Лекция 4

Program №1

Сигналы. Пример работы программы.

В начале работы программы срабатывает системный вызов signal() и привязывает сигнал SIGUSR1 к одному из обработчиков handler1. Далее выполнятеся проверка на то, что мы ребенок и, если мы им оказываеся, то переопределяем обработчик handler1 на handler2. Из ребенка используется системный вызов kill() и отправляется сигнал родителю. handler1 и handler2 - две обрабатывающие функции, которые принимают на вход наш сигнал с целочисленным типом данных.

Тестирование

```
alex@192:~/Рабочий стол/Lec4> ./Program_1
counter = 1
counter = 3
counter = 5
alex@192:~/Рабочий стол/Lec4>
```

Program №2

Неименованные каналы. Пример работы программы.

Для работы с неименованными каналами используются два дескриптора. Системный вызов pipe записывает два дескриптора с обработкой ошибок. Далее используется системный вызов fork() для получения дочернего процесса, и обрабатываем ошибки. Если cpid равняется 0, то процесс - потомок. Получается что один дескриптор - для чтения, а второй дескриптор - для записи. То есть сообщение передается между процессами с помощью каналов.

Тестирование

```
alex@192:~/Рабочий стол/Lec4> ./Program_2 smth
smth
alex@192:~/Рабочий стол/Lec4>
```

Program №3

Именованные каналы. Пример работы программы.

Создаём дескриптор канала и именованный канал с правами доступа для всех. После запускается цикл прослушивания канала. В момент, когда в этот канал попадает сообщение, то выводится количество символов, включая 0 конца строки, и само сообщение.

Тестирование:
У меня почему-то, когда я отправляю сообщение, он зависает и не выводит ничего, хотя pipe файл он создает. Не смог никак решить эту проблему(.

```
alex@192:~/Рабочий стол/Lec4> ./Program_3
mypipe is created
^Z
[2]+  Остановлен    ./Program_3
alex@192:~/Рабочий стол/Lec4> echo Hi! > mypipe
^Z^X^C
```