



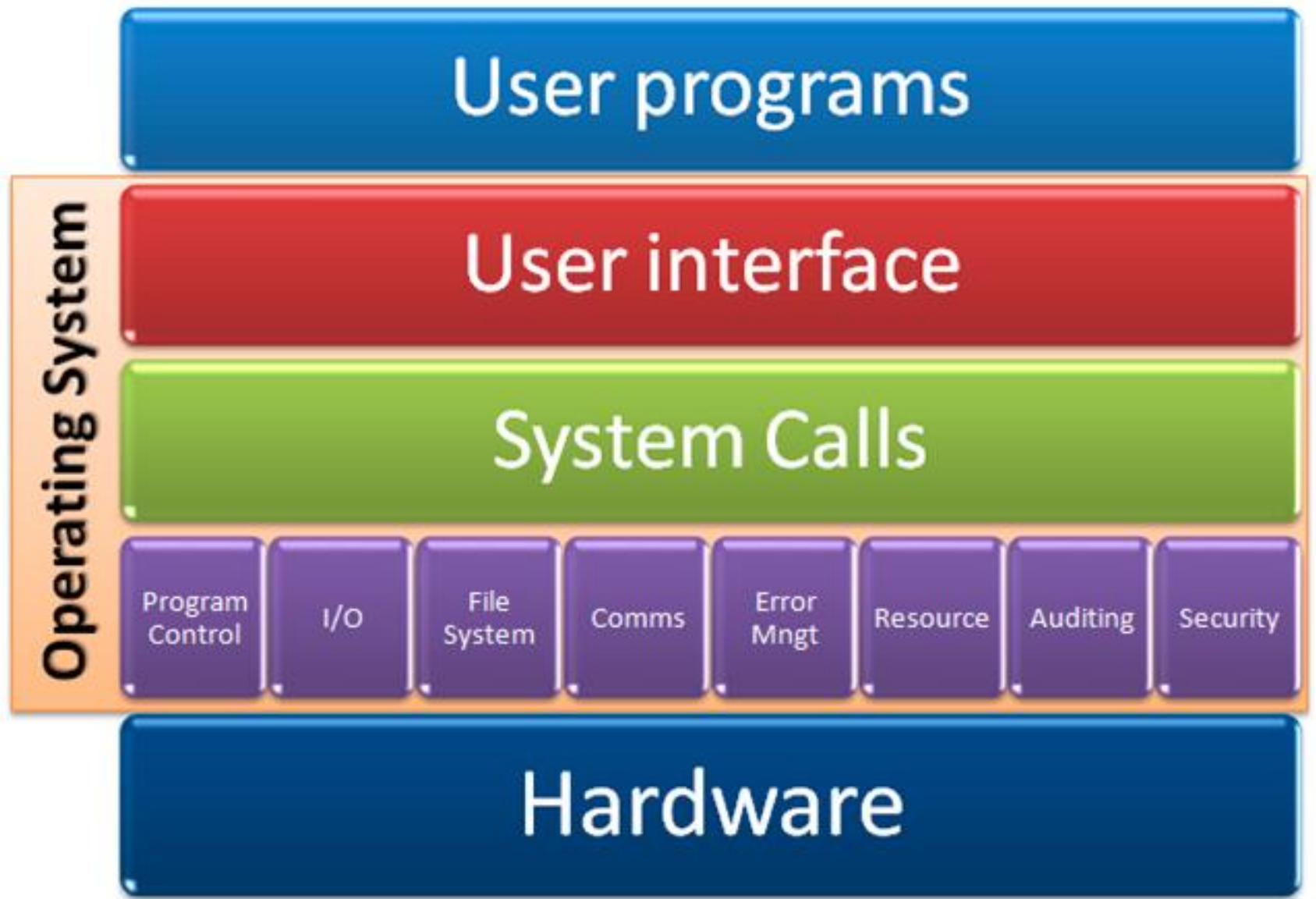
UNIVERSITÀ DEGLI STUDI
DI MILANO

Laboratorio Di Programmazione (A.A. 2024-2025)

Lezione 1 - Uso di Linux



Funzionalità di un Sistema Operativo

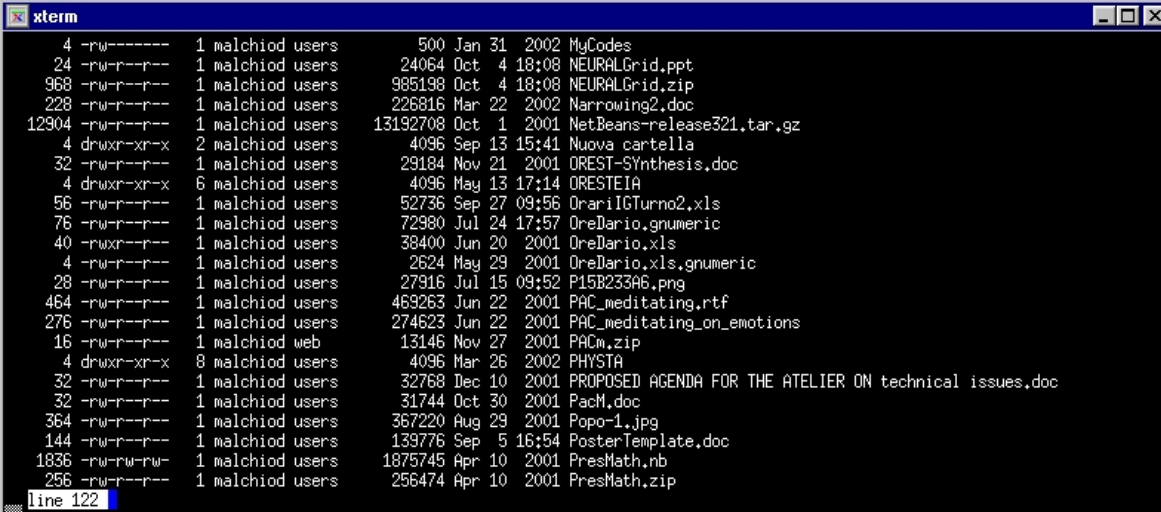


Graphical/Command-line User Interface

- Permettono l'interazione tra l'utente ed il sistema operativo.
- In alcuni sistemi operativi, ad es. Windows, l'interfaccia grafica è una **componente integrata**.
- In alcuni sistemi operativi, ad es. Linux, l'interfaccia grafica è una **componente opzionale** (si possono anche installare differenti interfacce).
- L'utente può interfacciarsi con il sistema operativo (indipendentemente da quale sia il sistema operativo) utilizzando la tastiera per impartire comandi sotto forma di linee di testo successive, comandi che vengono eseguiti da un opportuno interprete (**shell/Linux, Command Prompt/Windows**).
Questa "modalità di interazione" è detta **command-line user interface**.

Shell

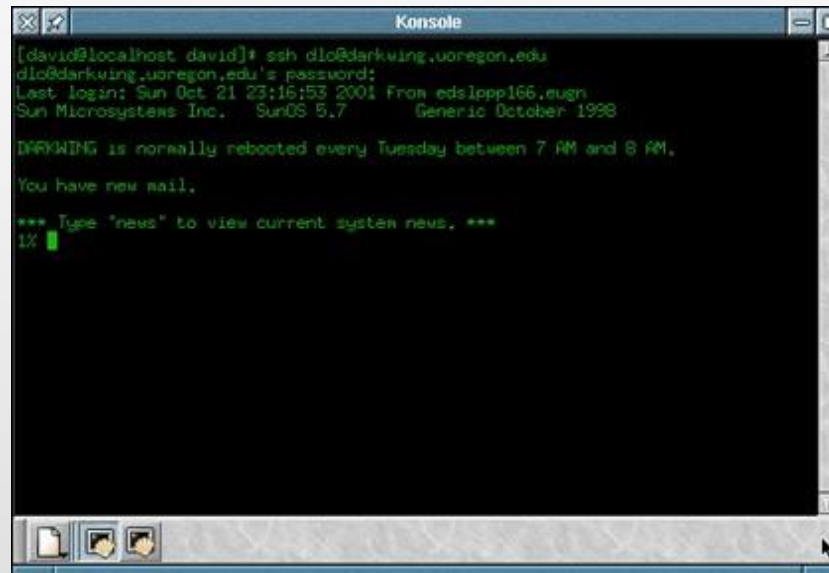
- Con il termine **shell** si identifica un generico interprete per i comandi che l'utente inserisce da un terminale a caratteri.
- Esistono diversi tipi di shell Linux. Alcuni esempi: Sh (shell), Csh (C shell), Tcsh (TC shell), Ksh (Korn shell), Bash (Bourne Again Shell).



```
xterm
4 -rw-r----- 1 malchiod users      500 Jan 31  2002 MyCodes
24 -rw-r----- 1 malchiod users    24064 Oct  4 18:08 NEURALGrid.ppt
968 -rw-r----- 1 malchiod users   985198 Oct  4 18:08 NEURALGrid.zip
228 -rw-r----- 1 malchiod users    226816 Mar 22  2002 Narrowing2.doc
12904 -rw-r----- 1 malchiod users 13192708 Oct  1  2001 NetBeans-release321.tar.gz
4 drwxr-xr-x  2 malchiod users      4096 Sep 13 15:41 Nuova cartella
32 -rw-r----- 1 malchiod users    29184 Nov 21  2001 OREST-Synthesis.doc
4 drwxr-xr-x  6 malchiod users      4096 May 13 17:14 ORESTEIA
56 -rw-r----- 1 malchiod users    52736 Sep 27 09:56 OrariIGTurno2.xls
76 -rw-r----- 1 malchiod users    72980 Jul 24 17:57 OreDario.gnumeric
40 -rw-r----- 1 malchiod users    38400 Jun 20  2001 OreDario.xls
4 -rw-r----- 1 malchiod users     2624 May 29  2001 OreDario.xls.gnumeric
28 -rw-r----- 1 malchiod users    27916 Jul 15 09:52 P15B233A6.png
464 -rw-r----- 1 malchiod users   469263 Jun 22  2001 PAC_meditating.rtf
276 -rw-r----- 1 malchiod users   274623 Jun 22  2001 PAC_meditating_on_emotions
16 -rw-r----- 1 malchiod web      13146 Nov 27  2001 PACm.zip
4 drwxr-xr-x  8 malchiod users      4096 Mar 26  2002 PHYSTA
32 -rw-r----- 1 malchiod users    32768 Dec 10  2001 PROPOSED AGENDA FOR THE ATELIER ON technical issues.doc
32 -rw-r----- 1 malchiod users    31744 Oct 30  2001 PacM.doc
364 -rw-r----- 1 malchiod users   367220 Aug 29  2001 Popo-1.jpg
144 -rw-r----- 1 malchiod users   139776 Sep  5 16:54 PosterTemplate.doc
1836 -rw-rw-rw- 1 malchiod users  1875745 Apr 10  2001 PresMath.nb
256 -rw-r----- 1 malchiod users   256474 Apr 10  2001 PresMath.zip
line 122
```

Shell - Cont'd

- Una shell indica all'utente la propria disponibilità ad accettare comandi visualizzando un messaggio (prompt).
- Il prompt è personalizzabile e può includere informazioni riguardo alla directory corrente, alla data e ora corrente, al sistema cui si è collegati, ...



```
[david@localhost david]* ssh dlo@darkwing.uoregon.edu
dlo@darkwing.uoregon.edu's password:
Last login: Sun Oct 21 23:16:53 2001 From eds1ppp166.eugn
Sun Microsystems Inc. SunOS 5.7 Generic October 1998

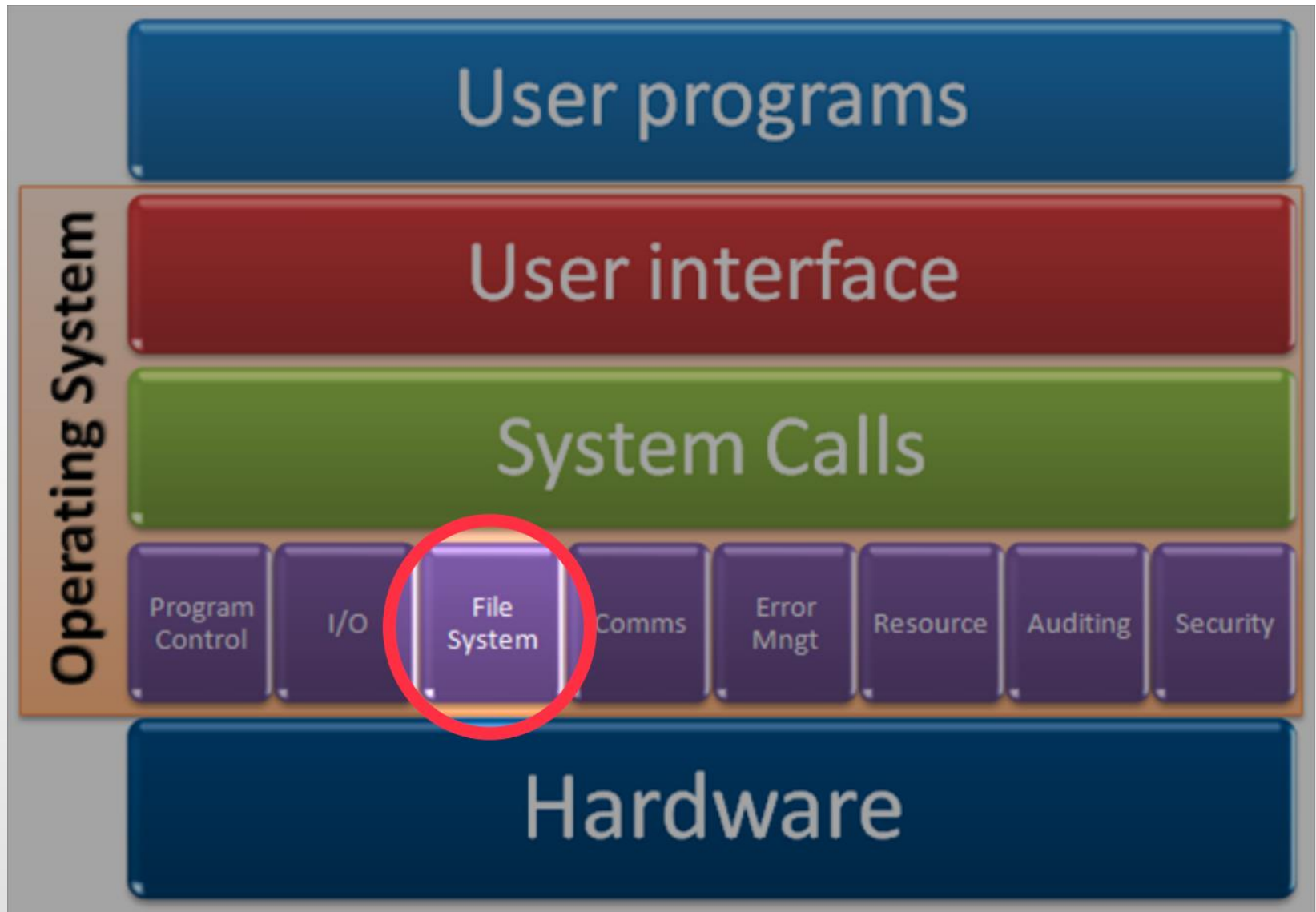
DARKWING is normally rebooted every Tuesday between 7 AM and 8 AM.
You have new mail.

*** Type 'news' to view current system news. ***
dlo █
```

Attenzione!

- In questa lezione ci focalizzeremo sulle principali operazioni che si possono effettuare utilizzando una shell Linux.

Funzionalità di un Sistema Operativo



Gestione del file system

- Il file system è la componente del sistema operativo preposta alla gestione delle informazioni memorizzate permanentemente, che risiedono tipicamente su disco.
- Le componenti di un file system sono:
 - il file
 - la directory

File e directory

- Un file memorizza permanentemente una serie di informazioni aventi unità logica
 - una applicazione (elaboratore testi, visualizzatore di clip multimediali, ...)
 - dei dati (una relazione, un video musicale, ...)
- Una directory (o cartella, o folder) è un "contenitore" che può includere file o altre directory.

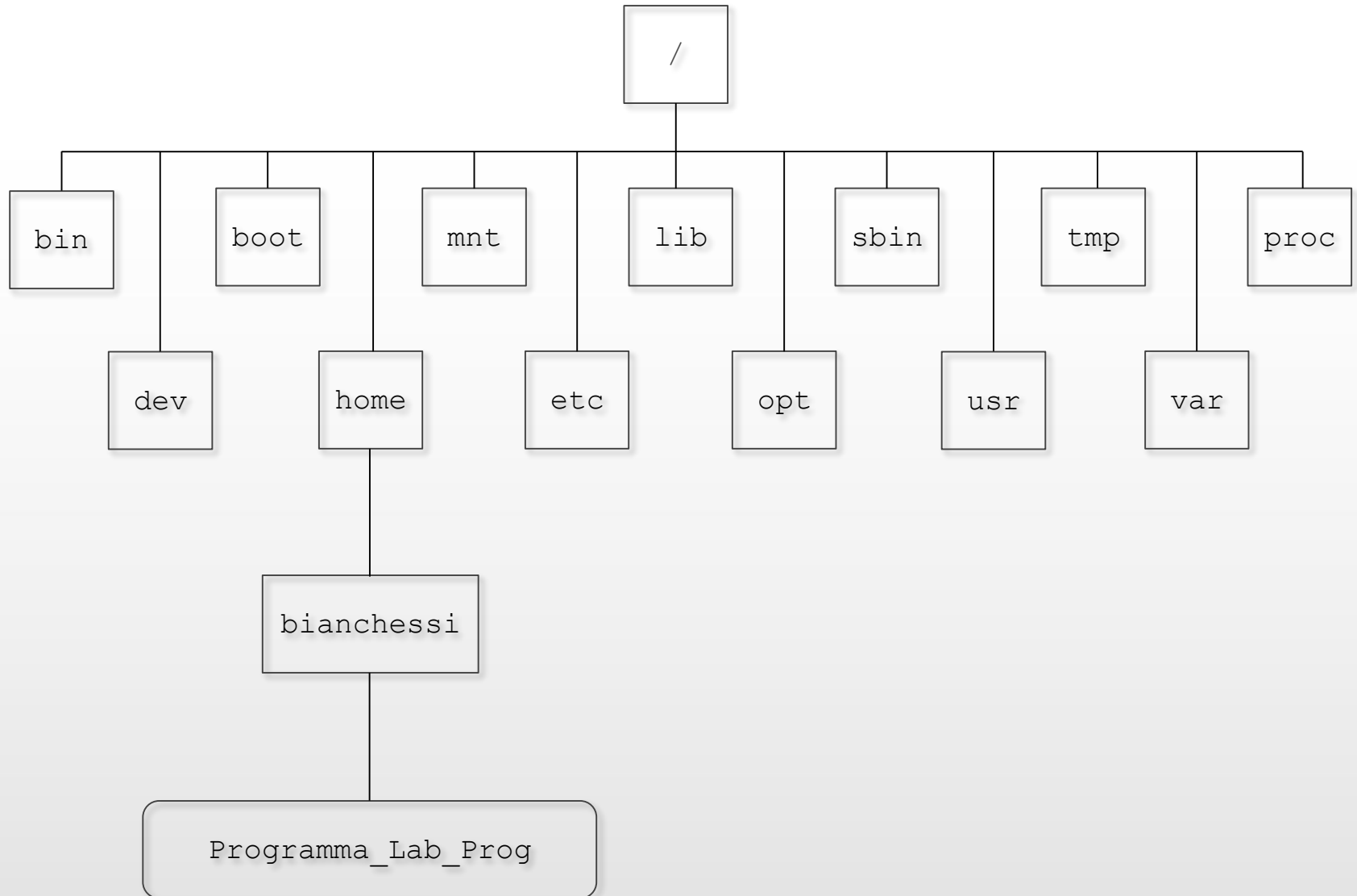
File system strutturati

- La possibilità di includere directory in directory fornisce l'opportunità di creare dei file system strutturati, in cui le informazioni sono memorizzate in modo ordinato.
- In file system ben strutturati risulta più facile localizzare ed accedere alle informazioni in modo veloce.

Gerarchia del file system

- Un file system può essere quindi rappresentato da una struttura gerarchica, ad albero rovesciato, nella quale:
 - esiste una directory principale (detta radice) cui tutta la struttura fa capo;
 - le altre directory rappresentano nodi intermedi dell'albero;
 - i file sono le foglie dell'albero.

Gerarchia del file system - Esempio (Linux)



Pathname

- Ogni file/directory all'interno del file system è individuato/a in base alla sua posizione nell'albero, cioè al cammino (**pathname** o **path**) che si deve percorrere per raggiungerlo/a partendo dalla radice.
- Relativamente all'esempio precedente, il pathname del file relativo al programma del corso di laboratorio è:
`/home/bianchessi/Programma_Lab_Prog`
 - Il primo carattere slash (/) indica la radice della struttura gerarchica, la root directory.
 - I successivi caratteri slash (/) separano i nodi nel pathname.

Alcuni comandi della shell di Linux per interagire con il file system...

Visualizzazione di directory

- Il comando `pwd` visualizza il pathname della directory in cui si è attualmente posizionati (directory corrente):

```
> pwd
```

```
/home/bianchessi
```

- Quando si apre un terminale, la directory corrente è automaticamente impostata alla propria home directory.

Visualizzazione di directory

- Il comando `ls` permette di visualizzare il contenuto della directory corrente.
- Se dopo `ls` si specifica il nome di una directory, vengono visualizzati i contenuti di quest'ultima.

Spostamento tra directory

- Il comando `cd` permette di cambiare la directory corrente:
 - se non vengono specificati argomenti, ci si posiziona nella home directory
 - se viene specificato come argomento il pathname di una directory, ci si posiziona in quest'ultima

Pathname assoluti e relativi

- Il pathname di una directory (o di un file) può essere indicato specificando:
 - un **pathname assoluto**, specificando tutto il percorso a partire dalla root directory;
 - un **pathname relativo**, specificando uno dei possibili percorsi a partire dalla directory corrente; a tal fine si consideri che:
 - . indica la directory corrente;
 - .. indica la directory che contiene la directory corrente.

Esercizio

- Inserite la directory corrente o il pathname relativo nelle celle vuote della seguente tabella:

Pathname assoluto	Directory corrente	Pathname relativo
/home/ciccioformaggio/Documents/document.odt	/home/ciccioformaggio/Documents	document.odt “oppure” ./document.odt
		home/ciccioformaggio/Documents/document.odt
	/home/ciccioformaggio/Images	
		../../document.odt
	/usr	
/home/ciccioformaggio/Documents	/home/ciccioformaggio/Documents	.
		home/ciccioformaggio/Documents
		..
	/home/ciccioformaggio/Images/sea	
		../home/ciccioformaggio/Documents

Operazioni con le directory

- Creazione: tramite il comando `mkdir`
- Eliminazione: tramite il comando `rmdir` (eseguito solo se la directory è vuota)
- Eliminazione di una directory e del suo contenuto: tramite il comando `rm -rf` (ATTENZIONE!)
- Altre operazioni (spostamento, variazione del nome, ...): tramite gli stessi comandi per i file (vedi slide successive)

Operazioni su file

- Creazione: tramite il comando `touch`
- Apertura
 - di file eseguibili: scrivendone il nome
 - di file di dati: tramite le relative applicazioni
- Visualizzazione dei contenuti: tramite i comandi `cat`, `more` e `less`
- Spostamento e/o modifica del nome: tramite il comando `mv` (vale anche per le directory)
- Creazione di una copia: tramite il comando `cp` (vale anche per le directory)
- Cancellazione: tramite il comando `rm`

Wildcard

- Nello specificare i nomi di file o directory, è possibile indicare un'espressione contenente dei **caratteri jolly** (o **wildcard**)
 - * indica una qualunque sequenza di caratteri
 - ? indica un qualunque carattere
 - [] indica un qualunque carattere appartenente alla sequenza indicata tra le parentesi

Esempi

	Diventa
m^*	m, ma, mb, ..., ma4, m1b, ...
$m?lo$	malo, mblo, m8lo, m+lo, ...
$m[aeiou]lo$	malo, melo, milo, molo, mulo

Permessi sui file/directory

- Gli utenti possono specificare i seguenti **permessi di accesso**:
 - **Read (R)** indica se possono essere letti i contenuti di un file o di una cartella
 - **Write (W)** indica se è possibile modificare il contenuto di un file o di una cartella
 - **Execute (X)** indica se è possibile eseguire un file o posizionarsi all'interno di una cartella

Permessi sui file/directory - Cont'd

- I permessi sui file sono specificabili su tre livelli:
 - relativamente all'utente che li crea
 - relativamente agli utenti facenti parte dello stesso gruppo dell'utente che li crea
 - relativamente ai rimanenti utenti
- Sono specificati da un terzetto di caratteri

```
$ ls -l
total 8
drwxrwxr-x 17 user group 4096 Jul 31 12:18 Dir
-rwxrwxr-x 1 user group 218 Jul 30 16:08 File
```

Documentazione

- Il comando `man` permette di visualizzare una descrizione delle funzionalità dei vari comandi utilizzabili all'interno di un terminale.

- Esempi

- `man ls`
- `man cd`
- `man man`

```
LS(1)                                User
Commands                             LS(1)
NAME
    ls - list directory contents
SYNOPSIS
    ls [OPTION]... [FILE]...
DESCRIPTION
    List information about the FILES (the current
    directory by default). Sort entries alphabetically
    if none of -cftuvSUX nor --sort is specified.

    Mandatory arguments to long options are mandatory for
    short options too.

    -a, --all
        do not ignore entries starting with .
```

Ricerche nel file system

- `find` esegue una ricerca ricorsiva a partire da una directory specificata. È una utility molto potente (è possibile specificare nomi parziali, indicare le date entro cui effettuare la ricerca, ...).

```
$ ls
Lab04

$ find Lab04 -name 'es*'
Lab04/Es10/es10.go
Lab04/Es03/es3.go
Lab04/Es03/es3_bis.go
Lab04/Es15/es15.go
```

Redirezione a/da file

È possibile utilizzare dei file per:

- memorizzare l'output (stdout) di un comando in un file:
 - scrivendo dopo il comando il carattere di maggiore (>) seguito dal nome del file (il file viene sovrascritto);
 - scrivendo dopo il comando due caratteri di maggiore (>>) seguito dal nome del file (in questo caso l'output del comando aggiunto in coda al contenuto del file, che non viene quindi sovrascritto).
- leggere l'input (stdin) di un comando da file: scrivendo dopo il comando il carattere di minore (<) seguito dal nome del file.

Esempio

```
$ ls
Es01  Es02  Es03  Es04  Es05  Es06  Es07  Es08  Es09

$ ls > output

$ cat output
Es01
Es02
Es03
Es04
Es05
Es06
Es07
Es08
Es09
output

$ mail -s "Oggetto email" marco < corpo_email.txt
```

Completamento automatico

- Spesso non è necessario scrivere per esteso il nome di un file: basta
 - iniziare a scriverne il nome
 - premere il tasto di tabulazione (TAB)
- Se i caratteri scritti individuano uno e un solo file nella directory corrente, il suo nome viene automaticamente “completato” dalla shell.

Completamento automatico - Cont'd

- Se i caratteri inseriti non individuano un unico file la pressione di TAB non ha effetti visibili e viene emesso un suono.
- Premendo una seconda volta TAB si otterrà un elenco dei file compatibili con i caratteri specificati.
- Il completamento automatico funziona anche con i nomi di comandi della shell.

Compilare ed eseguire un programma Go

Sono disponibili diversi strumenti per formattare, compilare ed eseguire codice Go:

- `go fmt`: formatta il codice di un singolo file o di un intero package
- `go doc`: restituisce informazioni su un package
- `go run`: compila ed esegue un file
- `go build`: compila package creando un eseguibile
- `go help [comando]`: fornisce la documentazione per lo strumento `go` [comando]

Esempi - go fmt

```
$ ls  
main.go  
  
$ go fmt main.go  
main.go
```

```
package main  
import "fmt"  
func main() {  
    fmt.Println("Hello world!")  
}
```



```
package main  
  
import "fmt"  
  
func main() {  
    fmt.Println("Hello world!")  
}
```

Esempi - go doc

```
$ ls
main.go

$ go doc
Hello World Package

Un semplice Hello World in go
```

```
/*
Hello World Package

Un semplice Hello World in go
*/
package main

import "fmt"

func main() {
    fmt.Println(    "Hello world!")
}
```

Esempi - go run

```
$ ls
main.go

$ go run main.go
Hello world!
```

```
/*
Hello World Package

Un semplice Hello World in go
*/
package main

import "fmt"

func main() {
    fmt.Println("Hello world!")
}
```