Prima Esercitazione

GNU/Linux e sviluppo in C

Andrea Reale
andrea.reale@unibo.it

Agenda

- Panoramica sui sistemi operativi GNU/Linux
- Strumenti di Amministrazione
  - Shell dei comandi
- Tool di sviluppo per il linguaggio C
  - Editor di testo: **vim**
  - Compilatore e debugger: **gcc** e **gdb**
- Esercitazione in C

GNU/Linux

Introduzione

UNIX e GNU/Linux

- **UNIX**: sviluppato negli anni '60-'70 presso gli AT&T Bell Labs
  - **UNIX certified**: sistemi fully-compliant (Solaris 10, IBM AIX, Mac OS X 10.5, ...)
  - **Unix-like**: sistemi NON fully-compliant (GNU/Linux, BSD, ...)

- **GNU Project: G**nu's **N**ot **U**nix
  - Progetto che mira alla creazione di un sistema operativo **libero Unix-like**

Originariamente kernel **Hurd**, adozione del **kernel Linux** nel '92

- **Linux**:
  - Kernel rilasciato nel 1991 ad opera dello studente svedese Linus Torvalds
  - Adozione ed integrazione del kernel Linux con i tool e le librerie *user-space* GNU

Distribuzioni GNU/Linux

- Distribuzione (e.g. Debian, Ubuntu, Gentoo, RedHat/Fedora, ...) : collezione di **kernel**, **tool** di basso livello, e **software applicativo**
  - strumenti per l'installazione, la configurazione automatica e la gestione del sistema

- Concetto di pacchetto software
  - archivio contenente software (codice o eseguibili) insieme a metadati che li descrivano (dipendenze, path di installazione)
  - usati per automatizzare e semplificare l'installazione di applicazioni

- Gestore di pacchetti:
  - Tool applicativo per gestire le operazioni di installazione, rimozione, e aggiornamento di pacchetti software
  - Si occupa di risolvere e gestire dipendenze fra pacchetti
  - differente per *famiglie* di distribuzione (APT, RPM, Portage, ...)

Ai fini del corso...

- **Installazione** di una distribuzione Linux su una macchina fisica
  - **maggiore apprendimento**
  - complessità e problematiche maggiori (partizionamento del disco, eventuale dual boot, ...)

- Uso distribuzioni **Linux Live CD**
  - nessuna installazione, ambiente di lavoro caricato in RAM e *ripetibile*
  - maggiori requisiti hardware, prestazioni penalizzate

- **Virtualizzazione**
  - installazione vera e propria su una *macchina virtuale*
  - priva di implicazioni per la macchina fisica
  - prestazioni penalizzate

Utenti e gruppi

- Sistema **multiutente**
  - per utilizzare il sistema bisogna possedere un **account** utente
  - necessità di proteggere le informazioni
- Concetto di **gruppo**
  - es. staff, users, students
  - Utenti di uno stesso gruppo possono lavorare agli stessi documenti
- Ogni utente appartiene ad **almeno un gruppo**

Accesso a Linux: login

- Per iniziare una **sessione** bisogna autenticarsi con il proprio **account**
  - username: (es. s0000180286, dll28493, ...)
  - password: (es. dfh@2#q, **a890, aPP&x., ...)
- **NOTA**: maiuscole / minuscole sono caratteri diversi!! (la password **a890 è diversa da **A890)
- **Accesso al sistema:**
  - **login**: s0000180286
  - **password**: ******

Esercizio - Login

Eseguire il login fornendo username e password

- **username**: calcolato in base alla matricola (es. s0000<matricola>)
- **password**: il vostro pin

Gestione utenti

- La maggiorparte delle operazioni normalmente eseguite (editing di testo, web browsing, ecc...) non richiede particolari privilegi
- Operazioni di amministrazione (configurazione hw, manutenzione e gestione del sofware) possono essere eseguite solo da **super-utente** (utente **root**)
- Diritti di scrittura/lettura/esecuzione su file e direttori critici per la sicurezza/stabilità del sistema
- E' buona norma limitare l'utilizzo dell'account root per compiere le sole operazioni che lo richiedono, utilizzando un utente "normale" per le altre
- In alcune distribuzioni l'utente root e` disabilitato
  - comando **sudo** per acquisire **temporaneamente** i privilegi di super-user

Shell (1/2)

- Una volta superata la fase di login, l'utente è collegato al sistema Linux.
  - Di norma è presente una finestra di **shell**

- La shell è l'**interprete del linguaggio comandi**
  - E' un **programma** che consente di far interagire l'utente col sistema.
  - Resta in attesa di comandi (da digitare con la tastiera), che manderà in esecuzione una volta ricevuto l'<ENTER>

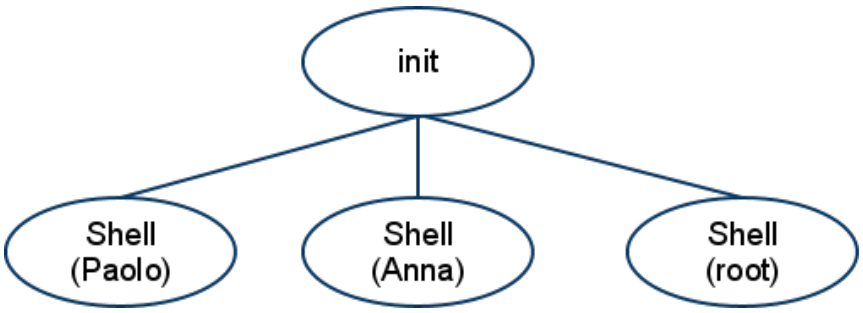
Shell (2/2)

- Esistono diverse Shell in Unix:
  - Bourne Shell (standard)
  - C Shell
  - Korn Shell
  - Tc Shell
  - etc

L'implementazione della Bourne shell sotto Linux si chiama **bash** (Bourne-Again shell)

Shell - Accesso al sistema

- Un utente può attivare più shell, anche diverse (tcsh, csh, bash, ...)
- Una shell in particolare è chiamata **shell di login**
  - quella per cui viene chiesta inizialmente la password
- Una shell di login per ciascun utente collegato al sistema
  - la shell è rappresentata da un **processo** assegnato all'utente



Uscita dalla shell

- Per uscire dal ciclo di una shell di login si può:
  - usare il comando **logout**, oppure
  - digitare **CTRL+D** (carattere di *end-of-file*)

- Una volta effettuato il logout, per riprendere a usare linux bisogna inserire nuovamente username e password (login)

- Per uscire da una shell *anche non di login* esiste il comando **exit**.

Comandi della shell di Unix

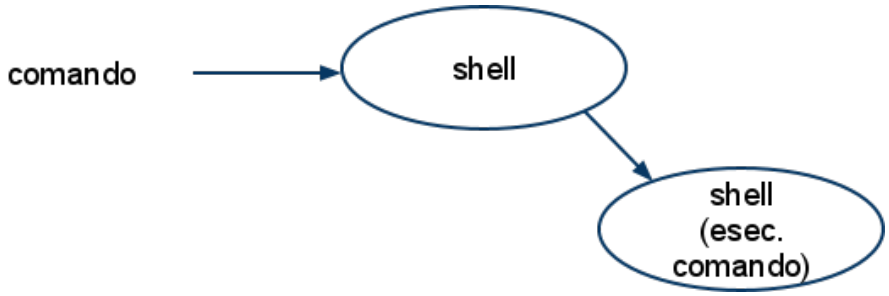
std input, std output, std error
tipi di comandi

Comandi Shell

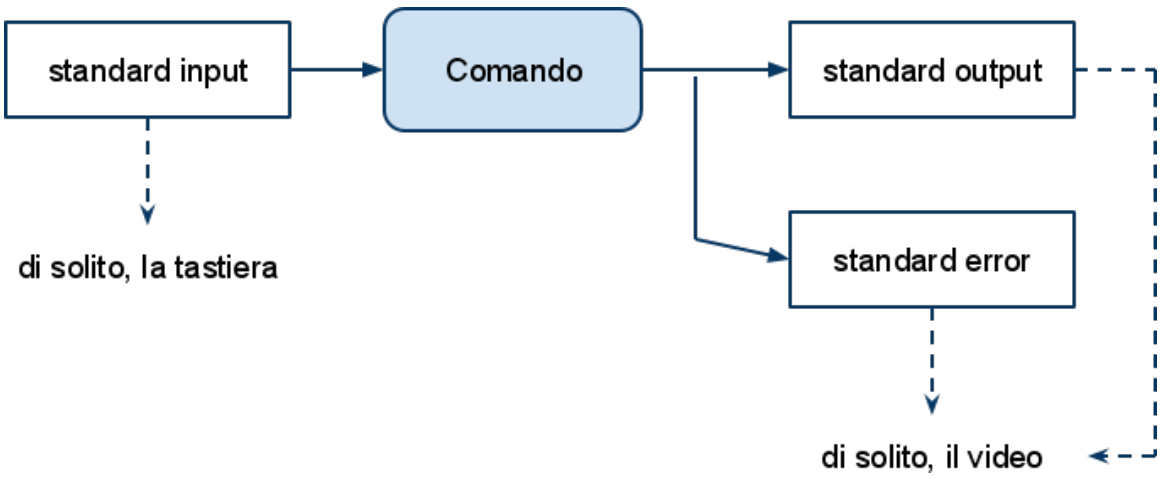
- Ogni comando richiede al kernel l'esecuzione di una particolare azione
- I comandi esistono nel file system come file binari, generalmente eseguibili da tutti gli utenti
  - direttorio /bin o /usr/bin
- Possibilità di realizzare **nuovi comandi**
  - programmazione in shell

Esecuzione dei comandi

- Per ogni comando da eseguire lo shell crea uno **shell figlio**, dedicato all'esecuzione del comando:



Input / Output di un comando



Esempi di comandi

pippo@lab3-linux:~$ date
Wed Apr 27 21:48:24 CEST 2005

pippo@lab3-linux:~$ who (connected users info)
root pst/3 Apr 9 14:02
root pst/4 Apr 22 17:11 (:0.0)
paolo pst/12 Apr 27 12:21 (deis32...

pippo@lab3-linux:~$ whoami
paolo pst/12 Apr 27 12:21 (deis32...

File System

Struttura logica, tipi di file,
percorsi assoluti e relativi,
comandi shell

File

- Logicamente, un **file** è una sequenza di bit, **a cui viene dato un nome**

- E' una **astrazione** molto potente

Consente di trattare allo stesso modo **entità fisicamente diverse,** come:

  - file di testo
  - dischi rigidi
  - stampanti
  - direttori
  - soft link
  - la tastiera
  - il video
  - etc

Tipi di file

- Diversi tipi di file
  - **Ordinari**: archivi di dati, testi, comandi, programmi sorgente, eseguibili, …
  - **Directory**: file gestiti direttamente solo dal S.O., che contengono riferimenti ad altri file
  - **Speciali**: dispositivi hardware, memoria centrale, hard disk, ...
  - **FIFO (pipe)**: file per la comunicazione tra processi
  - **Soft link**: riferimenti (puntatori) ad altri file o direttori

File - Nomi

- È possibile nominare un file con una **qualsiasi sequenza di caratteri** (max. 255)
  - a eccezione di **'.'** e '..' (sono nomi che hanno un significato particolare)
  - È sconsigliabile utilizzare per il nome di file dei caratteri speciali, ad es. metacaratteri e segni di punteggiatura

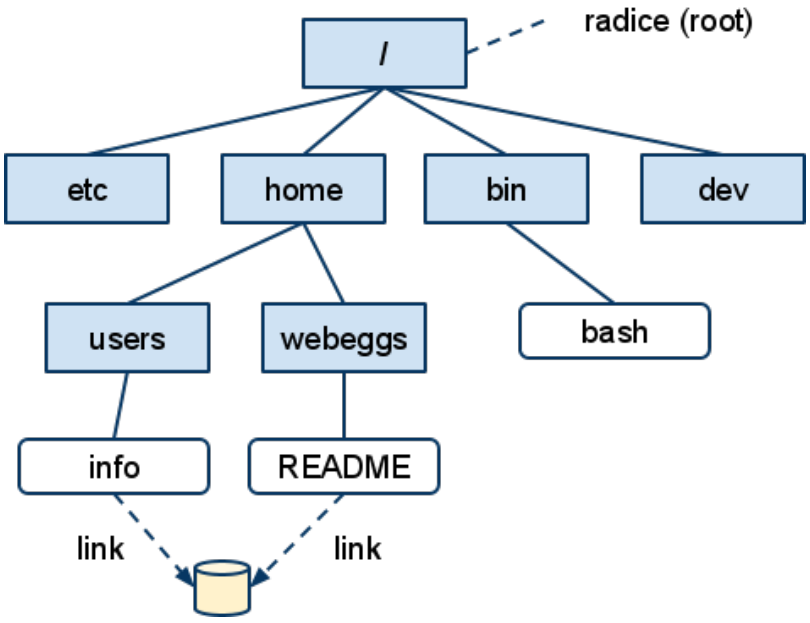
- Ad ogni file possono essere associati uno o più nomi simbolici (**link**)

MA

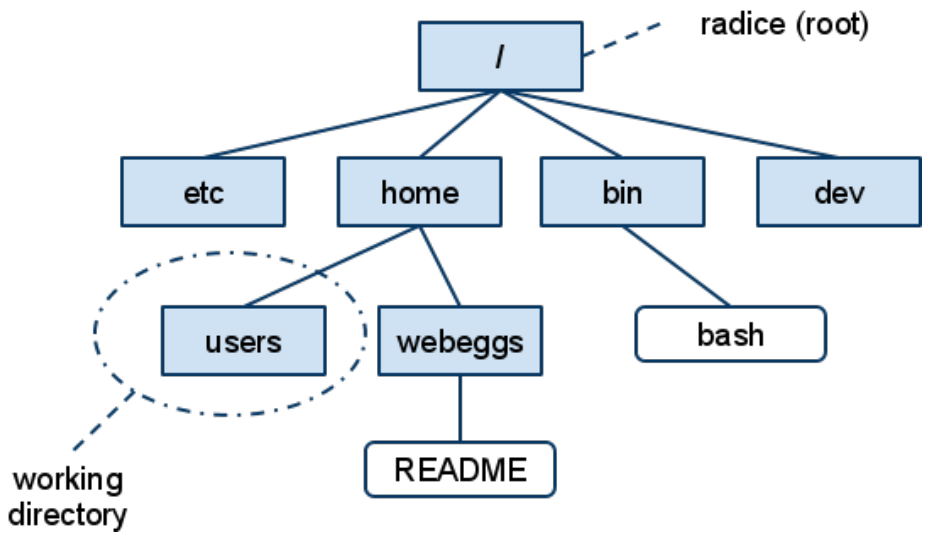
- Ad ogni file è associato *uno ed un solo descrittore* (i-node) identificato da un **intero** (i-number)

File System - Gerarchia

- Organizzazione a **grafo diretto aciclico (DAG)**



Nomi di percorsi (path names)



nome assoluto: /home/users/webeggs/README
nome relativo: ../webeggs/README

Gerarchie di directory

- Al login un utente comincia ad operare da una particolare directory, la sua ***home directory***
- Ci si può muovere attraverso la gerarchia dei direttori utilizzando appositi comandi shell
  - **pwd**: stampa il path (percorso) della directory di lavoropartendo dalla radice
  - **cd <dir>**: cambia la directory di lavoro corrente in dir
  - **cd ..** : sposta la directory di lavoro al genitore della directory corrente
  - **mkdir <dir>**: crea una nuova directory nel percorso specificato da dir
  - **rmdir <dir>**: rimuove la directory al percorso specificato da dir
- Le directory '.' e '..' (prendi in ogni directory del filesystem) hanno un significato speciale
  - **'.'** indica la directory corrente (es. cd. non fa cambiare la wd)
  - **'..'** indica la directory padre di quella corrente

Comando ls

- **ls <dir>**: lista il contenuto della directory passata come argomento
  - se senza argomenti lista il contenuto della directory corrente

- Numerosissime opzioni per configuare la visualizzazione della lista
  - es. **ls -l** per stampare la versione "lunga" (non solo il nome del file ma anche il proprietario, i permessi, ecc..)

Comando man

- **man pages**: pagine di manuale di sistema
  - contengono informazioni sul funzionamento dei comandi di shell e delle relative opzioni
  - anche descrizione di chiamate a ***system call*** (utili nelle prossime esercitazioni)

- **man <comando>**: visualizza la pagina di manuale relativa al comando passato come argomento
  - es. **man ls**

- Tasto **'q'** per uscire dalla pagina di manuale

Esercizi - comandi shell

1. Visualizzare il direttorio corrente con il comando pwd

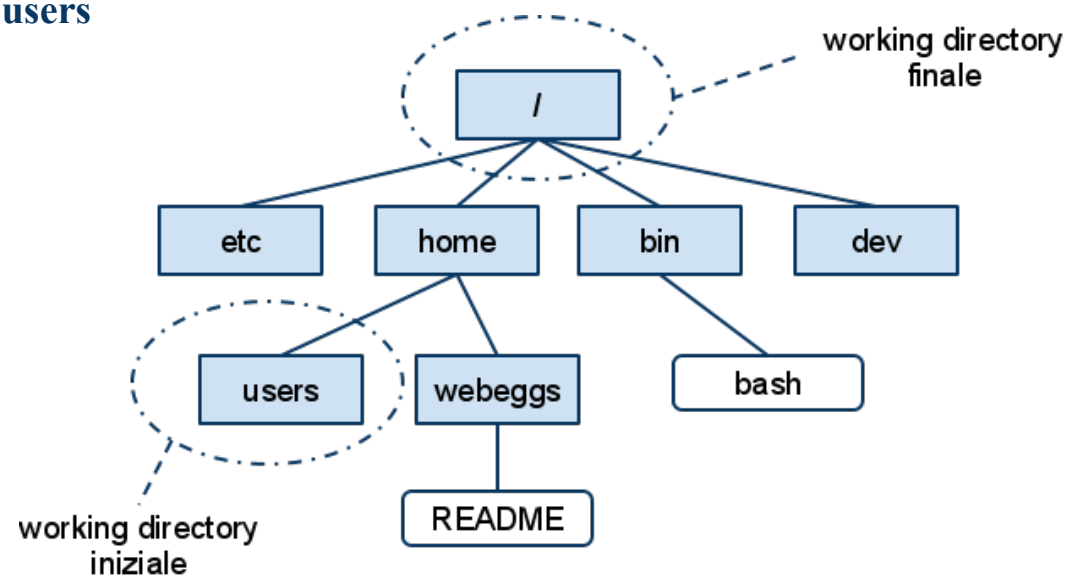
2. Provare il comando ls:
   - $ ls
   - $ ls –l
   - $ ls –la
   - ls a*

3. Provare il comando man:
   - man
   - man ls
   - man man

Il comando cd

- **cd** modifica il direttorio corrente
- Es.

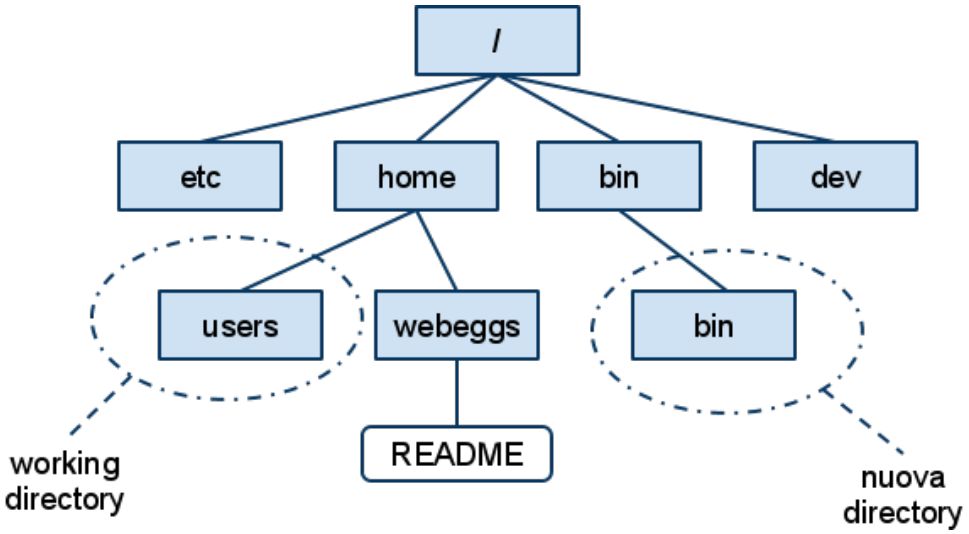
$ pwd
/home/users
$ cd /
$ pwd
/



Il comando mkdir

- **mkdir** crea un nuovo direttorio
- Es.

$ mkdir /bin/binequivalente a $ mkdir ../../bin/bin



Cancellazione, copia o spostamento di file

- E' necessario avere i diritti di scrittura sul file per modificarlo o eliminarlo

- Eliminazione di un file:
  - rm <nomefile>
- Copia di un file (e diritti):
  - cp <nomefile> <nuovofile>
- Spostamento di un file (e diritti):
  - mv <nomefile> <nuovofile>

Lettura di un file di testo

- E' necessario avere i diritti di lettura per visualizzare il contenuto di un file di testo
  - **cat <nomefile>**...: visualizza l'intero file
  - **less <nomefile>**...: visualizza per videate

- Altri comandi:
  - **grep <stringa> [<nomefile>...]** (ricerca di una stringa in un file),
  - **wc [-lwc] [<nomefile>...]** (conteggio di righe / parole / caratteri)

Esercizio - Gestione dei file

1. Verificare il contenuto della propria home directory
2. Spostarsi nella directory superiore e visualizzarne il percorso
3. Creare il file "pippo" nella propria home (ad esempio con: echo ciao! > pippo)
4. Leggere il contenuto del file pippo
5. Creare una nuova directory di nome "esercizi" all'interno della home
6. Entrare nella directory "esercizi".
7. Spostare il file "pippo" nella directory esercizi.

Comando **ps**

- **ps**: stampa lo stato dei processi nel sistema
  - numerose opzioni per controllare cosa stampare (man ps)
- Es. ps aux stampa informazioni sui tutti i processi del sistema riportandone anche l'utente proprietario.
- **man ps** per altre opzioni

Strumenti di sviluppo

editor, compilatore, e debugger

Vim - Introduzione

- **Vi**: editor di testo "standard" in sistemi Unix (1976)
- **Vim**: Vi IMproved, versione estesa dell'editor vi
  - oggi e` presente (o installabile tramite package manager) in tutte le distribuzioni di GNU/Linux

- Nella versione base, **interfaccia testuale**
  - Non ha bisogno di un'interfaccia grafica a finestre per essere usato
  - Utile, ad esempio, per l'amministrazione di macchine remote

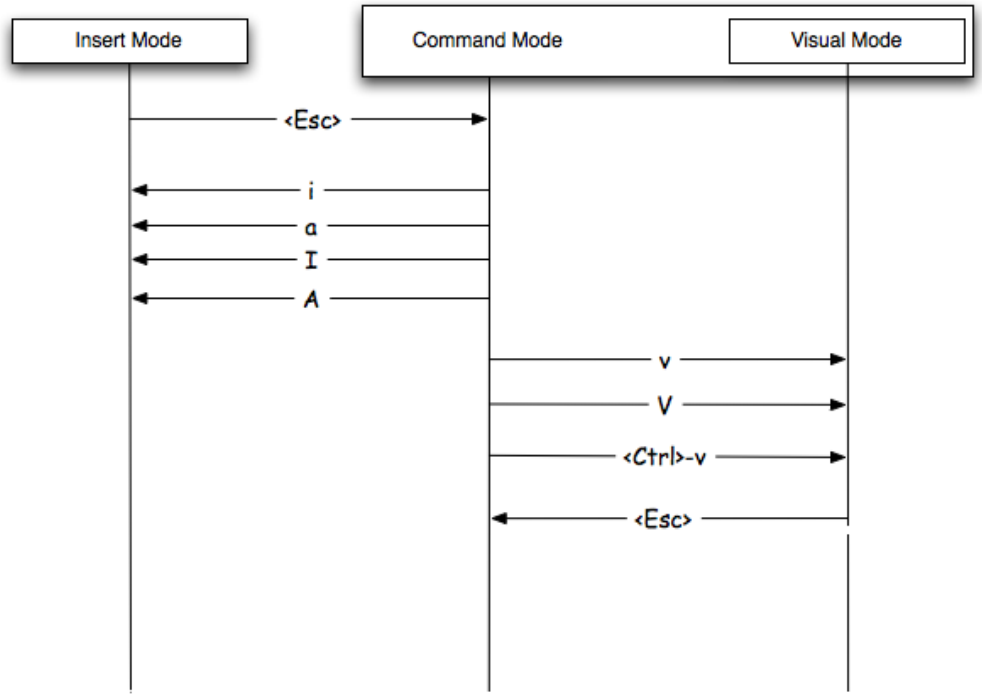
- Editor "modale", puo` essere usato in una delle seguenti modalità:
  - Insert-mode: i tasti premuti vengono inseriti come testo nel documento.
  - Command-mode: i tasti premuti vengono interpretati come comandi per l'editor (es. copia di testo, eliminazione di righe, ecc)
  - Visual-mode: simile alla modalità comandi, aiuta a selezionare visualmente parti di testo.

Vim - Introduzione

- Passaggio da una modalità all'altra
  - All'apertura, modalità comandi
  - Si passa alla modalità inserimento premendo il tasto `i` (inserisci prima del cursore), `a` ("append" dopo il cursore), oppure `R` ("replace" dal cursore in poi)
  - Si torna alla modalità comando con il tasto `<ESC>`

- In modalità comandi:
  - ci si può muovere nel testo con i tasti cursore, oppure con i tasti `h` (sinistra), `j` (giù), `k` (su), `l` (destra)
  - si possono dare numerosi altri comandi per muoversi rapidamente nel testo e per modificare parti dello stesso

Vim - Cambio di modalita`



Vim - Modalità comandi (tasto <ESC>)

- Salvataggio file
  - **:w** : salva il file
  - **:q** : esci da vim
  - **:wq** : salva ed esci da vim
  - **:q!** : esci da vim senza salvare
- Movimento avanzato
  - **w** : vai all'inizio della prossima parola
  - **b** : vai all'inizio della parola precedente
  - **^** : vai all'inizio della linea
  - **$** : vai alla fine della linea
- Editing
  - **dw** : elimina (e copia) la prossima parola
  - **dNw** : elimina (e copia) le prossime N parole (es. d2w)
  - **dd** : elimina (e copia) la linea corrente (es dd oppure 3dd)
  - **yw**, **yNw** : copia la prossima parola (le prossime N parole)
  - **p**, **P** : incolla l'oggetto copiato dopo (prima) del cursore

Vim - Altri comandi utili

- **/ <string>** : cerca all'interno del testo (poi, n per continuare a cercare in avanti)
- **]]** : in C, muoviti alla prossima funzione
- **K** : visualizza la pagina di manuale (man page) della funzione sotto il cursore
- **:!<comando>** : esegui il comando a livello di SO (es. :! ls -l)
- **:help <comando>** : visualizza la descrizione di un comando

- VimTutor: tutorial di 30 minuti su quanto detto (e molto di piu), avviabile utilizzando il comando **vimtutor**

GCC - GNU C Compiler (1/2)

- Compilatore C standard in sistemi Unix-like

- Dato un file sorgente (es. hello_world.c) preprocessa, compila ed esegue il linking, producendo un eseguibile

- Es. **gcc hello_world.c**

produce l'eseugibile con il nome default a.out

- Opzione -o: si specifica il nome desiderato per l'eseguibile

gcc -o hello_w hello_world.c # produce hello_w

- Opzione -c: non viene effettuato il linking, e viene prodotto solo il modulo oggetto
  - **gcc -c hello_world.c** # produce hello_w.o

GCC - GNU C Compiler (2/2)

- Opzione -W all: si attiva la stampa di tutti i messaggi di warning durante la compilazione. Utile per rivelare possibili errori di programmazione.
  - **gcc -Wall -o hello_w hello_world.c**
- Opzione -g gdb: include i simboli di debug, permettendo il debug del programma con gdb o altri debugger.
  - **gcc -Wall -ggdb -o hello_w hello_world.c**

GDB - The GNU Debugger

- **Debugger** standard usabile da terminale a caratteri
- Dato un file eseguibile hello_w avvia una sessione **interattiva** di debug
  - Es. **gdb hello_w**
- Opzione -d: specifica la directory in cui vanno cercati i sorgenti del programma
  - Es. **gdb -d src/ hello_w**
- **man gdb** o comando **help** durante una sessione di debug per i comandi basilari.

Esercitazione C

Gestione parametri d'ingresso e
manipolazione di stringhe

Obiettivi

- Programmazione C
  - Gestione dei parametri d'ingresso
    - argomenti **argc** ed **argv**
    - controllo correttezza parametri in ingresso
  - Gestione delle stringhe
    - libreria **string.h**

Esercitazione 1 - Testo (1/2)

Si realizzi un programma C che abbia un'interfaccia del tipo

lista_treni treno1 ... trenoN

e che prenda in ingresso un numero arbitrario di stringhe rappresentanti il codice del treno, ciascuna nel formato

<TIPOTRENO><NUMEROTRENO>

- <TIPOTRENO>: stringa di 2 caratteri che rappresenta il tipo di treno (si assuma, per semplicità che esistano solo tre tipi di treno, identificati rispettivamente da "ES", "IC" e "RG"
- <NUMEROTRENO>: identificativo univoco del treno, composto esattamente da 4 cifre.

Esercitazione 1 - Testo (2/2)

Il programma deve:

- Controllato che sia stato passato **almeno un treno**
- Controllare che ogni codice passato sia **conforme alle caratteristiche** indicate (in particolar modo, rispetti la lunghezza di 6 caratteri)
- **Stampare** a video i soli identificativi, raggruppati per tipologia di treno

Suggerimento:

- **man 3 string** : apre la pagina di manuale relativa alla libreria string.h

Esercitazione 1 - Esempio

$ lista_treni RG4556 ES2456 RG1802 ES2132

IC inseriti:
ES inseriti:
- ***) 2456**
- ***) 2132**

RG inseriti:
- ***) 4556**
- ***) 1802**