Операционные системы
лекции 11, 12

- Взаимодействие процессов

  файл
  неименованный канал (pipe)
  именованный канал (fifo)
  сигналы

4.11.2010

Пример (повторение)

Что напечатает следующая программа (если вариантов несколько — назвать их все):

```
int main(){
  int q;
  q=1;
  if (fork()){
    q=2;
    wait(NULL);
  }
  else {
   printf("%d \n",q);
   q=3;
  }
  printf("%d \n",q);
  return 0;
}
```

Взаимодействие процессов

Процесс 1 (диалог с пользователем)

получает от пользователя (со стандартного ввода) 100 символов и передает их процессу 2.

./get_c

Процесс 2 (проверяющий)

если переданный символ является цифрой — печатает в стандартный вывод эту цифру, иначе — ничего не делает.

./check_digits

```
.....
if (fork()==0) {  execl("./get_c","get_c",NULL);  return 1;  }
if (fork()==0) {  execl("./check_digits","check_digits",NULL); return 1;   }
wait(NULL);
wait(NULL);
.....
```

Неименованный канал

```
#include <unistd.h>
int pipe (int * fd)
```

Использование:

```
int fd[2];
char c='*';

pipe( fd);

write(fd[1],&c,1);
read(fd[0],&c,1);
```

Особые случаи работы с каналом

Задача

Организация конвейера

Реализация конвейера процессов

pr1 | pr2 — для двух процессов

pr1 | pr2 | ..... | prn - для n процессов

Именованные каналы

Имеют точку привязки в файловой системе — имя

Создание именованного канала:

```
#include <sys/types.h>
#include <sys/stat.h>
int mkfifo ( const char *pathname, mode_t mode );
```

Затем канал нужно открыть — open
O_RDONLY или O_WRONLY

Применимы системные вызовы
read
write
close

unlink (pathname) — удаление.

Сигналы

Сигнал — средство уведомления процесса о некотором событии (ошибке, внешнем событии, выполнении запроса).

Количество сигналов в современных версиях UNIX – около 30
<signal.h>

	Константа	Значение сигнала
	SIGINT	Прерывание выполнения по нажатию **Ctrl-C**
	SIGTERM	аварийное завершение работы, уничтожение процесса
	SIGKILL	**уничтожение процесса**
	SIGALRM	**сигнал от программного таймера**
	SIGCHLD	**сигнал о завершении процесса потомка**

Сигналы (продолжение)

Варианты реакции процесса на сигнал

1)по умолчанию

2)назначить некоторую функцию обработчиком и её вызывать при приходе сигнала

3)игнорировать

2) и 3) кроме SIGKILL или SIGSTOP

Установка функции обработчика

```
#include <signal.h>
void (*signal(int, void (*handler)(int)))(int);
```

Отправка сигнала

```
int kill(pid_t pid, int sig);

int raise(int sig);
```

Пример программы

```
#include <stdio.h>
#include <signal.h>
int cnt = 0;
void sigint_handler(int signo){
 /* signal(SIGINT, sigint_handler); */
  printf("Ctrl-C pressed\n");
  if (++cnt == 3) {
    signal(SIGINT, SIG_DFL);
    raise(SIGINT);
  }
}

int main(void){
  signal(SIGINT, sigint_handler);
  while (1) {
  printf("Some string to print\n");
  }
  return 0;
}
```

Анализ завершения процесса по сигналу

Получение статуса завершения потомка родителем

```
#include <sys/types.h>
#include <sys/wait.h>
pid_t wait(int *status);

WIFEXITED(*status)
WEXITSTATUS(*status)

WIFSIGNALED(*status)
WTERMSIG(*status)
```

Обработка сигналов от программного таймера

```
#include <unistd.h>
unsigned int alarm(unsigned int seconds);
-------------------------------------
void alrm(int s) {
printf("\n жду имя \n");
alarm(5);
/* signal(SIGALRM, alrm); */
}
int main(int argc, char **argv){
char s[80];
     signal(SIGALRM, alrm);
     alarm(5);
     printf("Введите имя \n");
     fgets(s,80,stdin);
     return 0;
}
```

```
int target_pid, cnt, fd[2], status;
void SigHndlr(int s)   {
      if (cnt < MAX)    {
            read(fd[0], &cnt, sizeof(int));printf("%d \n", cnt); cnt++;
            write(fd[1], &cnt, sizeof(int));
            kill(target_pid, SIGUSR1);
      }
      else if (target_pid == getppid()) {
         close(fd[1]); close(fd[0]); exit(0);
              } else
                   kill(target_pid, SIGUSR1);
}
int main(int argc, char **argv){
      pipe(fd);
      if (target_pid = fork())    { signal (SIGUSR1, SigHndlr);
     wait(&status);close(fd[1]); close(fd[0]);return 0;
      }
      else{ signal (SIGUSR1, SigHndlr);
    target_pid = getppid();     write(fd[1],&cnt,sizeof(int));
            kill(target_pid, SIGUSR1);
      }
}  /* в каких случаях программа НЕ реализует игру в пинг-понг ?*/
```