Лабораторная работа 2. Управление процессами

Задание

1. Написать программу, которая добавляет новые строки в среду процесса.
2. Кто выполняется первым после fork: отец или сын? Написать программу, которую можно использовать для проверки данной ситуации.
3. Написать программу, в которой процесс порождает новый и позволяет порожденному процессу завершиться. Оба процесса должны выводить свои идентификаторы (процесса, родительского процесса, группы процессов). Проверить идентификатор и статус завершения порожденного процесса. Выполнить данную программу и посмотреть ее результаты.
4. Изменить программу п. 3 так, чтобы родительский процесс завершился раньше порожденного. Какой процесс становится родительским для порожденного после того, как исходный родительский процесс завершился?
5. Изменить программу п. 3 так, чтобы родительский процесс выполнялся, не ожидая завершения порожденного процесса. Что произойдет с порожденным процессом, когда он завершится? Как убить зомби?
6. Изменить программу п. 3 так, чтобы оба процесса выполняли системный вызов pause после вывода идентификаторов. Запустить программу в фоновом режиме и посмотрите (с помощью ps), какие процессы существуют для данного пользователя. Выполнить kill -9 pid для родительского процесса. Что произойдет с порожденным? После очистки текущего сеанса, вновь выполнить программу, но уже в интерактивном режиме. Нажать на терминале клавишу CTRL-C. Что произойдет с родительским и порожденным процессами? Можно ли получить такой же результат, если нажать клавишу прерывания после того, как родительский процесс завершится?
7. Модифицировать программу, включив в нее setpgrp в порожденный процесс до того, как он будет выполнять pause. Повлияет ли на порожденный процесс нажатие клавиши прерывания в тот момент, когда родительский процесс "спит"? Будет ли показан порожденный процесс в выводе команды ps?
8. Открыть файл (достаточно большого размера), доступный для чтения. Имя файла передается из командной строки. После открытия файла создать параллельный процесс с помощью только fork. В обоих процессах создать свои файлы для записи, читать информацию из общего файла и копировать ее в собственные выходные файлы (не на экран). Вывести на экран содержимое полученных файлов из порожденного процесса по окончании записи в файл и из родительского процесса, дождавшись завершения порожденного процесса. Посмотреть, что изменится, если читаемую процессами информацию сразу выводить на экран.
9. Выполнить п. 8 при условии, что общий файл для чтения открывается в каждом из порожденных процессов.
10. Создать (с помощью связки fork - exec) параллельный процесс. Имя исполняемого файла для exec передается с помощью аргумента командной строки. Передать в порожденный процесс некоторую информацию через

список параметров (список аргументов в функции main). Каждый процесс (и родительский, и порожденный) должен вывести на экран список переданных ему аргументов и свое окружение.
11. Выполнить из программы на Си какую-либо команду Shell (cp или ls): с помощью вызовов fork-exec, с помощью библиотечной функции system. Необходимые для команды данные передать через аргументы командной строки.