Министерство образования Республики Беларусь
Учреждение образования
“Брестский Государственный технический университет”
Кафедра ИИТ

Лабораторная работа №4

По дисциплине “Операционные системы”

Тема: “GCC. Процессы”

Выполнил:
Студент 2 курса
Группы ИИ-21
Кирилович А. А.
Проверил:
Козинский А. А.

Брест 2022

Цель работы: ознакомится с работой компилятора gcc. А также изучить функции для работы с процессами в линуксе.

Ход работы:

Написать программу, которая будет реализовывать следующие функции:

• сразу после запуска получает и сообщает свой ID и ID родительского процесса;

• перед каждым выводом сообщения об ID процесса и родительского процесса эта информация получается заново;

• порождает процессы, формируя генеалогическое дерево согласно варианту, сообщая, что "процесс с ID таким-то породил процесс с таким-то ID";

• перед завершением процесса сообщить, что "процесс с таким-то ID и таким-то ID родителя завершает работу";

• один из процессов должен вместо себя запустить программу, указанную в варианте задания.

На основании выходной информации программы предыдущего пункта изобразить генеалогическое дерево процессов (с указанием идентификаторов процессов). Объяснить каждое выведенное сообщение и их порядок в предыдущем пункте.

Вариант 5. Fork - 0112233; exec – 5.

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

int main() {
	pid_t pid;
	printf("Порождение процесса 1 PID=%d PPID=%d\n", getpid(), getppid());

	if ((pid = fork()) < 0) {
		printf("Ошибка при порождении процесса 2!\n");
		exit(1);
	}
	else if (pid == 0) {
		printf("Порождение процесса 2 (от процесса 1) PID=%d PPID=%d\n", getpid(), getppid());

		if ((pid = fork()) < 0) {
			printf("Ошибка при порождении процесса 4!\n");
			exit(1);
		}
		else if (pid == 0) {
			printf("Порождение процесса 4 (от процесса 2) PID=%d PPID=%d\n", getpid(),
getppid());
			printf("Завершился процесс 4 PID=%d PPID=%d\n", getpid(), getppid());
			exit(0);
		}
		else sleep(1);

		if ((pid = fork()) < 0) {
			printf("Ошибка при порождении процесса 5!\n");
			exit(1);
		}
		else if (pid == 0) {
			printf("Порождение процесса 5 (от процесса 2) PID=%d PPID=%d\n", getpid(),
getppid());
			printf("Завершился процесс 5 PID=%d PPID=%d\n", getpid(), getppid());
			execl("/bin/df","df",NULL);
			exit(0);
		}
		else sleep(1);

			printf("Завершился процесс 2 PID=%d PPID=%d\n", getpid(), getppid());
			exit(0);
		}
	else sleep(1);
```

```
	if ((pid = fork()) < 0) {
		printf("Ошибка при порождении процесса 3!\n");
		exit(1);
	}
	else if (pid == 0) {
		printf("Порождение процесса 3 (от процесса 1) PID=%d PPID=%d\n", getpid(), getppid());

		if ((pid = fork()) < 0) {
			printf("Ошибка при порождении процесса 6!\n");
			exit(1);
		}
		else if (pid == 0) {
			printf("Порождение процесса 6 (от процесса 3) PID=%d PPID=%d\n", getpid(),
getppid());
			printf("Завершился процесс 6 PID=%d PPID=%d\n", getpid(), getppid());
			exit(0);
		}
		else sleep(1);

		if ((pid = fork()) < 0) {
			printf("Ошибка при порождении процесса 7!\n");
			exit(1);
		}
		else if (pid == 0) {
			printf("Порождение процесса 7 (от процесса 3) PID=%d PPID=%d\n", getpid(),
getppid());
			printf("Завершился процесс 7 PID=%d PPID=%d\n", getpid(), getppid());
			exit(0);
		}
		else sleep(1);

		printf("Завершился процесс 3 PID=%d PPID=%d\n", getpid(), getppid());
		exit(0);
	}
	else sleep(1);

	return 0;
}
```

```
arsbrest@Ars:~/Code/UNIVERSITY/OS/LABA4$ gcc task.c -o task
arsbrest@Ars:~/Code/UNIVERSITY/OS/LABA4$ ./task
Порождение процесса 1 PID=14692 PPID=14634
Порождение процесса 2 (от процесса 1) PID=14693 PPID=14692
Порождение процесса 4 (от процесса 2) PID=14694 PPID=14693
Завершился процесс 4 PID=14694 PPID=14693
Порождение процесса 3 (от процесса 1) PID=14697 PPID=14692
Порождение процесса 5 (от процесса 2) PID=14698 PPID=14693
Завершился процесс 5 PID=14698 PPID=14693
Порождение процесса 6 (от процесса 3) PID=14699 PPID=14697
Завершился процесс 6 PID=14699 PPID=14697
Filesystem     1K-blocks     Used Available Use% Mounted on
tmpfs            1573300     2084   1571216   1% /run
/dev/nvme0n1p5  92773208 14268376  73769920  17% /
tmpfs            7866492   223784   7642708   3% /dev/shm
tmpfs               5120        4      5116   1% /run/lock
/dev/nvme0n1p4  29999996   444648  29555348   2% /data
/dev/nvme0n1p9 153707984 10640420 135186864   8% /home
/dev/nvme0n1p1     98304    68807     29497  70% /boot/efi
tmpfs            1573296     4752   1568544   1% /run/user/1000
Завершился процесс 2 PID=14693 PPID=14692
Порождение процесса 7 (от процесса 3) PID=14700 PPID=14697
Завершился процесс 7 PID=14700 PPID=14697
```

Вывод: ознакомился с работой компилятора gcc. А также изучил функции для работы с процессами в линуксе.