Non-blocking I/O와 Select

Non-blocking I/O

- Blocking I/O
  - file/socket/pipe등에서 read를 하려고 했는데, 읽을 내용이 없거나, write를 하려고 했는데 가용 공간이 없어서 쓸 수 없는 경우 blocking이 일어남
  - 대안 1
    - 파이프의 경우 fstat으로 읽거나 쓸 수 있는 데이터의 양을 먼저 확인하고 read/write를 수행
  - 대안 2
    - fcntl을 이용하여 O_NONBLOCK 플래그 설정

```
#include <fcntl.h>

if (fcntl(filedes, F_SETFL, O_NONBLOCK) == -1)
        perror("fcntl");
```

    - 이렇게 설정하는 경우 read/write는 blocking 되지 않고 바로 -1을 return하며 errno는 EAGAIN이 됨
    - 따라서 그 상황에 대한 대처를 하는 방식으로 처리
      - 일정 기간 sleep 하여 blocking 상황이 해소된 후 다시 read/write 시도

select (1)

- 동시에 여러 개의 파일 기술자들을 감시하면서 저장된 파일 기술자 중 읽기, 쓰기, 또는 미결 상태의 오류가 있는 기술자들을 표시함
- 또한 select 명령시 시간을 지정하여 일정 시간 후에는 반드시 종료되도록 할 수도 있음
- 사용법

```
#include <sys/time.h>
int select (int nfds, fd_set *readfds, fd_set *writefds, fd_set *errorfds, struct
                    timeval *timeout);
```

  - nfds : 잠재적인 흥미를 갖고 있는 파일 기술자 수
    - 일반적으로 마지막으로 open하거나 생성시킨 fd 값 + 1
  - readfds, writefds, errorfds : 흥미를 가지고 있는 파일 기술자들의 집합(bit masks)
    - select가 반환되면서 이 집합들에는 실제 사건이 일어난 파일 기술자들이 표현됨
    - 따라서 반복하여 select를 호출하는 경우 이 집합들을 매번 다시 설정해야 함

select (2)

- 사용법 (계속)
  - 파일기술자 집합 조정
    - void FD_ZERO(fd_set *fdset);
    - void FD_SET(int fd, fd_set *fdset);
    - void FD_CLR(int fd, fd_set *fdset);
  - timeout
    - NULL : 사건이 일어날 때까지 봉쇄
    - 0 : 즉각 복귀
    - 0이 아닌 값 : 지정된 시간 후 복귀
  - return 값
    - 오류시 -1
    - 타임아웃시 0
    - 그외 흥미있는 파일 기술자 수
  - return 값이 양수인 경우 파일기술자 집합들에 설정된 비트 마스크를 하나씩 확인해야 함
    - 파일기술자 확인 매크로
      - int FD_ISSET(int fd, fd_set *fdset);
      - 예 :
        - if (FD_ISSET(0, &fdset)) {
          - ...

샘플 프로그램

- Select를 이용하여 표준 입력과 파이프 3개를 동시에 확인하여 처리