965/0x86bb:  mprotect(0x1005B8000, 0x4000, 0x0)                = 0 0
 2965/0x86bb:  mprotect(0x1005C4000, 0x4000, 0x0)                = 0 0
 2965/0x86bb:  mprotect(0x1005C8000, 0x4000, 0x0)                = 0 0
 2965/0x86bb:  mprotect(0x1005D4000, 0x4000, 0x0)                = 0 0
 2965/0x86bb:  mprotect(0x1005D8000, 0x4000, 0x0)                = 0 0
 2965/0x86bb:  mprotect(0x1005E4000, 0x4000, 0x0)                = 0 0
 2965/0x86bb:  mprotect(0x1005B0000, 0xA0, 0x1)          = 0 0
 2965/0x86bb:  mprotect(0x1005B0000, 0xA0, 0x3)          = 0 0
 2965/0x86bb:  mprotect(0x1005B0000, 0xA0, 0x1)          = 0 0
 2965/0x86bb:  mprotect(0x1005E8000, 0x4000, 0x1)                = 0 0
 2965/0x86bb:  mprotect(0x1005EC000, 0xA0, 0x1)          = 0 0
 2965/0x86bb:  mprotect(0x1005EC000, 0xA0, 0x3)          = 0 0
 2965/0x86bb:  mprotect(0x1005EC000, 0xA0, 0x1)          = 0 0
 2965/0x86bb:  mprotect(0x1005B0000, 0xA0, 0x3)          = 0 0
 2965/0x86bb:  mprotect(0x1005B0000, 0xA0, 0x1)          = 0 0
 2965/0x86bb:  mprotect(0x1005E8000, 0x4000, 0x3)                = 0 0
 2965/0x86bb:  mprotect(0x1005E8000, 0x4000, 0x1)                = 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x3)               = 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x1)               = 0 0
 2965/0x86bb:  objc_bp_assist_cfg_np(0x1961A1000, 0x80000018001C1048, 0x0)= -1 Err#5
```

```
 2965/0x86bb:  issetugid(0x0, 0x0, 0x0)          = 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x3)               = 0 0
 2965/0x86bb:  getentropy(0x16FCAD4E8, 0x20, 0x0)                = 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x1)               = 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x3)               = 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x1)               = 0 0
 2965/0x86bb:  getattrlist("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/child.out\0",
0x16FCADD60, 0x16FCADD78)             = 0 0
 2965/0x86bb:  access("/Users/maxgiga/dev/mai/MAI_OS/lab01/src\0", 0x4, 0x0)= 0 0
 2965/0x86bb:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src\0", 0x0, 0x0)= 3 0
 2965/0x86bb:  fstat64(0x3, 0x13F6044E0, 0x0)            = 0 0
 2965/0x86bb:  csrctl(0x0, 0x16FCADF8C, 0x4)             = -1 Err#1
 2965/0x86bb:  fgetattrlist(0x3, 0x16FCAE030, 0x16FCADFB0)               = 0 0
 2965/0x86bb:  __mac_syscall(0x1A184B652, 0x2, 0x16FCADFB0)              = 0 0
 2965/0x86bb:  fcntl(0x3, 0x32, 0x16FCADC48)             = 0 0
 2965/0x86bb:  close(0x3)                = 0 0
 2965/0x86bb:  open("/Users/maxgiga/dev/mai/MAI_OS/lab01/src/Info.plist\0", 0x0, 0x0)
= -1 Err#2
 2965/0x86bb:  proc_info(0x2, 0xB95, 0xD)                = 64 0
 2965/0x86bb:  csops_audittoken(0xB95, 0x10, 0x16FCADFD0)                = 0 0
 2965/0x86bb:  sysctl([unknown, 3, 0, 0, 0, 0] (2), 0x16FCAE328, 0x16FCAE320,
0x1999A6D3D, 0x15)                = 0 0
 2965/0x86bb:  sysctl([CTL_KERN, 138, 0, 0, 0, 0] (2), 0x16FCAE3B8, 0x16FCAE3B0, 0x0,
0x0)             = 0 0
 2965/0x86bb:  csops(0xB95, 0x0, 0x16FCAE45C)            = 0 0
 2965/0x86bb:  mprotect(0x100560000, 0x40000, 0x3)               = 0 0
 2965/0x86bb:  read(0x0, "1\0", 0x1)             = 1 0
 2965/0x86bb:  read(0x0, "\n1\0", 0x2)           = 2 0
 2965/0x86bb:  write(0x1, "1\n\0", 0x2)          = 2 0
 2965/0x86bb:  read(0x0, "24\0", 0x2)            = 2 0
 2965/0x86bb:  read(0x0, " 12 \0", 0x4)          = 4 0
 2964/0x867b:  read(0x3, "1\n\0", 0x80)          = 2 0
 2965/0x86bb:  read(0x0, "-1\n\n2\n1\n\0", 0x8)          = 8 0
 2965/0x86bb:  write(0x1, "135\n\0", 0x4)                 = 4 0
 2965/0x86bb:  read(0x0, "-2317\n-23 \0", 0xA)           = 10 0
 2964/0x867b:  write(0x1, "1\n\0", 0x2)          = 2 0
 2965/0x86bb:  write(0x1, "0\n\0", 0x2)          = 2 0
 2964/0x867b:  read(0x3, "135\n0\n\0", 0x80)             = 6 0
 2965/0x86bb:  read(0x0, "2\0", 0x1)             = 1 0
 2964/0x867b:  write(0x1, "135\n0\n\0", 0x6)              = 6 0
 2965/0x86bb:  write(0x1, "2\n\0", 0x2)          = 2 0
 2964/0x867b:  read(0x3, "2\n\0", 0x80)          = 2 0
 2965/0x86bb:  read(0x0, "3 \0", 0x2)            = 2 0
 2965/0x86bb:  write(0x1, "1\n\0", 0x2)          = 2 0
 2965/0x86bb:  read(0x0, "12\0", 0x2)            = 2 0
 2965/0x86bb:  write(0x1, "-2317\n\0", 0x6)              = 6 0
 2965/0x86bb:  read(0x0, " -11\n2\0", 0x6)               = 6 0
 2965/0x86bb:  write(0x1, "1\n\0", 0x2)          = 2 0
 2965/0x86bb:  read(0x0, " 4\n\0", 0xE)          = 3 0
 2965/0x86bb:  write(0x1, "6\n\0", 0x2)          = 2 0
 2964/0x867b:  write(0x1, "2\n\0", 0x2)          = 2 0
 2965/0x86bb:  read(0x0, "\0", 0xF)              = 0 0
 2964/0x867b:  read(0x3, "1\n-2317\n1\n6\n\0", 0x80)             = 12 0
 2964/0x867b:  write(0x1, "1\n-2317\n1\n6\n\0", 0xC)              = 12 0
 2964/0x867b:  read(0x3, "\0", 0x80)             = 0 0
 2964/0x867b:  wait4(0xB95, 0x0, 0x0)            = 2965 0
```

Красным обозначены системные вызовы, произведенные родительским процессом, зеленым – системные вызовы дочернего процесса, синим – системные вызовы дочернего процесса вызванного через execv.

# Вывод

В ходе выполнения лабораторной работы была составлена и отлажена программа на языке C, осуществляющая вызов дочернего процесса, переопределения стандартных ввода вывода для него и взаимодействие между дочерним и родительским процессами в операционной системе macos. Была также написана программа, осуществляющая ввод из стандартного ввода, обработку и вывод информации в стандартный вывод без использования stdio.h из стандартной библиотеки языка c.