

Seconda Lezione

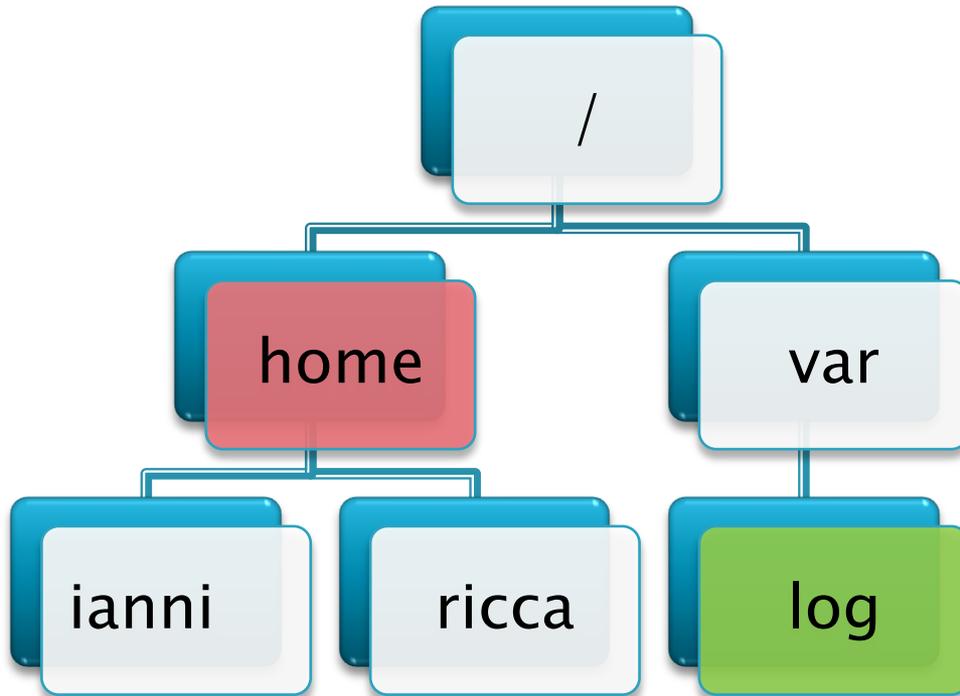
SITO DEL CORSO:

- ▶ `http://so.gibbi.com`
- ▶ [https://www.mat.unical.it/informatica/Sistemi Operativi](https://www.mat.unical.it/informatica/Sistemi%20Operativi)
- ▶ Gruppo Facebook

La shell Linux

- ▶ Come si accede a BASH
 - Tasti utili: Ctrl-Alt-F1, Ctrl-Alt-F7
- ▶ Il Primo comando : **exit** (CASE SENSITIVE!!)
 - Si può uscire con CTRL-D
- ▶ Il secondo comando: **ls**
 - **Opzioni:** `ls -l`, `ls -a`, `ls -R`
 - `ls --full-time`
 - **Parametri:** `ls /usr`
- ▶ Come avere aiuto: **man**, **info**, opzione **--help**
- ▶ Non dimenticate le Linux HOW-TO

Path assoluti e relativi



- Percorso assoluto
`/home/ianni`

- Percorso relativo
`../ianni`
`../../home/ianni`
`Ianni`

Percorsi assoluti nei vostri programmi == Problemi

Le directories

- ▶ Qui si usa “/” anzichè “\”
- ▶ I riferimenti alle directory sono
 - Assoluti: `/usr/bin`
 - Relativi alla **directory corrente**: `../usr`
 - Simboli speciali: `.` e `..`
- ▶ Alcune directory predefinite
 - `/`
 - `/bin`, `/sbin`, `/usr/bin`, `/usr/sbin` (eseguibili)
 - `/dev` (dispositivi. tutto è un file!)
 - `/etc` (file di configurazione)
 - `/home` (home degli utenti)
 - `/lib` (librerie)
 - `/mnt` (file system esterni, non ci sono lettere di drive!)
 - `/opt` (componenti opzionali)
 - `/tmp` (file temporanei)
 - `/usr` (molti eseguibili e tanto altro)
 - `/var` (file variabili, soggetti a modifica continua, e.g. log in `/var/log`)
 - `/proc` (processi e varie)

Come navigare le directory

In ogni momento la shell ha una sua *cartella corrente*.
Ogni utente possiede una sua cartella \$HOME

- ▶ **mkdir** <nomedir> **Crea cartella**
- ▶ **cd** <nomedir>
corrente **Cambia cartella**
- ▶ **cd** **Va in \$HOME**
- ▶ **rmdir** <nomedir>
cartella **Cancella**
- ▶ **pwd** **Dove sono?**
- ▶ **ls** <nomedir> **Cosa c'è qui?**

Come gestire i file: comandi

- ▶ **cp** file1 file2 **Copia**
- ▶ **cp** file1 ... filex directory
- ▶ **mv** file1 file2 **Sposta/Rinomina**
- ▶ **mv** file1 ... filex directory
- ▶ **less** filename **Mostra**
- ▶ **file** filename **Analizza**

Caratteri Jolly

▶ Asterisco "*"

- Sta per 0 o più caratteri
- Esempi:
 - *.txt
 - *txt
 - *txt*

▶ Punto interrogativo "?"

- Sta per un qualsiasi singolo carattere
- Esempi: x??, t?t

▶ Parentesi quadre [] e "!" oppure "^"

- [AB]* tutti i file che cominciano per A o B

Come funzionano i caratteri jolly

- ▶ C'è una fase di pre-espansione!
 - `ls y*` equivale a
 - `ls yacc yes ypcat ... ypmatch ...`
- ▶ Questo crea una **GROSSA** differenza con Windows
 - Windows: `copy *.txt *.bak`
 - Linux: `cp *.txt *.bak`
 - (pensate al meccanismo di espansione)
- ▶ I file nascosti vengono di solito ignorati (quelli che cominciano per “.”)

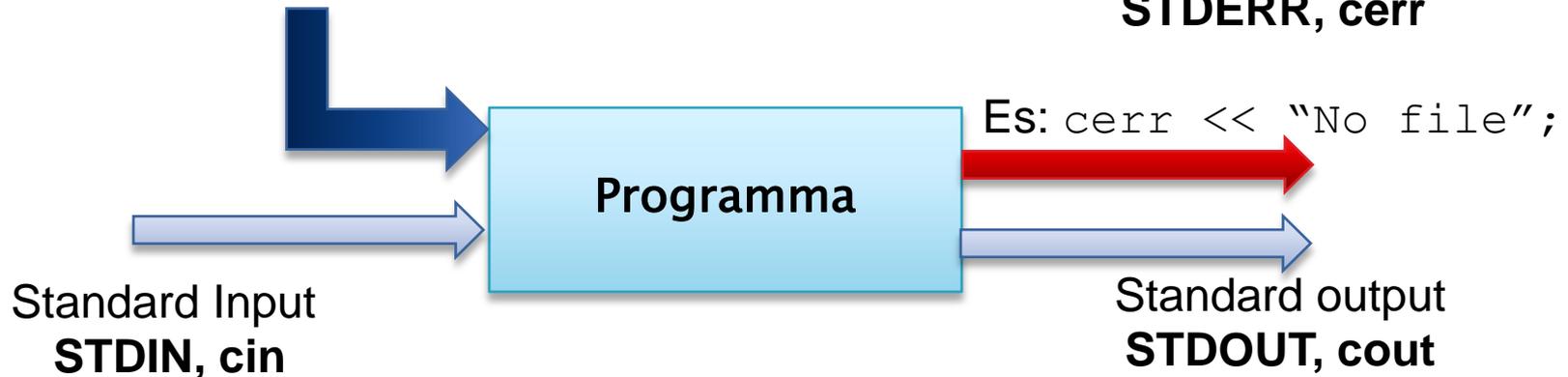
Navigare tra i comandi

- ▶ CTRL-R: cerca comandi che cominciano per una certa lettera
- ▶ Muoversi con le frecce: scorre i precedenti comandi
- ▶ Tab: completa i nomi di file (come con CMD in Windows)

Programmi console e loro canali di input

Linea di comando
`@ARGV, argv[]`

Es: `nomedir = argv[1];`

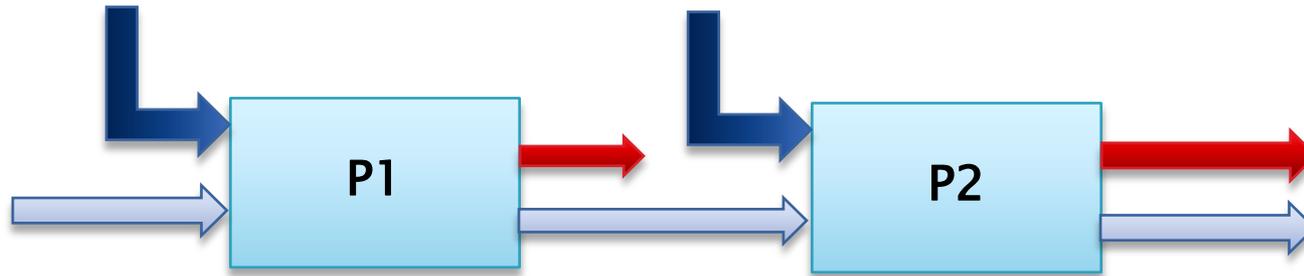


Es: `cin >> nomeUtente;`

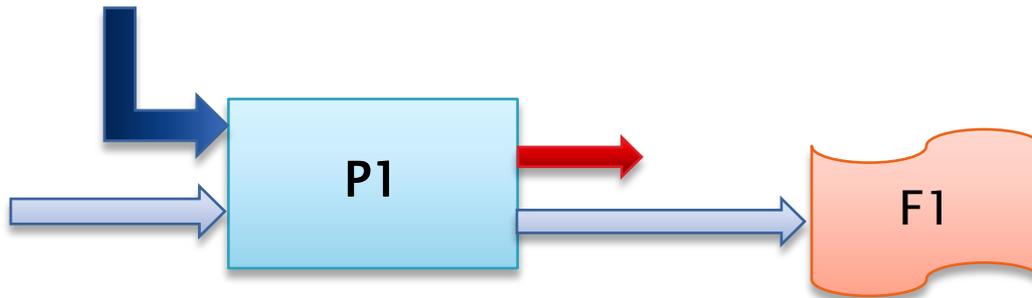
Es: `cout << "Ciao";`

- La linea di comando viene specificata **PRIMA** che il programma sia lanciato
- I valori vengono letti da STDIN **MENTRE** il programma gira

Redirezione e piping

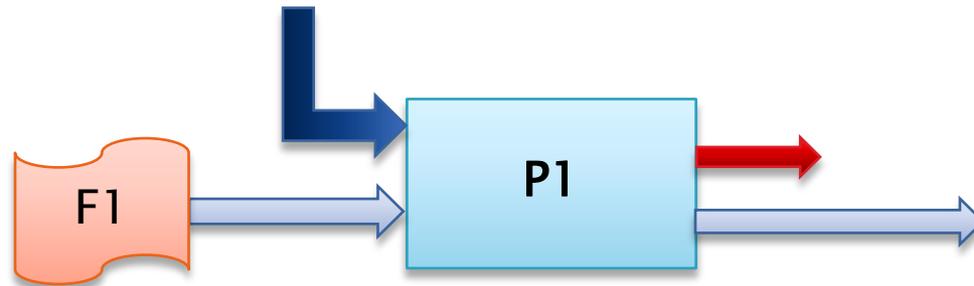


- **P1 | P2** Esegue P1 e P2: STDOUT di P1 diventa STDIN di P2



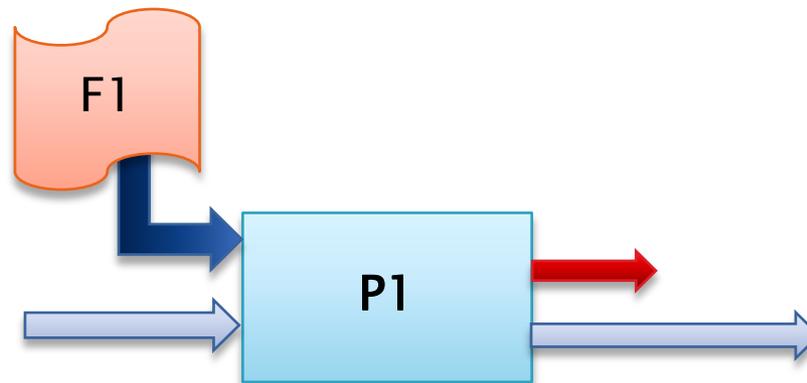
- **P1 > F1** STDOUT di P1 finisce sul *File* F1

Redirezione e piping



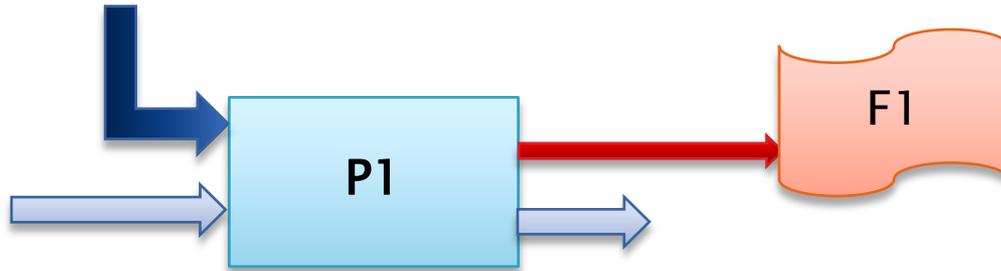
- $P1 < F1$ STDIN di P1 riceve IL CONTENUTO del *File* F1

NON E' LA STESSA COSA DI



- $P1\ F1 \rightarrow \text{ARGV}[1] = "F1";$

Casi speciali

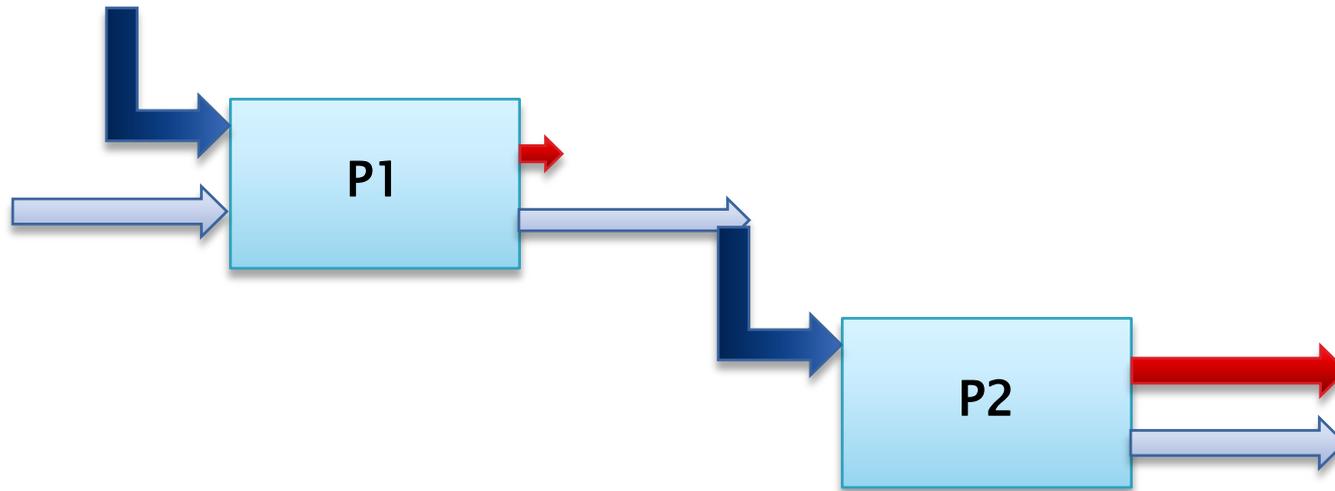


- **P1 2>&F1** STDERR di P1 finisce sul *File* F1
Linux Bash e Sh: P1 2>F1



- **P1 2>&1 >F1** STDERR e STDOUT di P1 finiscono entrambi sul *File* F1

Casi speciali (2)



- **P1 | xargs P2** Esegue P1 e P2: STDOUT di P1 diventa ARGV di P2

Redirezione e piping

- ▶ Non dimenticate che un programma è sempre agganciato a un canale di input (cin) e alcuni di output (cout e cerr)
- ▶ Agganciare **cout** a qualcosa di diverso dallo schermo:
 - `ls > out.txt` (crea un file out.txt)
 - `ls >> out.txt` (appende a out.txt)
- ▶ Agganciare **cin** a qualcosa di diverso
 - `grep "cat" < out.txt`
- ▶ Usare le pipe (leggi 'PAIP')
 - `cat /etc/services | sort`
- ▶ Si possono innestare le pipe
 - `cat /etc/services | sort | tail -n 50 | less`

Variabili di ambiente

- ▶ **env** Mostra tutte queste variabili
- ▶ **export** Cambia il valore di una di esse
- ▶ Variabili speciali
 - \$SHELL, \$PATH, \$HOME

Diritti di accesso

- ▶ file:///C:/Users/gio/Documents/Corsi/SMFN/Sistemi%20Operativi/Il%20tutorial%20-%20chmod%20tradotto/chmod_1.html