Pipe

- Pipes:
  - two types of pipes, *named pipes* and *unnamed pipes*
  - name pipes:
    - like a file (create a named pipe (*mknod*), open, read/write)
    - can be shared by any number of processes
  - Unnamed pipes:
    - an unnamed pipe does not associated with any file
    - can only be shared by related processes (descendants of a process that creates the unnamed pipe).
    - Created using system call pipe().

- The pipe system call
  - open unnamed pipes
  - syntax

    int pipe(int fds[2])
  - semantic

    create a pipe and returns two file descriptors fds[0] and fds[1]

    a read from fds[0] accesses the data written to fds[1] on a fifo basis.

    the pipe has a limited size (64K in most systems) -- cannot write to the pipe infinitely.

```
#include <unistd.h>
#include <stdio.h>
main()
{
    char *s, buf[1024];
     int fds[2];
     s = “hello world\n”;
     pipe(fds);
     write(fds[1], s, strlen(s));
     read(fds[0], buf, strlen(s));
     printf(“fds[0] = %d, fds[1] = %d\n”, fds[0], fds[1]);
     write(1, buf, strlen(s));
}
```

/ example1.c */*

```
#include <unistd.h>
#include <stdio.h>
main()
{
    char *s, buf[1024];
     int fds[2];
     s = “hello world\n”;
     pipe(fds);
     if (fork() == 0) {
         printf(“child process: \n”);
         write(fds[1], s, 12); exit(0);
     }
     read(fds[0], buf, 6);
     write(1, buf, 6);
}
```

/ example2.c : using pipe with fork*/*

IPC can be used to enforce the order of the execution of processes.

```
main()
{
    char *s, buf[1024];                                    11111   33333   11111
    33333
     int fds[2];                                           22222   44444   33333
    11111
     s = “hello world\n”;                                  33333   11111   22222
    22222
     pipe(fds);                                            44444    22222   44444
    44444
     if (fork() == 0) {
         printf(“11111 \n”);     /* how to make 111111 before 444444 */
         read(fds[0], s, 6);
         printf(“22222\n”);
     } else {
       printf(“33333\n”);
       write(fds[1], buf, 6);
       printf(“44444\n”)
    }
} /* example3.c */
```