Pipes

pipe

```
int pipe(int fd[2])
```

- **pipe** cria um pipe sem nome e associa um descritor de ficheiro a cada extremo
- **fd** é um array onde serão colocados os dois descritores de acesso ao pipe:
  - **fd[0]** terá o descritor usado para leitura
  - **fd[1]** terá o descritor usado para escrita
- pipe retorna **0** se bem sucedido e **-1** caso contrário
- A escrita e leitura num pipe sem nome fazem-se através das primitivas **write** e **read**, considerando os descritores **fd[1]** e **fd[0]**, respetivamente
- Ler dados de um pipe **remove** esses dados do pipe
- Um processo **bloqueia** quando tenta ler de um pipe vazio
- Logo que possível, os processos devem fechar, através de **close**, os descritores/ extremidades dos pipes que não vão usar

write

```
int write(int fd, const void *buf, size_t count)
```

- **fd** é o descritor do pipe (ou ficheiro), **buf** é o endereço dos dados a escrever, e **count** é o número de bytes a escrever
- Retorna o número de bytes escritos, ou **-1** em caso de erro

read

```
int read(int fd, void *buf, size_t count)
```

- **fd** é o descritor do pipe (ou ficheiro), **buf** é o endereço do buffer onde os dados serão armazenados, e **count** é o número máximo de bytes a ler
- Retorna o número de bytes lidos, ou **-1** em caso de erro

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(){
    int fd[2], num;
    pipe(fd);

    if(fork()==0){
        close(fd[0]);
        do{
            scanf("%d", &num);
            write(fd[1], &num, sizeof(int));
        } while(num!=0);
        exit(0);
    }

    close(fd[1]);
    do{
        read(fd[0], &num, sizeof(int));
        printf("%d\n", num);
    } while(num!=0);

    return 0;
}
```