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 КИ19-17/1б 031939175 _____ А. Д. Непомнящий
номер группы, зачетной книжки — подпись, дата — инициалы, фамилия

Красноярск 2021

1 Цель работы

Цель состоит в изучении особенностей программной реализации многозадачных приложений в ОС GNU/Linux.

2 Задачи

Выполнение работы сводится к следующим задачам.

1. Ознакомиться с краткими теоретическими сведениями по управлению процессами в ОС GNU/Linux.

2. Используя изученные механизмы, разработать и отладить программный код для родительского и дочернего процессов. Предусматривается запуск дочернего процесса родительским с обязательным использованием концепции fork-and-exec.

3. Произвести разработку юнит-тестов основных функциональных блоков код дочернего процесса. Рекомендуется использование фреймворка CUnit.

4. Написать настоящий отчет и представить его к защите с исходными текстами программы. Исходные тексты программ должны содержать комментарии в стиле системы doxygen и включать юнит-тесты основных функционпльных блоков программного кода.

Вариант 14. Программа принимает от пользователя беззнаковое целое десятичное число N – основание системы счисления ($N > 1$ и $N \leq 20$) и последовательности цифр в соответствии с заданной системой счисления. Затем программа выводит число на экран, переводит его в десятичную систему, выводит на экран, осуществляет его реверс, выводит на экран значение измененной последовательности, переводит его в десятичную систему и выводит на экран.

3 Исходные тексты программ

Листинг 1 – Код в файле ParentProgram/main.c

```
/*! \file   main.c
 *  \brief  Main file of the child program which contains main function
 */

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


/*! \brief Spawns a child process
 *
 *  \details Spawn a child process running a new program.
 *
 *  \param program    The name of the program to run.
 *  \param argList    A NULL-terminated list of character strings to be
 *                    passed as the program's argument list.
 *  \return           PID of the spawned process.
 */
int spawn(char* program, char** argList)
{
    pid_t childPid;
    childPid = fork();
    if (childPid == 0)
    {
        execl(program, NULL, NULL);
        fprintf(stderr, "an error occurred in exec\n");
        abort();
    }
    else
    {
        printf("This is a parent process");
    }
    return 0;
}

/*! \brief Main function
 *
 *  \details Main function. Execs Child program by relative path
 *  "../ChildProgram/ChildProgram". If arguments
 *
 *  \return Integer 0 upon successful exit.
 */
int main()
{
    int childStatus;
    char* child = "../ChildProgram/ChildProgram";
    char* argList[] = {NULL};
    spawn(child, argList);
    wait(&childStatus);
    if (WIFEXITED(childStatus))
    {
        printf("The child process exited normally with code %d.\n",
               WEXITSTATUS(childStatus)
```

Окончание листинга 1

```
        );
    }
    else
    {
        printf("The child process exited abnormally.\n");
    }
    return 0;
}
```

Листинг 2 – Код в файле ChildProgram/child.c

```
/*! \file   child.c
 *  \brief  Main file of the child program which contains the main function
 */

#include <stdio.h>
#include <string.h>
#include "input.h"
#include "task14.h"


/*! \brief Main function
 *
 *  \details Main function. Reads radix and number in appropriate numeral
 *  system. Each input continue until correct value is read.
 *
 *  \param argc Count program arguments.
 *  \param argList Array string which contains args.
 *  \return Integer 0 upon successful exit.
 */
int main(int argc, char** argList)
{
    int radix;
    char number[INPUT_SIZE];

    printf("Enter base of numeral system (2 - 20)\n");
    radix = CheckedInputInt(RadixInputCheck);
    printf("Enter number in chosen system. Use \'A\' - \'J\' as"
           "digits for >10-based systems\n");
    while (true)
    {
        scanf("%s", number);
        if (CheckIntOverflow(number, radix) && CheckRadixMatch(number, radix))
        {
            break;
        }
        printf("Wrong format or too big number!\n");
    }

    char reversedNumber[INPUT_SIZE];
    for (int i = strlen(number) - 1; i >= 0; i--)
    {
        reversedNumber[strlen(number) - (i + 1)] = number[i];
    }
    reversedNumber[strlen(number)] = '\0';

    while (reversedNumber[strlen(reversedNumber) - 1] == '0')
```

Окончание листинга 2

```
    {
        reversedNumber[strlen(reversedNumber) - 1] = '\0';
    }

    printf("Original: %s\n", number);
    printf("To decimal: %d\n", AnyNumeralSystemToDecimal(number, radix));
    printf("Reversed: %s\n", reversedNumber);
    if (CheckIntOverflow(reversedNumber, radix))
    {
        printf("Reversed to decimal: %d\n",
               AnyNumeralSystemToDecimal(reversedNumber, radix));
    }
    else
    {
        printf("Reversed number is too big");
    }

    return 0;
}
```

Листинг 3 – Код в файле ChildProgram/task14.h

```
/*! \file   task14.h
 *  \brief  Header file of functions with numeral systems
 *          essential for task 14
 */

#include <malloc.h>
#include <stdbool.h>
#include <math.h>
#include <string.h>
#include <stdio.h>

#ifndef LAB1_TASK14_H
#define LAB1_TASK14_H


/*! \brief Converts number in any (2-20) numeral system to decimal
 *
 *  \param number number to convert.
 *  \param radix radix of numeral system.
 *  \return Integer conversion result.
 */
int AnyNumeralSystemToDecimal(char* number, int radix);

/*! \brief Checks if number only contains digits, allowed for this numeral
 *  system
 *
 *  \param numberToCheck number to check.
 *  \param radix radix of numeral system.
 *  \return true if number only contains digits, allowed for this numeral
 *  system, false - otherwise.
 */
bool CheckRadixMatch(char* numberToCheck, int radix);

/*! \brief Checks if number is not too big to be written to int after
 *  conversion
```

Окончание листинга 3

```
 *
 *  \param numberToCheck number to check.
 *  \param radix radix of numeral system.
 *  \return true if number is not too big to be written to int after
 *  conversion, false - otherwise
 */
bool CheckIntOverflow(char* numberToCheck, int radix);

/*! \brief Checks if number can be numeral system radix for task 14
 *
 *  \param intToCheck number to check.
 *  \return true if number can be numeral system radix for task 14
 *  false - otherwise
 */
bool RadixInputCheck(int intToCheck);


#endif //LAB1_TASK14_H
```

Листинг 4 – Код в файле ChildProgram/task14.c

```
/*! \file   input.c
 *  \brief  Implements functions of task14.h
 */

#include "task14.h"


/*! \enum
 *  \brief  Essential constants for task 14
 */
enum NumeralSystemsConstants
{
    VEGESIMAL_A = 'A', /** Digit next to 9 */
    MIN_RADIX = 2,     /** Minimal numeral system radix */
    MAX_RADIX = 20     /** Maximal numeral system radix for task */
};

int AnyNumeralSystemToDecimal(char* number, int radix)
{
    int result = 0;
    int multiplier = 1;
    int currentDigit;
    for (int i = strlen(number) - 1; i >= 0; i--)
    {
        if (number[i] >= VEGESIMAL_A)
        {
            currentDigit = 10 + number[i] - VEGESIMAL_A;
        }
        else
        {
            currentDigit = number[i] - '0';
        }
        result += currentDigit * multiplier;
        multiplier *= radix;
    }
    return result;
```

Окончание листинга 4

```
}

bool CheckRadixMatch(char* numberToCheck, int radix)
{
    int currentDigit;
    for (int i = 0; i < strlen(numberToCheck); i++)
    {
        if (numberToCheck[i] >= VEGESIMAL_A)
        {
            currentDigit = 10 + numberToCheck[i] - VEGESIMAL_A;
        }
        else
        {
            currentDigit = numberToCheck[i] - '0';
        }
        if (currentDigit >= radix || currentDigit < 0)
        {
            return false;
        }
    }
    return true;
}

bool CheckIntOverflow(char* numberToCheck, int radix)
{
    return strlen(numberToCheck) <
           (log((double) __INT_MAX__) / log((double) radix) - 1);
}

bool RadixInputCheck(int intToCheck)
{
    if (intToCheck < MIN_RADIX || intToCheck > MAX_RADIX)
    {
        return false;
    }
    return true;
}
```

Листинг 5 – Код в файле ChildProgram/input.h

```
/*! \file   input.h
 *  \brief  Header file of function to read integer with additional check
 */

#include <stdbool.h>

#ifndef LAB1_INPUT_H
#define LAB1_INPUT_H

/*! \enum
 *  \brief  Size of string for input
 */
enum Sizes
{
    INPUT_SIZE = 200
};
```

Окончание листинга 5

```
/*! \brief Reads int
 *
 *  \details Reads int with additional check. Continues reading until
 *  correct value is read.
 *
 *  \param bool* Pointer to the function that checks additional condition.
 *  \return Integer read correct integer.
 */
int CheckedInputInt(bool(* additionalCheck)(int))


#endif //LAB1_INPUT_H
```

Листинг 6 – Код в файле ChildProgram/input.c

```
/*! \file   input.c
 *  \brief  Implements functions of input.h
 */

#include <stdio.h>
#include "input.h"


int CheckedInputInt(bool(* additionalCheck)(int))
{
    int result;
    char inputString[INPUT_SIZE];

    while (true)
    {
        scanf("%s", inputString);
        int flag = sscanf(inputString, "%d", &result);
        if (flag == 0 || flag == EOF)
        {
            printf("Input error!\n");
            continue;
        }
        if (!additionalCheck(result))
        {
            printf("Input error!\n");
            continue;
        }
        return result;
    }
}
```

4 Тестовые примеры работы программы

На рисунках далее приведены примеры работы программы.

```
Enter base of numeral system (2 - 20)
16
Enter number in chosen system. Use 'A' - 'J' asdigits for >10-based systems
B00BA
Original: B00BA
To decimal: 721082
Reversed: AB00B
Reversed to decimal: 700427
This is a parent processThe child process exited normally with code 0.
```

Рисунок 1 – Пример работы программы 1

```
Enter base of numeral system (2 - 20)
10
Enter number in chosen system. Use 'A' - 'J' asdigits for >10-based systems
010240
Original: 010240
To decimal: 10240
Reversed: 04201
Reversed to decimal: 4201
This is a parent processThe child process exited normally with code 0.
```

Рисунок 2 – Пример работы программы 2, число с лидирующими нулями

```
Enter base of numeral system (2 - 20)
8
Enter number in chosen system. Use 'A' - 'J' asdigits for >10-based systems
13A
Wrong format or too big number!
1234567
Original: 1234567
To decimal: 342391
Reversed: 7654321
Reversed to decimal: 2054353
This is a parent processThe child process exited normally with code 0.
```

Рисунок 3 – Пример работы программы 3, с попыткой ввода неверного формата