Introduzione ai Sistemi Operativi

Mirko Cesarini - Dario Pescini
nome.cognome@unimib.it

Hardware e software in un elaboratore

- Dall'hardware . . . al software applicativo
  - Software applicativo: dedicato a soddisfare specifiche esigenze dell'utente
  - Software di sistema (o sw di base): focus sulla gestione delle risorse dell'elaboratore
    - Sistema operativo: si occupa della gestione dell'hardware e delle risorse condivise del computer
    - Applicazioni di sistema: programmi (simili ai programmi applicativi) che contribuiscono allo svolgimento delle attività del sistema operativo
  - Infrastruttura Hardware. L'hardware utilizzato per elaborare le informazioni

		Personal Computer	**Smartphone**
SW Applicativo		Posta Elettronica, Word Processor, . . .	Posta Elettronica, Word Processor, WhatsApp, . . .
SW di sistema	Applicazioni di sistema	Control Panel, Printer Manager, Network Manager, . . .	Network manager, WiFi Manager, Screen Brightness Manager . . .
	Sistema operativo	Windows, Linux, Mac OS	IOS, Android, Windows

Un po' di storia

- Epoca dei *software monolitici* (primi anni dell'informatica)
  - Si sviluppavano programmi che governavano tutti gli aspetti del computer, dalla gestione delle periferiche alla logica applicativa.
  - Non c'era separazione tra software applicativo e software per gestire i dettagli hardware
  - Problemi
    - Ogni software doveva essere sviluppato da da zero.
    - Cambiare un componente hardware richiede la modifica del software
- Separazione tra *sistema operativo* e *software applicativo* (anni successivi)
  - Il sistema operativo fornisce un'*interfaccia* che permette ai programmi applicativi di astrarre dall'hardware
  - Software applicativo, sviluppato a partire dall'astrazione fornita dal sistema operativo
  - In questo modo chi sviluppa sistemi operativi può concentrarsi sulla gestione dell'hardware e delle applicazioni, mentre chi sviluppa applicazioni può focalizzarsi sulle esigenze degli utenti

Esempio di interfaccia: il driver di periferiche

- Interfaccia: un punto di contatto e scambio di informazioni tra due componenti (hardware o software) diversi.
- Es. (hardware): la porta USB
- Es. (software): l'interfaccia tra un driver di periferica e il sistema operativo
  - Il sistema operativo per interagire con un componente hardware utilizza un *Driver*
  - Driver: software per gestire un componente hardware (es. ogni scheda grafica, mouse, . . . ha il suo driver specifico)
  - Driver di hardware simili (ma di produttori diversi) hanno un'*interfaccia* comune, ma implementazioni diverse
  - Il meccanismo dei driver permette al Sistema Operativo di astrarre dalle caratteristiche di un componente (es. la tastiera)
- Il sistema operativo fornisce ai programmi applicativi un'interfaccia con un alto livello di astrazione sull'hardware. Es., la print di python usa un servizio offerto dal sistema operativo

Compiti del Sistema Operativo

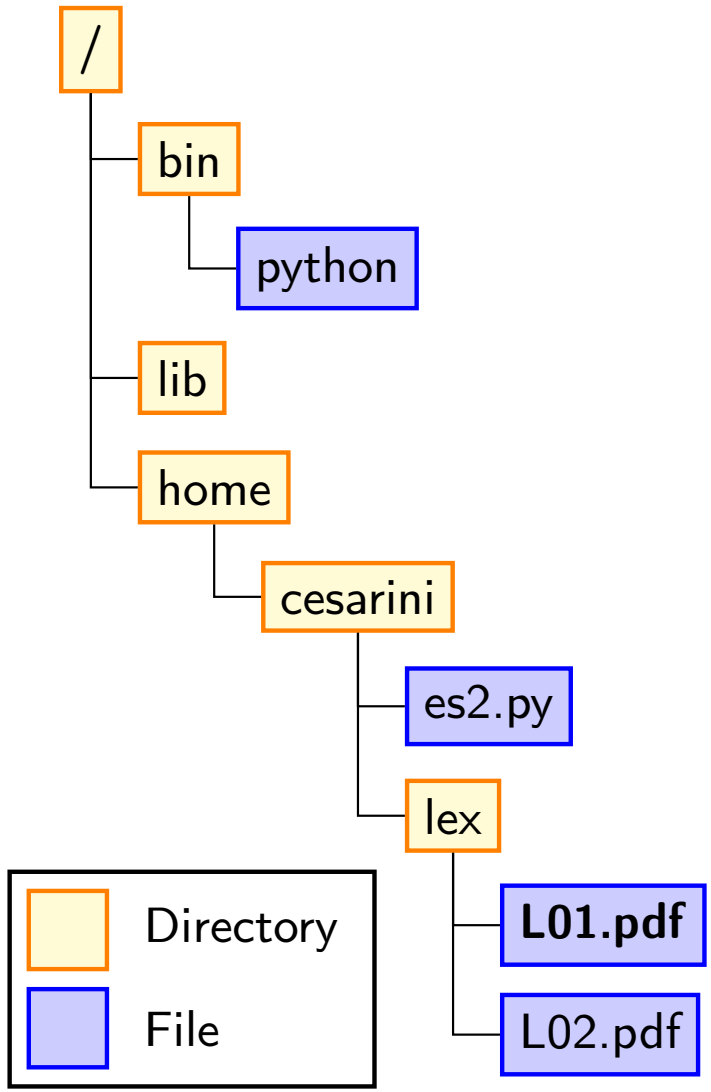
- Gestire avviamento e spegnimento del computer
- Gestire le risorse condivise (periferiche, memorie di archiviazione, . . .)
  - Disciplinare l'accesso alle risorse
  - Allocare le risorse ai diversi utilizzatori
- Esempi
  - Gestire la memoria di lavoro del computer
  - Gestire i processi in esecuzione
  - Gestire il File System
- Gestire l'interconnessione con altri computer (per realizzare sistemi distribuiti)
- Fornire una user interface (interfaccia utente) per l'interazione tra l'uomo e il computer
- . . .

E ora?

- Fra poco ci occuperemo
  - File System
  - User Inteface (in particolare la Command Line Interface)
- Gli altri elementi del sistema operativo li vedremo in una delle prossime lezioni

Cenni sul File System

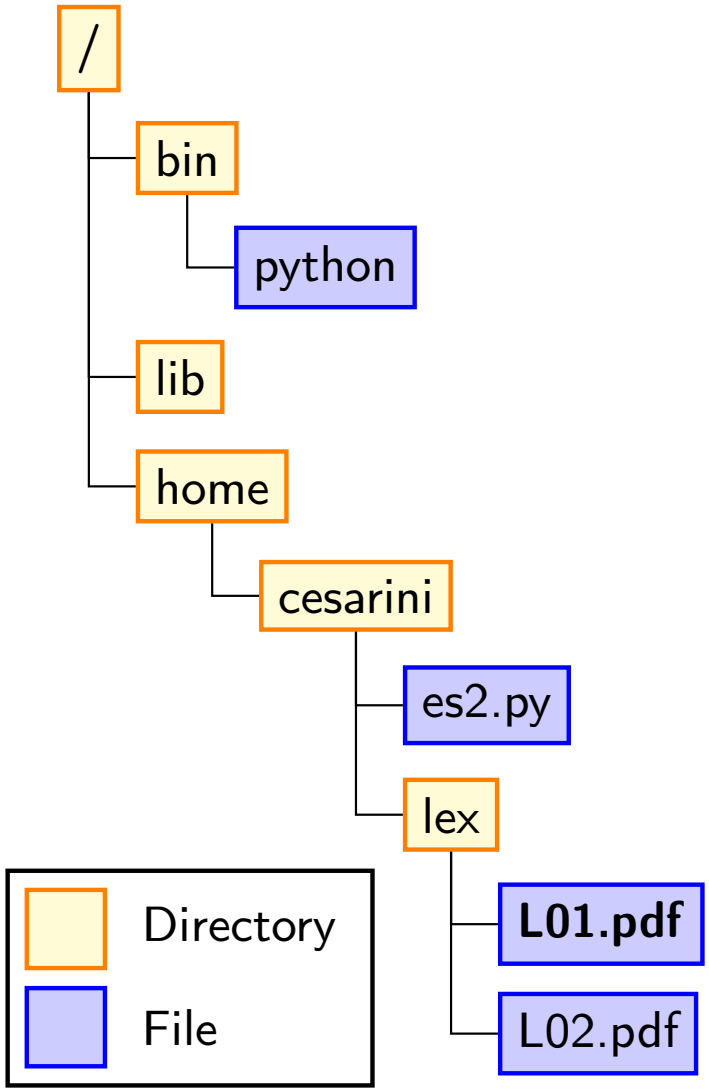
- File Systen: metodi e strutture dati che il sistema operativo usa per gestire i dati memorizzati su dischi o in generale su unità di archiviazione
- Due concetti fondamentali
  - File: un insieme di dati. Memorizzati tipicamente attraverso una sequenza di bit (i primi file sono stati memorizzati su nastri magnetici)
  - Directory: una struttura che raccoglie un insieme di file ed eventualmente (sotto) directory
- L'insieme dei file e directory è organizzato ad albero



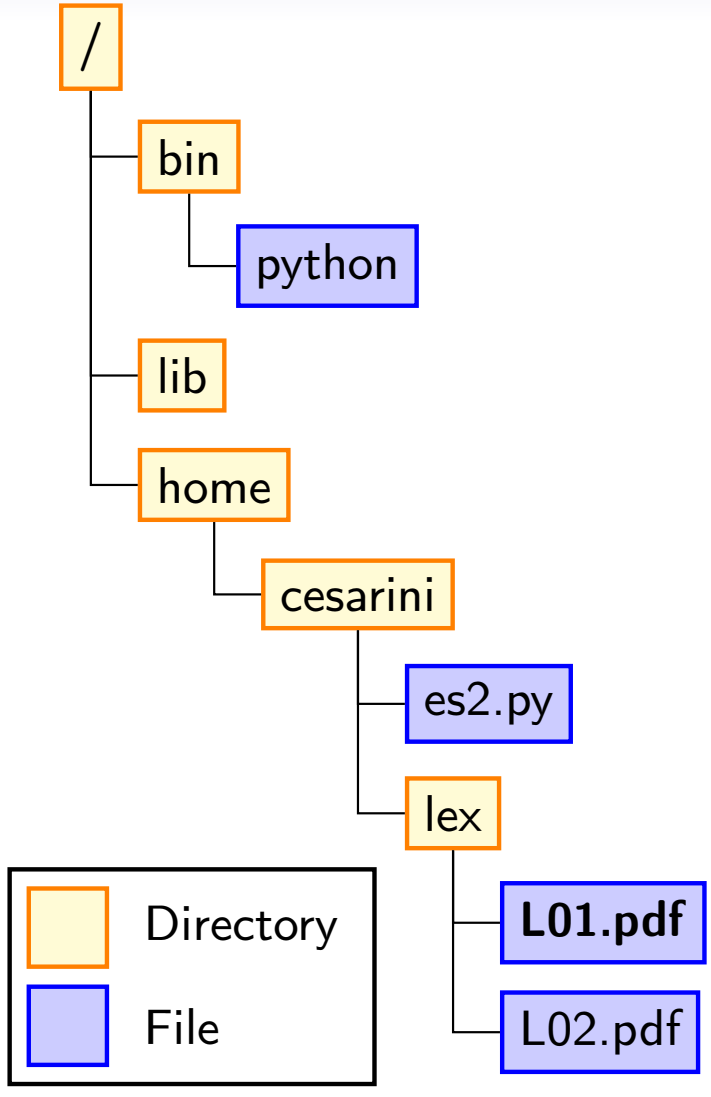
File system, terminologia

Dato il file L01.pdf nel file system a dx,

- L01.pdf è il **nome** del file (base name, o file name)
- pdf è l'**estensione** del file
- /home/cesarini/lex/L01.pdf è il **percorso completo** (*fully qualified name* o *full path*)
- /home/cesarini/lex è la **directory** che contiene il file
- **percorso relativo**
  - Supponiamo che /home/cesarini sia la **directory di lavoro** corrente
  - lex/L01.pdf è il **percorso relativo** di L01.pdf rispetto alla *directory di lavoro* (*relative path*)



- Nella slide precedente abbiamo introdotto i concetti di nome e percorso (relativo e assoluto) per un file
- In maniera analoga è possibile parlare di nome e percorso (. . .) per una directory
- Dato il percorso /home/cesarini/lex/L01.pdf
  - / è la **root**, la directory che contiene tutte le altre directory e gli altri file di un system. E' l'inizio di ogni percorso
  - i rimanenti / sono dei separatori, si tratta di un espediente per separare i nomi di file e directory tra loro
  - il simbolo / in entrambi i casi è chiamato *slash*



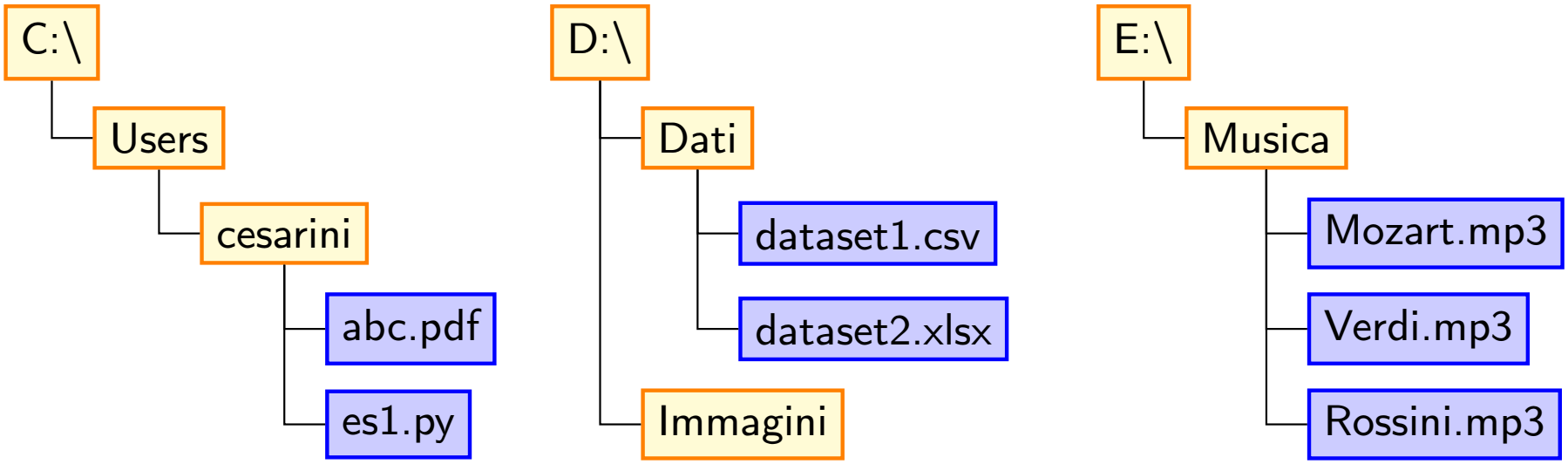
File Systems in Unix e Windows

Vediamo come la terminologia sui file system appena introdotta è declinata in due famiglie di sistemi operativi

OS	**Root Directory**	**Directory Separator**	**Examples**
Unix	/	/	/home/user/docs/Letter.txt
Win	[drive letter:]\	\	C:\User\MrX\Letter.txt \User\MrX\Letter.txt MrX\Letter.txt

- / è chiamato *slash*
- \è chiamato *back-slash*
- Le [ ] attorno a *drive letter:* significatno che l'indicazione del drive (disco) in Windows è opzionale
- Unix comprende Linux, Mac OS, ...
- Sia in Win sia in Unix sono definiti i percorsi sia assoluti sia relativi

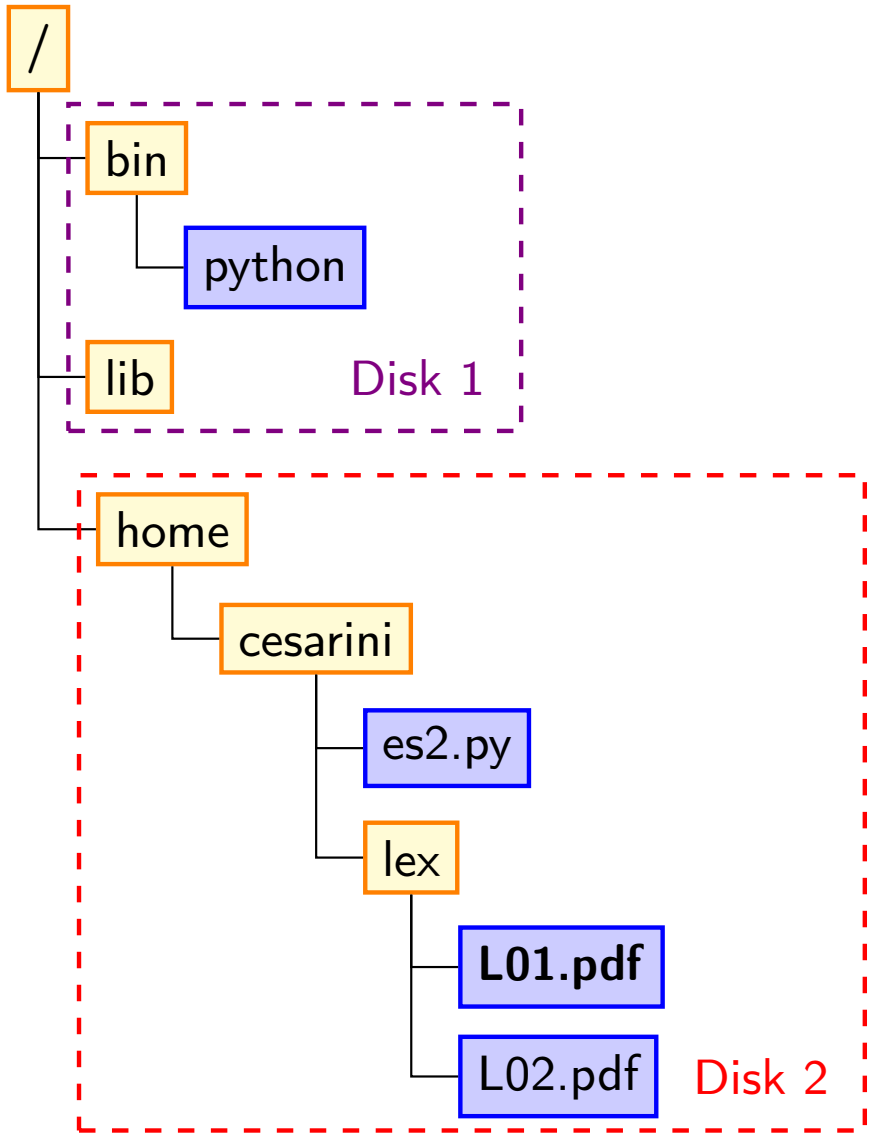
Windows, Dischi multipli e Foreste di Alberi



- I file system di Windows in realtà è una foresta di alberi
- Ogni drive (disco) è la radice di un albero separato dagli altri

Unix e dischi multipli

- Anche quando in un computer sono presenti più dischi, nel file system Unix apparentemente non c'è traccia dei diversi dischi
- In realtà i dischi sono *montati* sull'unico albero esistente
- Ciò rende possibile spostare gruppi di directory da un disco all'altro senza che l'utente se ne accorga



File, Directory e Metadati

Ad un file, oltre al contenuto, sono associate delle informazioni (chiamate metadati)

- Data e orario di *creazione* o *ultima modifica*
- Il proprietario e/o il creatore del file
- Permessi di accesso al file

Le stesse informazioni possono essere associate anche alle directory
Ecco un esempio di metadati in un file system Unix

dir	User Perm.	Group Perm.	Others Perm.	Owner Name	Group Name	Size	Last Modified	Name
-	rw-	r–	r–	cesarini	gmc	1.6M	Dec 11 14:50	L06.pdf
-	rw-	r–	r–	cesarini	gmc	28K	Dec 11 14:50	L06.tex
d	rwx	r-x	r-x	cesarini	staff	170B	Dec 10 17:59	img

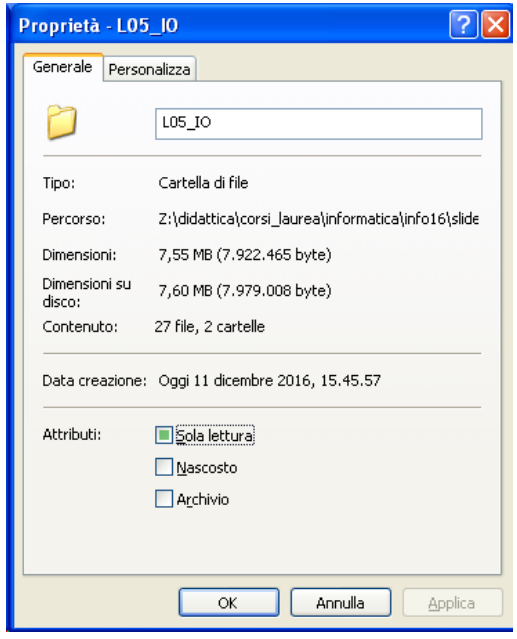
Alcuni metadati di un sistema windows

Informazioni ottenibili a riga di comando

Last Modified	dir	size	name
10/12/2016 17.59	<DIR>		img
11/12/2016 15.20		1.657.585	L06.pdf
11/12/2016 15.22		29.410	L06.tex

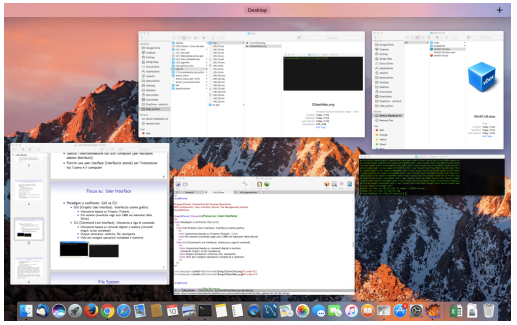
- Nei sistemi Windows l'estensione del file (es. .pdf, .tex) da informazioni su qual applicazione può aprire il file
- Cambiando l'estensione, si modifica l'applicazione associata al file
- Se sul vostro PC windows non vedete le estensioni . . . il vostro sistema operativo le sta mascherando (è l'impostazione di default)

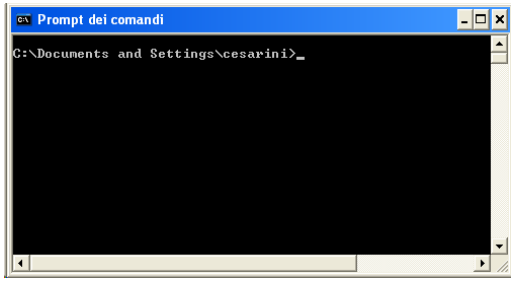
Informazioni ottenibili tramite GUI

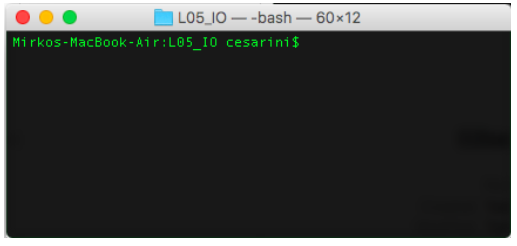


User Interface

- Paradigmi a confronto: GUI vs CLI
- GUI (Graphic User Interface). Interfaccia utente grafica.
  - Interazione basata su Finestre, Pulsanti, ...
  - Più recente (inventata negli anni 1980 nei laboratori PARC della Xerox)
- CLI (Command Line Interface). Interaccia a riga di comando.
  - Interazione basata su comandi digitati a tastiera (comandi singoli, script complessi)
  - Output attraverso: schermo, file, stampante
  - Utile per svolgere operazioni complesse e ripetitive





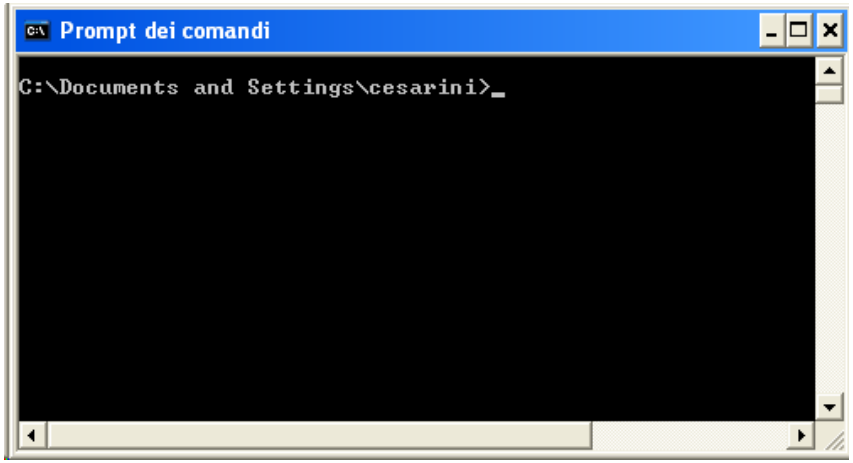


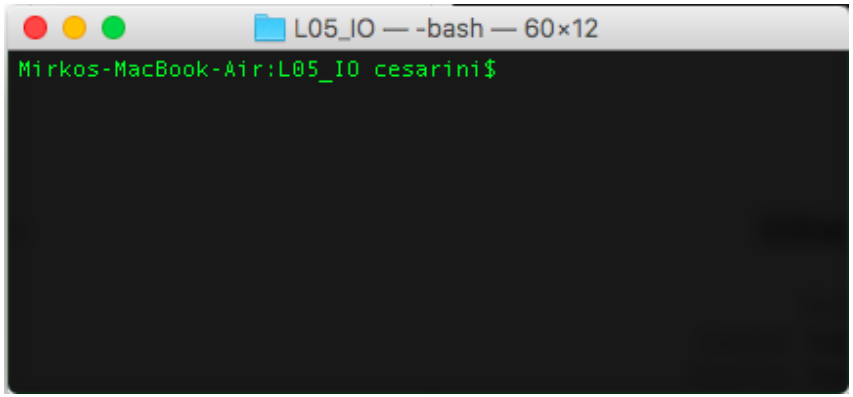
Shell CLI e File System

- Shell: strumento di interazione con il computer e il sistema operativo basato su un paradigma CLI (Command Line Interface)
- Perchè studiare le Shell in un'epoca di interfacce visuali?
- E' utile per svolgere attività ripetitive o complesse
- Es. per cancellare tutti i file che terminanon con .txt (100 file, sparsi tra altri)
  - GUI: devo trascinare nel cestino 100 file
  - CLI: basta un comando (ora *non* ve lo spiego)
- La gestione di grossi dataset può essere effettuata su macchine (server) molto grandi, alle quali ci si connette via Shell in remoto
- I linguaggi di programmazione
  - Il processo di compilazione può essere svolto tramite comandi dati a Shell
  - In questo corso vedremo l'interpretazione ed esecuzione del codice python, sia tramite Shell, sia tramite GUI

Shell CLI e File System

Questi sono due esempi di Shell (Win e Unix)





- Directory corrente (concetto molto importante): la directory nella quale si sta lavorando, in un certo istante
  - Qualsiasi riferimento a file (senza specificare un percorso) fa implicitamente riferimento a file della directory corrente
  - Se voglio utilizzare un file in una posizione diversa dalla directory corrente, dobbiamo fornire il percorso (assoluto o relativo)
  - I percorsi relativi, sono sempre calcolati a partire dalla directory corrente

Alcuni comandi di Shell

Descrizione	**Comando Win**	**Comando Unix**	**Note**
Cambia directory corrente	**cd** *percorso*	**cd** *percorso*	Il nome del comando è lo stesso in entrambi i SO
Scelta del disco su cui lavorare	lettera_disco:		Es., c:, d:, . . . , z:. Non si applica ai sistemi Unix
Per vedere il contenuto della directory corrente	**dir**	**ls**	
Visualizza la directory corrente	**cd**	**pwd**	In win, cd senza parametri visualizza la directory corrente
Per eseguire uno script python	**python** sc1.py	**python** sc1.py	il file sc1.py si deve trovare nella directory corrente
Per copiare un file	**copy** origine destinazione	**cp** origine destinazione	origine e destinazione sono i percorsi dei file corrispondenti
Per cancellare un file	**del** nomefile	**rm** nomefile	State attenti!!!

Eseguire un comando da Shell

- Il prompt della Shell attende un comando.
- Al prompt è possibile digitare il comando seguito da eventuali parametri. Premendo invio (il tasto return) il comando viene eseguito.
- Il prompt di Win e Unix sarà rappresentato rispettivamente da >e $qua di seguito

Win	**Unix**	Nota
>python es1.py	$python es1.py	Esecuzione di un file nella directory locale
>python scripts\leggi.py	$python scripts/leggi.py	parametro con percorso relativo
>python C:\users\es3.py	$python /home/es3.py	. . . percorso assoluto

- Un comando ha questo pattern:
  comando [parametro_1] [parametro_2]. . . [parametro N]
- tutto ciò che è compreso tra [e] è opzionale

Eseguire uno script python da Shell

- Con un editor di testi, salvate il file *primo.py* in una directory di vostra scelta. Nel file scrivete il testo
  print("Hello Word")
- eseguite i comandi seguenti (> e $ *non* vanno digitati)

Win	**Unix**	**Note**
Menù avvio / esegui / cmd	Applications / Utility / Terminal	Aprite la Shell
>c: ¶ >cd \users\cesarini ¶	$cd /home/cesarini/ ¶	¶: punto in cui premere return
>python primo.py ¶	$python primo.py ¶	Osservate il risultato
>copy primo.py secondo.py ¶	$cp primo.py secondo.py ¶	creo nella directory corrente il file secondo.py, copia di . . .
>del secondo.py ¶	$rm secondo.py ¶	Cancello il file appena creato

- Al posto di *\users\cesarini* o */home/cesarini* digitate il percorso della directory in cui avete salvato il file

Alcuni trucchi

- Nell'usare la Shell, occorre spesso ripetere l'esecuzione dei comandi
  - Utilizzate i tasti freccia, $\uparrow$ e $\downarrow$, ...
  - ... per recuperare i comandi dati in precedenza
  - Questa funzionalità si chiama *history*
- Scrivere i percorsi dei file può essere noioso, soprattutto se i percorsi sono lunghi, come nell'esempio seguente: /Users/cesarini/documenti/didattica/corsi_laurea/informatica/info16/slide_python/
  - Appena scritte le prime lettere del nome di una directory ...
  - ... provate a premere (il tasto) tab, anche diverse volte ...
  - ... e osservate cosa succede.
  - Questa funzionalità si chiama *autocompletamento*