Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

Лабораторная работа №1. Управление процессами в ОС GNU/Linux
тема

Преподаватель _____ А.С. Кузнецов
подпись, дата инициалы, фамилия

Студент КИ18-17/1б 031830504 _____ Е.В. Железкин
номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2019

СОДЕРЖАНИЕ

Содержание 2

1 Цель работы 3

2 Задача работы 3

3 Ход работы 3

4 Вывод 10

1 Цель работы

Изучение особенностей программной реализации многозадачных приложений в ОС GNU/Linux.

2 Задача работы

Требуется: разработать программу в виде Linux-приложения, которая представляет собой родительский процесс. Результат выполнения выводится на терминал/консоль. Программа должна быть устойчива к некорректному пользовательскому вводу. В следующих вариантах заданий оговаривается только функционал программы, представляющей собой дочерний процесс.

Вариант 6. Программа принимает от пользователя два целых числа A и B, где A – сдвигаемое число, B – величина логического, арифметического или циклического сдвига. После этого осуществляется логический, арифметический или циклический сдвиг числа A на величину B, и три результата выводятся на экран.

3 Ход работы

1) В среде разработки CLion были созданы родительский и дочерний проекты – SP_1 и SP_1_sub соответственно. Проекту SP_1_sub добавлена и подключена директория cunit-master, содержащая Фреймворк CUnit. Далее файл main.c проекта SP_1 заполняется кодом, который вызывает дочерний процесс и ожидает его завершения, а файл main.c проекта SP_1_sub кодом, который запускает юнит – тесты для функций логического, арифметического и циклического сдвига и продолжает работу с пользователем в интерактивном режиме. Так же в проект SP_1_sub были добавлены заголовочные файлы и файлы с кодом для функций и тестов.

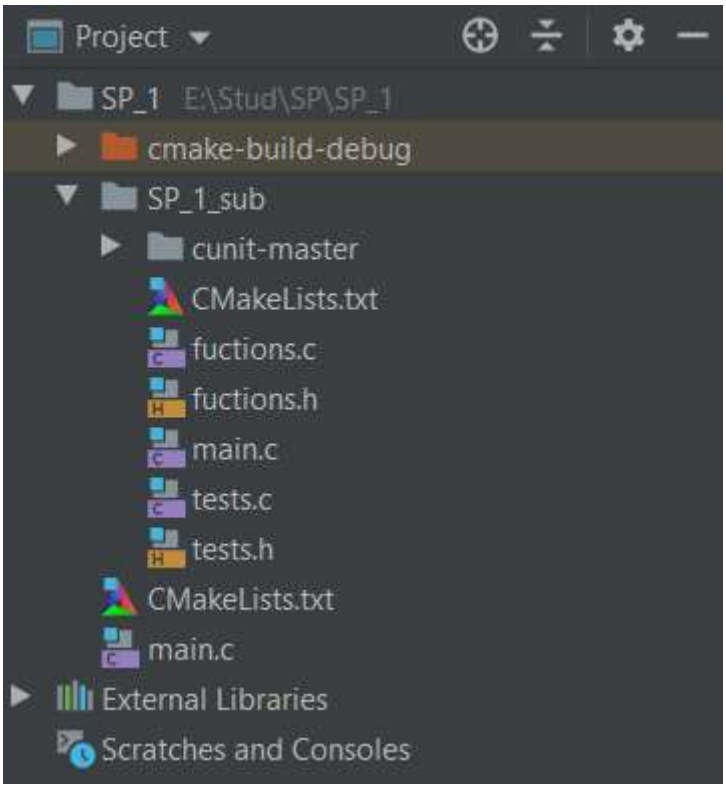

Рисунок 1 – дерево проектов SP_1 и SP_1_sub

Листинг 1 – содержание файла main.c проекта SP_1:

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

int spawn(char* program, char** argList)
{
    pid_t childPid;
    childPid = fork();
    if (0 != childPid)
        return childPid;
    else
    {
        execvp(program, argList);
        abort();
    }
}

int main(int argc, char** argv)
{
    int childStatus;

    spawn("SP_1_sub/SP_1_sub", NULL);

    wait(&childStatus);
    if(WIFEXITED(childStatus))
        return childStatus;
```

```
    else
        return 0;

}
```

Листинг 2 – содержание файла main.c проекта SP_1_sub

```
/*! \file
 *  \brief   Главный файл дочерней программы.
 */
 #include "CUnit/Basic.h"
#include "tests.h"
#include "fuctions.h"
#define MAX_LENGTH 50

int main (int argc, char** argv)
{

    CU_pSuite suite;
    CU_initialize_registry();
    suite = CU_add_suite("main_suite", NULL, NULL);
    CU_ADD_TEST(suite, logicalShiftTest);
    CU_ADD_TEST(suite, ariShiftTest);
    CU_ADD_TEST(suite, cycleShiftTest);
    CU_basic_run_tests();

    CU_cleanup_registry();

    int A = 0;
    int B = 0;

    char str1[MAX_LENGTH];
    char str2[MAX_LENGTH];

    scanf("%s", str1);
    A = checker(str1);
    if(A == 0)
    {
        printf("Введены некорректные данные!");
        return 0;
    }

    scanf("%s", str2);
    B = checker(str2);
    if(B == 0) {
        printf("Введены некорректные данные!");
        return 0;
    }

    if(B >= 0)
        printf("Результат логического сдвига: %d\n", A >> B);
    else
        printf("Результат логического сдвига: %d\n", A << (B * -1));

    printf("Результат арифметического сдвига: %d\n", ariShift(A, B));

    printf("Результат циклического сдвига: %d\n", cycleShift(A, B));

    return CU_get_error();
}
```

Листинг 3 – содержание файла functions.h проекта SP_1_sub

```
/*! \file
 *  \brief   Файл объявлениями с функций.
 */
 #ifndef SP_1_FUCTIONS_H
#define SP_1_FUCTIONS_H

/*! \brief Проверяет введённые данные.
 *
 *  \details Если получает строку, содержащую целое число, то возвращает его, а иначе
возвращает 0.
 *
 *  \param tempForCheck Строка с потенциальным числом.
 *  \return возвращает полученное число, преобразованное к формату int.
 */
int checker(char* tempForCheck);

/*! \brief Выполняет арифметический сдвиг.
 *
 *  \details Выполняет арифметический сдвиг числа A на значение B.
 *
 *  \param A Сдвигаемое число.
 *  \param B шаг сдвига.
 *  \return Возвращает сдвинутое по арифметическому принципу число.
 */
int ariShift(int tempA, int tempB);

/*! \brief Выполняет циклический сдвиг.
 *
 *  \details Выполняет циклический сдвиг числа A на значение B.
 *
 *  \param A Сдвигаемое число.
 *  \param B шаг сдвига.
 *  \return Возвращает сдвинутое по циклическому принципу число.
 */
int cycleShift(int tempA, int tempB);

#endif //SP_1_FUCTIONS_H
```

Листинг 4 – содержание файла functions.c проекта SP_1_sub

```
/*! \file
 *  \brief   Файл с функциями.
 */
#define bitShift 2

 int checker(char* tempForCheck)
{
    int number = 0;
    int flag = 0;

    for(int i = 0; i < strlen(tempForCheck); i++)
    {
        if(i == 0 && tempForCheck[0] == '-') {
            flag++;
            continue;
        }
```

```
        if(tempForCheck[i] == '0' || tempForCheck[i] == '1' || tempForCheck[i] == '2'
|| tempForCheck[i] == '3' ||
           tempForCheck[i] == '4' || tempForCheck[i] == '5' || tempForCheck[i] == '6'
||
           tempForCheck[i] == '7' || tempForCheck[i] == '8' || tempForCheck[i] == '9')
number = number * 10 +

(tempForCheck[i] - 48);
        else
            return 0;
    }

    if(flag > 0) number *= -1;

    return number;
}

int ariShift(int tempA, int tempB)
{
    int temp = 1;

    if(tempB >= 0) {
        while (1) {
            if (temp <= tempA) temp *= bitShift;
            else
                break;
        }

        temp /= bitShift;

        return (tempA >> tempB) + temp;
    } else
    {
        tempB *= -1;
        return tempA << tempB;
    }
}

int cycleShift(int tempA, int tempB)
{
    int temp = 1;

    if(tempB >= 0) {

        while (1) {
            if (temp <= tempA) temp *= bitShift;
            else
                break;
        }

        temp /= bitShift;

        if (tempA % 2 == 0)
            return tempA >> tempB;
        else
            return (tempA >> tempB) + temp;
    } else
    {
        tempB *= -1;
        return (tempA << tempB) + 1;
```

```
    }
}
```

Листинг 5 – содержание файла tests.h проекта SP_1_sub

```
/*! \file
 *  \brief   Файл объявлениями с тестовых кейсов.
 */
 #ifndef SP_1_TESTS_H
#define SP_1_TESTS_H

/*! \brief Содержит 3 теста для функции логического сдвига.
 */
void logicalShiftTest(void);

/*! \brief Содержит 3 теста для функции арифметического сдвига.
 */
void ariShiftTest(void);

/*! \brief Содержит 3 теста для функции циклического сдвига.
 */
void cycleShiftTest(void);

#endif //SP_1_TESTS_H
```

Листинг 6 – содержание файла tests.c проекта SP_1_sub

```
/*! \file
 *  \brief   Файл с тестами.
 */
 #include <CUnit/CUnit.h>
#include "fuctions.h"

void logicalShiftTest(void)
{
    CU_ASSERT_EQUAL(33 >> 3, 4);
    CU_ASSERT_EQUAL(322 >> 5, 10);
    CU_ASSERT_EQUAL(1024 >> 10, 1);
    CU_ASSERT_EQUAL(8 << 1, 16);
}

void ariShiftTest(void)
{
    CU_ASSERT_EQUAL(ariShift(33, 3), 36);
    CU_ASSERT_EQUAL(ariShift(322, 5), 266);
    CU_ASSERT_EQUAL(ariShift(1024, 10), 1025);
    CU_ASSERT_EQUAL(ariShift(18, -5), 18 << 5);
}

void cycleShiftTest(void)
{
    CU_ASSERT_EQUAL(cycleShift(33, 3), 36);
    CU_ASSERT_EQUAL(cycleShift(322, 5), 10);
    CU_ASSERT_EQUAL(cycleShift(1024, 10), 1);
    CU_ASSERT_EQUAL(cycleShift(512, -3), 4097);
}
```

2) Запуск юнит – тестов:

/mnt/e/Stud/SP/SP_1/cmake-build-debug/SP_1_sub/SP_1_sub

CUnit - A unit testing framework for C - Version 3.1.0-cunity
http://cunit.sourceforge.net/

Run Summary	-	Run	Failed	Inactive	Skipped
Suites	:	1	0	0	0
Asserts	:	12	0	n/a	n/a
Tests	:	3	0	0	0

Elapsed Time: 0.000(s)

3) Примеры работы с консолью:

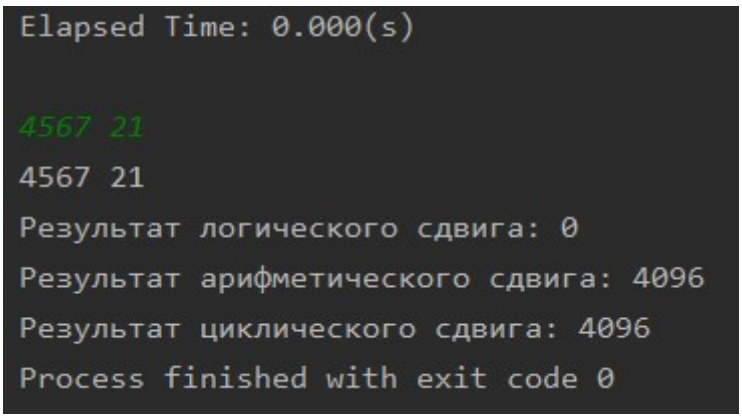

Рисунок 2 – результат работы при вводе «4567 21»

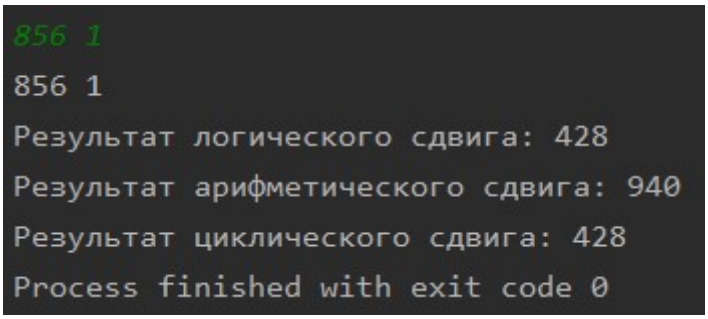

Рисунок 3 – результат работы при вводе «856 1»

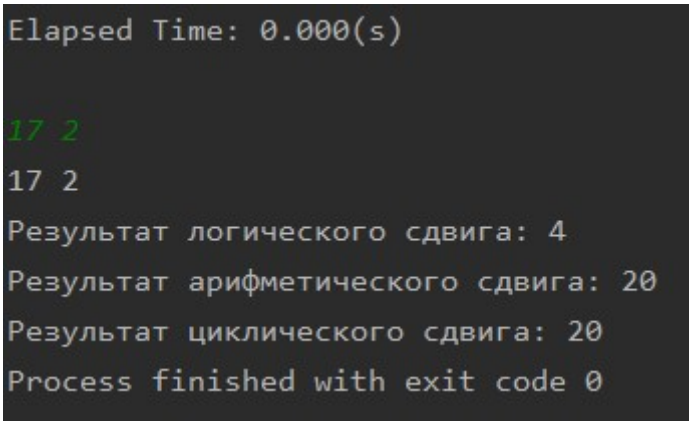

Рисунок 4 - результат работы при вводе «17 2»

4 Вывод

В ходе данной лабораторной работы были изучены азы взаимодействия с дочерним процессом.