**SYSTEM CALL LAB SO**

**dup2(*fd1, fd2*)** copia il file descriptor del 1° nel 2°,

es: **dup2(*fd[1],STDOUT\_FILENO*);**

**fork()** crea un processo figlio, ritorna un valore minore di 0 se fallisce, METTERE SEMPRE UN CONTROLLO, si deve salvare in una variabile pid\_t

**read(*fd,string,strlen(string)*)** Legge da un file descriptor fino a un numero specifico di byte (*specificato nel 3°parametro*) e lo salva in una stringa (*specificata nel 2°parametro*).

**open(*path,args*)**  Apre un file dato dal path e con i permessi definiti dal 2°parametro, come:

O\_RDONLY, O\_WRONLY, O\_CREAT, O\_RDWR

**write(*fd,stringa,strlen(stringa)*)** Scrive nel file descriptor dato la stringa passata come 2°argomento per un numero di byte definito dal 3°parametro, ricordiamo i file descriptor più importanti:

STDOUT\_FILENO STDIN\_FILENO

**pipe(*fd*)** deve avere come parametro un array di 2 elementi interi e crea un pipe, se restituisce un valore minore di 0 vuol dire che non è andata a buon fine.

DA RICORDARE: fd[0] è la parte in lettura, fd[1] è la parte in scrittura

Chiudere sempre il lato corretto nel figlio o padre

**Pipe2(*fd,tag*)0020**

**lseek(*fd,byte\_offset,tag da dove iniziare*)**permette di spostarsi all’interno di un file velocemente, sfruttando un offset di byte di cui ci si vuole spostare, il 3°parametro rappresenta il punto iniziale da cui spostarti, possono esserci 3 tag:

SEEK\_SET SEEK\_CUR SEEK\_END

**Opendir(path)**, da assegnare ad una variabile **DIR \***

**Readdir(DIR)**, data una variabile DIR \* essa restituisce una variabile **struct dirent**, da ricordare:

dt->d\_name

dt->d\_type es: dt->d\_type!=DT\_DIR

**Stat(path, buffer)**, data una path di un file ed una variabile **struct stat** essa restituisce tutti i dettagli di un file, da ricordare:

Immagine che contiene testo, schermata, numero, Carattere

Descrizione generata automaticamente