Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра «Вычислительной техники»

ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №4

«Порождение и замещение процессов»

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

Студент КИ20-07Б, 032049287 А.С. Базаров
номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2022

Содержание

Введение

Цель работы: изучение понятия процесса, функций порождения и замещения процессов. Получение опыта создания многопроцессных программ.

Задание: составить программу на языке С, в которой вызывается меню:

1. Выполнить введенную команду
2. Запустить на исполнение файл со сценарием (см. вариант в работе 1)
3. Выполнить изменение указанных прав доступа для указанного файла
4. Выход

Для выполнения пунктов 1-3 использовать порождение дочернего процесса

Для третьего пункта предполагается реализация посредством команд *shell*, сформированных в строки аргументов функций *execlp* или *system*, но если число команд *shell* велико тогда можно создать командный файл и запустить его с помощью *execlp* или *system*

По заданию требуется составить схему процессов, указать программный код, выполняемый каждым процессом, указать действия, выполняемые процессами. Ответить на вопросы:

- сколько процессов создается во время работы программы?
- все ли процессы завершаются?

Ход работы

1 Программный код

Запуск программы начинается с запуска меню в цикле, в котором и будет выполняться вся программа (рис. 1).

```
86 int main()
87 {
88         while (1)
89         {
90                 int choise = choose();
91                 switch(choise)
92                 {
93                         case 1:
94                         execCommand();
95                         break;
96                         case 2:
97                         runScript();
98                         break;
99                         case 3:
100                        editRightsForFile();
101                        break;
102                        default:
103                        return 0;
104                }
105        }
106        return 0;
107 }
```

Рисунок 1 – Функция *main*

Функция выбора пункта меню изображено на рисунке 2.

```
 9 int choose()
10 {
11         int choise = 0;
12         printf("[PID %d] Выберите режим работы:\n",getpid());
13         puts("1. Выполнить введенную команду");
14         puts("2. Удалить нелидирующие процессы, PID которых >= N");
15         puts("3. Изменить указанные права доступа для указанного файла");
16         puts("4. Выход");
17         do
18         {
19                 printf(">> ");
20                 scanf("%i", &choise);
21         }
22         while (choise < 1 && choise > 4);
23         return choise;
24 }
```

Рисунок 2 – Вызов функции меню

В данной функции вызывается вызов функции *fork()*, после которой появляется процесс считывания команды и выполнение путем функции *system()* (рис. 3)

```
26 void execCommand()
27 {
28         int st;
29         signal(SIGCHLD, SIG_IGN);
30         if (fork() == 0)
31         {
32                 printf("[PID %d] Введите необходимую команду: ", getpid());
33                 fflush(stdin);
34                 char *temp = NULL, command[255];
35                 scanf("%s", temp);
36                 fgets(command, sizeof command, stdin);
37                 system(command);
38                 exit(0);
39         }
40         wait(&st);
41 }
```

Рисунок 3 – Функция выполнения команды

На рисунке 4 представлена функция выполнения скрипта, который удаляет процессы, чей $PID \geq N$. Запуск функции порождается функцией *fork()*.

```
43 void runScript()
44 {
45         int st;
46         signal(SIGCHLD, SIG_IGN);
47         if (fork() == 0)
48         {
49                 printf("[PID %d] Введите N: ", getpid());
50                 fflush(stdin);
51                 char N[255], command[255] = "sudo ~/Documents/lab1.sh ";
52                 scanf("%s", N);
53                 strcat(command, N);
54                 system(command);
55                 printf("[PID %d] Скрипт завершен дочерним процессом\n", getpid());
56                 exit(0);
57         }
58         wait(&st);
59 }
```

Рисунок 4 – Функция вызова выполнения скрипта

На рисунке 5 изображена функция изменения прав файлу. Функция также порождается отдельный процессом.

```c
void editRightsForFile()
{
        int st;
        signal(SIGCHLD, SIG_IGN);
        if (fork() == 0)
        {
                printf("[PID %d] Введите путь к файлу, у которого необходимо изменить права: ", getpid());
                fflush(stdin);
                char path[255];
                scanf("%s", path);
                printf("[PID %d] Введите права в формате символьного или восьмеричного типа записи прав: ", getpid());
                fflush(stdin);
                char rights[255];
                scanf("%s", rights);
                char command[255] = "chmod ";
                strcat(rights, " ");
                strcat(command, rights);
                strcat(command, path);
                system(command);
                printf("[PID %d] Файл %s на права %sизменен успешно.\n", getpid(), path, rights);
                exit(0);
        }
        wait(&st);
}
```

Рисунок 5 – Функция изменения прав файлу

2 Составление схемы процессов

По варианту требуется составить схему процессов для следующей программы: $if(fork() == 0)\ for\ (int\ i = 0\ ; i < 2; i++)fork();$

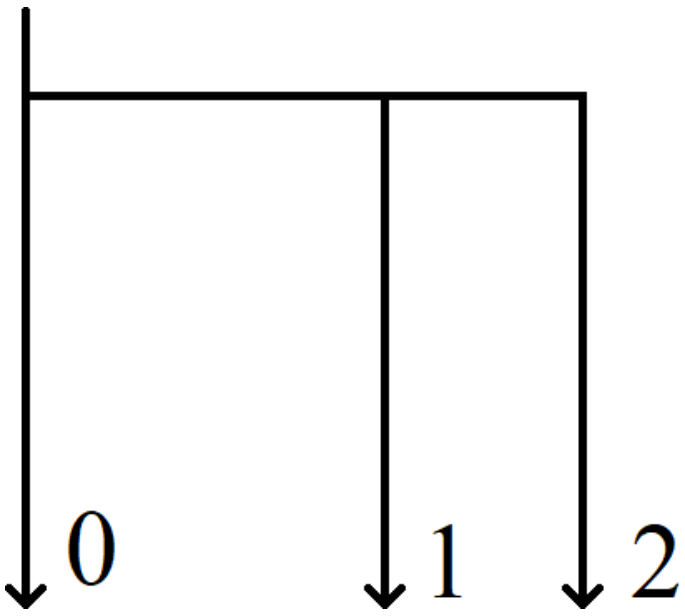

Рисунок 6 – Схема процессов

В таблице 1 представлен программный код, выполняемый каждым процессом, а в таблице 2 – словесное описание

Таблица 1 – Выполнение программного кода

№ процесса	**Выполняемый код**
0	$for\ (int\ i = 0\ ; i < 2; i + +) fork()$
1	–
2	–

Таблица 2 – Описание выполняемого программного кода

№ процесса	**Выполняемый код**
0	Порождение дочернего процесса, который выполняет порождение двух процессов
1	–
2	–

Заключение

В данной работе я изучил базовые утилиты, используемые в Linux для работы с текстом, включая awk, а также построил схему процессов по заданному по варианту программе.