

FONDAMENTI DI PROGRAMMAZIONE

INTRODUZIONE ALLA RIGA DI COMANDO

Dott. Federico Concone

federico.concone@unipa.it

Informazioni utili

- Ricevimento:
 - Edificio 6, 3° Piano, Laboratorio Intelligenza Artificiale e Sistemi Distribuiti;
- Orario ricevimento:
 - Da concordare tramite e-mail;
- Testo consigliato:
 - Deitel, Deitel, Il linguaggio C – Fondamenti e tecniche di programmazione 8 Ed., Pearson, Italia, 2016
- Link materiale:
 - goo.gl/mrcWL7

Sommario

- Linux
- File system
- File e directory
- Shell
- Comandi Unix
- Consigli per navigare fra le directory
- Primo programma C
- Esercizi

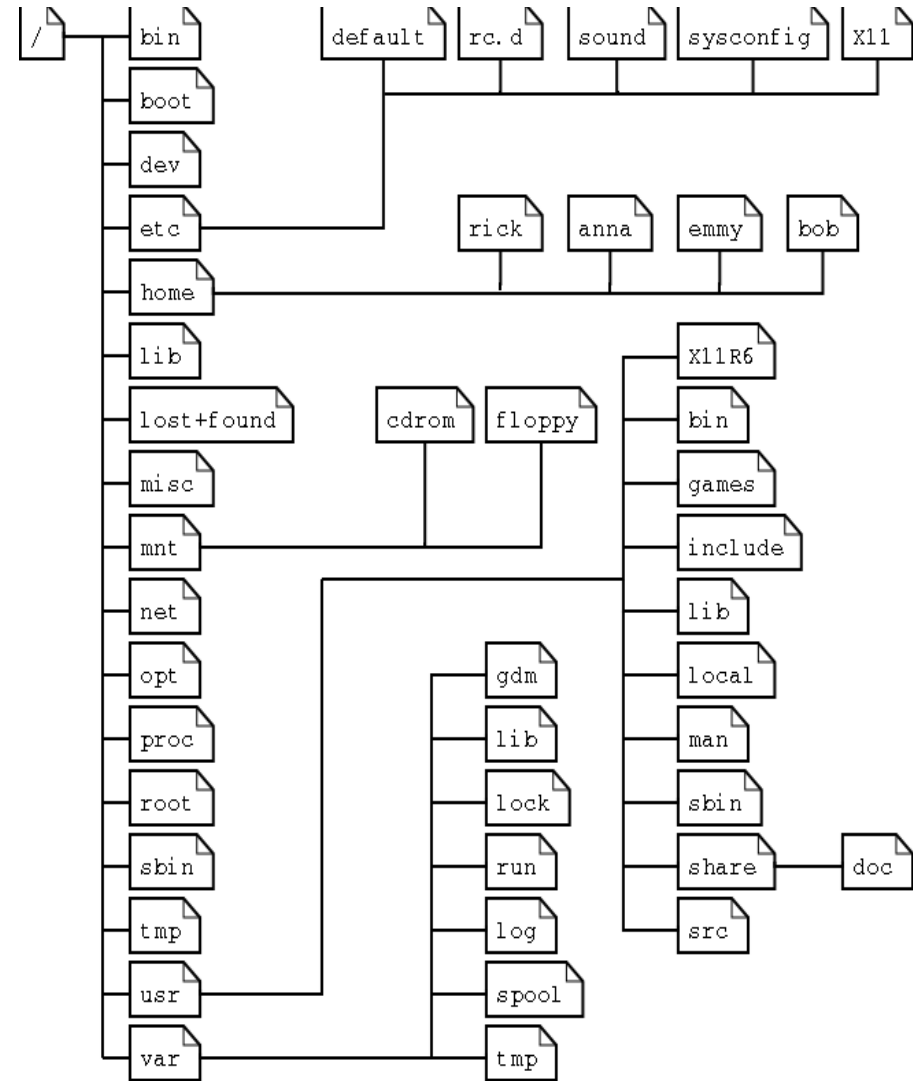
Sistemi Unix-Like

- I sistemi Unix-Like sono una famiglia di sistemi operativi basati su Unix;
- Caratteristiche principali:
 - sistemi *multiutente*;
 - *sicuri*;
 - *Open Source*;
 - continuamente *aggiornati*;
 - ...
- Esistono diverse distribuzioni ognuna delle quali ha caratteristiche differenti;



File system

- Il *file system* è il meccanismo fornito dal sistema operativo che **regola l'organizzazione fisica e logica** delle informazioni sui dispositivi;
- Ogni file system per potere essere utilizzato deve essere "*montato*", ovvero occorre collegarlo all'albero del file system in un preciso *mountpoint*;
- Il livello più alto del file system è la *directory root* (indicata con *"/"*);

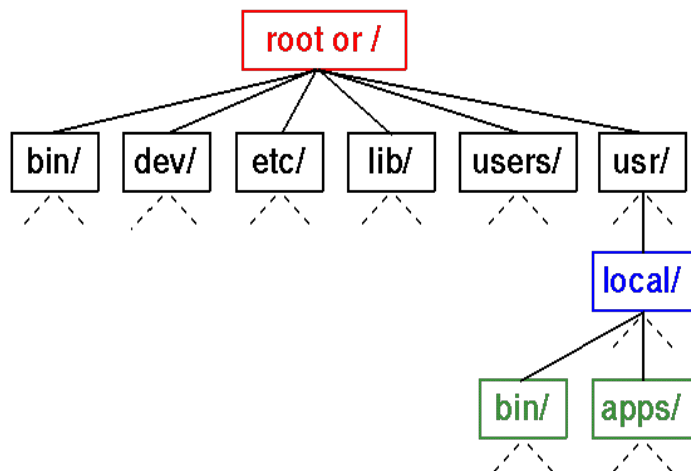


File system

- Directory comuni a tutte le distribuzioni Linux:
 - **/bin**: applicazioni binarie importanti;
 - **/boot**: file di configurazione sul boot;
 - **/etc**: file di configurazione, script di avvio, etc...;
 - **/home**: directory home degli utenti locali;
 - **/lib**: librerie di sistema;
 - **/mnt**: filesystem montati;
 - **/root**: home directory dell'utente root;
 - **/tmp**: file temporanei;
 - **/usr**: file e applicazioni che sono per la maggior parte disponibili a tutti gli utenti (users);
 - **/var**: file variabili come log e database;
 - ...

File e directory

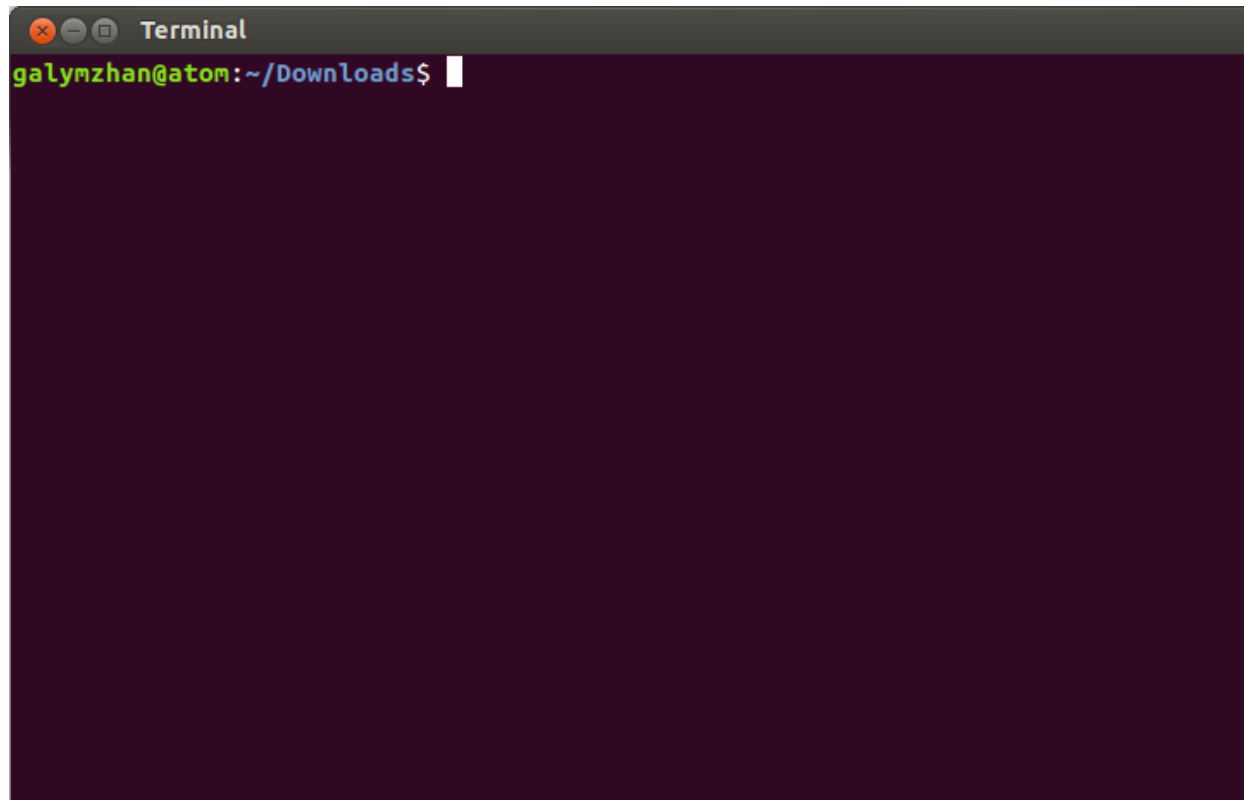
- Un **file** è una **sequenza non strutturata di bytes** (unità logica di memorizzazione) ed ha diversi attributi (tipo, permessi, dimensione, data di creazione ...)
- Una **directory** è un **file** che indicizza altri file.
- Il **path name** indica la posizione specifica di un file o di una directory e può essere:
 - *assoluto*: riferito alla radice della gerarchia (/);
 - *relativo*: riferito alla posizione dell'utente nel file system.



- **Assoluto** → **/usr/local/bin/my.txt**
- **Relativo** → **../bin/my.txt**

Shell

- La **shell** testuale (o interprete dei comandi) è un programma dotato di un'**interfaccia a riga di comando**;
- La **shell** permette all'utente di immettere dei comandi testuali, che vengono interpretati ed eseguiti dal sistema;



Comandi Unix

- I *comandi esistono nel file system come file binari*, generalmente eseguibili da tutti gli utenti;
- La sintassi tipica dei comandi UNIX è la seguente:

comando <opzioni> <argomenti>

- ***<opzioni>*** sono facoltative e influiscono sul funzionamento del comando. Generalmente consistono nel simbolo del “-” seguito da una sola lettera;
- ***<argomenti>*** si possono avere più argomenti o anche nessuno in base al comando.

Comandi Unix

Cosa fa	Sintassi comando
Mostra il path della directory corrente	<i>pwd</i>
Mostra contenuto directory	<i>ls [-option] [<dir₁> ... <dir_N>]</i>
Cambia directory	<i>cd [<dir>]</i>
Rimuovi directory	<i>rm [-options] <file₁> ... <file_N></i>
Crea directory	<i>mkdir [-options] <dir₁> ... <dir_N></i>
Copiare un file	<i>cp [-options] <file₁> <file₂></i>
Spostare un file	<i>mv [-options] <file₁> <file₂></i>
Visualizzare contenuto file	<i>cat [-options] <file₁> ... <file_N></i>
Descrizione completa del comando	<i>man <command></i>

E MOLTI ALTRI...

<https://ss64.com/bash/>

Consigli per navigare fra le directory

- **Il carattere tilde ~**

- Questo indica la home dell'utente.

- **I due punti ..**

- I due punti consecutivi indicano la directory di livello superiore.

- **Il tasto TAB**

- Con l'uso del TAB abilitiamo il completamento automatico del nome del file o della directory che stiamo digitando. Così, se si comincia a scrivere il nome, ed a metà si digita TAB, questo verrà completato automaticamente, sempre che la directory o il file siano presenti nella directory.

PRIMO PROGRAMMA IN C

FONDAMENTI DI PROGRAMMAZIONE

Primo programma in C

```
1  #include <stdio.h>
2
3  int main(void){
4      printf("Hello world\n");
5
6      return 0;
7  }
```

- A cosa serve **#include**?
- Cos'è il **main**?
- Cosa rappresenta il carattere '**\n**'?
- A cosa servono le parente graffe "**{ }**"?
- A cosa servono i punti e virgola "**;**"?

Primo programma in C

```
1  #include <stdio.h>
2
3  int main(void){
4
5      printf("Hello world\n");
6
7      return 0;
8  }
9
```



- A cosa serve ***#include***?
 - ***#include*** è una direttiva del preprocessore che permette di richiamare le librerie del standard C.
 - Senza le librerie un programma non avrebbe a disposizione funzioni necessarie al funzionamento del programma stesso;

Primo programma in C

```
1  #include <stdio.h>
2
3  int main(void){
4
5      printf("Hello world\n");
6
7      return 0;
8  }
9
```



- A cosa serve **#include**?
 - **#include <stdio.h>** stiamo *includendo* la libreria standard Input/Output, che ci permette di utilizzare operazioni di input/output;
 - La funzione **printf()** è definita all'interno della libreria **<stdio.h>**;
 - La funzione **printf()** stampa a video tutto ciò che gli viene passato come argomento;

Primo programma in C

```
1  #include <stdio.h>
2
3  int main(void){
4      printf("Hello world\n");
5
6      return 0;
7  }
8
9
```



- Cos'è il **main**?
 - Il **main()** è la funzione principale di un qualsiasi programma C;
 - Il **main()** è indispensabile;
 - Il **main()** è unico;
 - Può non avere alcun parametro oppure riceverli da riga di comando;

Primo programma in C

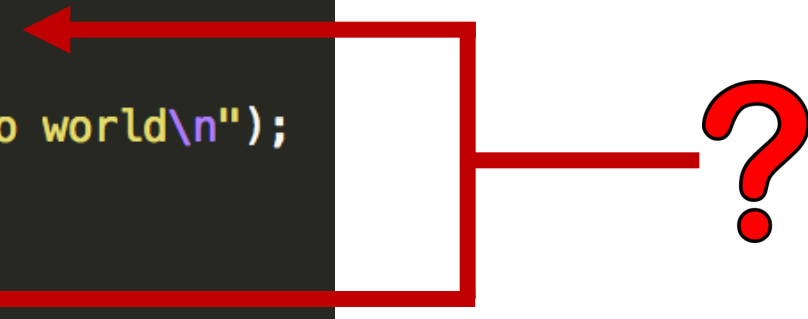
```
1  #include <stdio.h>
2
3  int main(void){
4
5      printf("Hello world\n");
6
7      return 0;
8  }
9
```



- Cosa rappresenta il carattere '**\n**'?
 - Il carattere '****' prende il nome di *escape character* e indica alla **printf()** di fare qualcosa di particolare;
 - La *escape sequence* '**\n**' dice alla **printf()** di andare a capo (*newline*);
 - Permette di andare a capo quando l'output ("Hello world") viene stampato a schermo;
 - Il carattere '**\n**' non viene stampato in output;

Primo programma in C

```
1  #include <stdio.h>
2
3  int main(void){
4      printf("Hello world\n");
5
6      return 0;
7  }
8
9
```



- A cosa servono le parente graffe “{ }”?
 - Le “{ }” servono a delimitare blocchi di istruzioni (**statement**);
 - Gli **statement** vengono eseguiti in ordine dall’alto verso il basso;
 - Il blocco di istruzione della funzione **main()** contiene lo **statement printf()** e la **statement return**;
 - Lo **statement return** viene utilizzato per terminare l’esecuzione di una funzione e restituire il controllo alla funzione chiamante;

Primo programma in C




```
1  #include <stdio.h>
2
3  int main(void){
4
5      printf("Hello world\n");
6
7      return 0;
8  }
9
```



- A cosa servono i punti e virgola “;”?
 - I “;” servono a “chiudere” un’istruzione;
 - Se non inserite i “;” alla fine di ogni istruzione il compilatore vi segnalerà un **errore**;

Primo programma in C

- Spesso in un programma si incorre in alcuni **errori**:
 - **Errore sintattico**: è legato alla sintassi del codice che scrivete (es. ; mancanti, funzioni scritte male ecc.);
 - **Errore semantico**: è legato alla semantica del codice che scrivete (es. ha senso fare 4/0);
- Il programma potrebbe essere sintatticamente e semanticamente corretto (quindi il compilatore non ci segnali alcun problema) ma potrebbe non restituire il risultato atteso dal programmatore:
 - **Errore logico**: errore legato alla logica del programma ;

 Sintattico	 Semantico	 Logico
Poco grave	Mediamente grave	Molto grave
Facile da trovare	Non molto facile da trovare	Difficile da trovare

Primo programma in C

```
1  #include <stdio.h>
2
3  int main(void){
4
5      printf("Hello world\n");
6
7      return 0;
8  }
9
```

- Se non ci sono errori sintattici è possibile compilare il programma:

gcc [options] [file_sorgente] [file_oggetto] [-o file_output]

- Per eseguire il programma:

./file_output.out

Primo programma in C

```
1  #include <stdio.h>
2
3  int main(void){
4
5      printf("Hello world\n");
6
7      return 0;
8  }
9
```

HelloWorld.c

- Se non ci sono errori sintattici è possibile compilare il programma:

gcc HelloWorld.c -o HelloWorld.out

- Per eseguire il programma:

./HelloWorld.out

Primo programma in C

```
student@student-VirtualBox: ~/Scrivania/FP
student@student-VirtualBox:~$ ls
Documenti  examples.desktop  Immagini  Modelli  Musica  Pubblici  Scaricati  Scrivania  Video
student@student-VirtualBox:~$
student@student-VirtualBox:~$
student@student-VirtualBox:~$ cd Scrivania/FP/
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ gcc main.c -o main.out
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c  main.out
student@student-VirtualBox:~/Scrivania/FP$ ./main.out
Hello world!!
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
```

Primo programma in C

```
student@student-VirtualBox: ~/Scrivania/FP
student@student-VirtualBox:~$ ls
Documenti  examples.desktop  Immagini  Modelli  Musica  Pubblici  Scaricati  Scrivania  Video
student@student-VirtualBox:~$
student@student-VirtualBox:~$
student@student-VirtualBox:~$ cd Scrivania/FP/
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ gcc main.c -o main.out
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c  main.out
student@student-VirtualBox:~/Scrivania/FP$ ./main.out
Hello world!!
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
```


Primo programma in C

```
student@student-VirtualBox: ~/Scrivania/FP
student@student-VirtualBox:~$ ls
Documenti  examples.desktop  Immagini  Modelli  Musica  Pubblici  Scaricati  Scrivania  Video
student@student-VirtualBox:~$
student@student-VirtualBox:~$ cd Scrivania/FP/
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ gcc main.c -o main.out
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c  main.out
student@student-VirtualBox:~/Scrivania/FP$ ./main.out
Hello world!!
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
```

Primo programma in C

```
student@student-VirtualBox: ~/Scrivania/FP
student@student-VirtualBox:~$ ls
Documenti  examples.desktop  Immagini  Modelli  Musica  Pubblici  Scaricati  Scrivania  Video
student@student-VirtualBox:~$
student@student-VirtualBox:~$
student@student-VirtualBox:~$ cd Scrivania/FP/
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ gcc main.c -o main.out
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c  main.out
student@student-VirtualBox:~/Scrivania/FP$ ./main.out
Hello world!!
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
```

Primo programma in C

```
student@student-VirtualBox: ~/Scrivania/FP
student@student-VirtualBox:~$ ls
Documenti  examples.desktop  Immagini  Modelli  Musica  Pubblici  Scaricati  Scrivania  Video
student@student-VirtualBox:~$
student@student-VirtualBox:~$
student@student-VirtualBox:~$ cd Scrivania/FP/
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ gcc main.c -o main.out
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c  main.out
student@student-VirtualBox:~/Scrivania/FP$ ./main.out
Hello world!!
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
```

Primo programma in C

```
student@student-VirtualBox: ~/Scrivania/FP
student@student-VirtualBox:~$ ls
Documenti  examples.desktop  Immagini  Modelli  Musica  Pubblici  Scaricati  Scrivania  Video
student@student-VirtualBox:~$
student@student-VirtualBox:~$
student@student-VirtualBox:~$ cd Scrivania/FP/
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ gcc main.c -o main.out
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c  main.out
student@student-VirtualBox:~/Scrivania/FP$ ./main.out
Hello world!!
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
```

Primo programma in C

```
student@student-VirtualBox: ~/Scrivania/FP
student@student-VirtualBox:~$ ls
Documenti  examples.desktop  Immagini  Modelli  Musica  Pubblici  Scaricati  Scrivania  Video
student@student-VirtualBox:~$
student@student-VirtualBox:~$
student@student-VirtualBox:~$ cd Scrivania/FP/
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ gcc main.c -o main.out
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$ ls
main.c  main.out
student@student-VirtualBox:~/Scrivania/FP$ ./main.out
Hello world!!
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
student@student-VirtualBox:~/Scrivania/FP$
```

GCC opzioni

gcc [options] [file_sorgente] [file_oggetto] [-o file_output]

Opzione	Descrizione
-c	Compila i file sorgente in file oggetto senza linking
-Wall	Abilita tutti i messaggi di warning
-Wextra	Abilita i messaggi di warning extra
-w	Disabilita tutti i messaggi di warning
-v	Stampa informazioni di compilazione
-E	Produce l'output della fase di preprocessing
-S	Produce il codice assembly senza creare il file oggetto

E MOLTI ALTRI...

<https://gcc.gnu.org/onlinedocs/gcc-4.8.2/gcc/Option-Index.html#Option-Index>

ESERCIZI

Primo programma in C

FONDAMENTI DI PROGRAMMAZIONE



- Aprire un editor di testo qualsiasi (es. **SublimeText**):

1. **Scrivere** il seguente codice

```
#include<stdio.h>
int main(void){
    printf( "Hello world \n" );
    return 0;
}
```

2. **Salvare** il file nella *directory Scrivania* con nome *"helloworld.c"*;

- Eseguire i seguenti passi tramite riga di comando:

1. **Controllare** il path della *directory corrente*;
2. **Controllare** il contenuto della *directory corrente*;
3. **Spostarsi** nella *directory "Scrivania"*;
4. **Creare** una nuova directory chiamata *"FP"*;
5. **Spostarsi** nella *directory "FP"*;
6. **Controllare** il path della *directory corrente*;
7. **Creare** una nuova directory chiamata *"PrimoProgramma"*;
8. **Spostare** il file *"helloworld.c"* dalla Scrivania alla *"PrimoProgramma"*;
9. **Compilare** il file *"helloworld.c"* e **lanciare** l'eseguibile *"helloworld.out"*;



- Aprire un editor di testo qualsiasi (es. **SublimeText**):

1. **Scrivere** il seguente codice

```
#include<stdio.h>
int main(void){
    printf( "Hello ");
    printf("world");
    printf("\n" );
    return 0;
}
```

2. **Salvare** il file nella *directory Scrivania* con nome *"helloworld_1.c"*;

- **Compilare** il file *"helloworld_1.c"* e **lanciare** l'eseguibile *"helloworld_1.out"*

L'output è lo stesso?



- Secondo voi il seguente codice *compila*?

```
#include<stdio.h>  
  
int main(void)  
    printf( "Hello world  
    ");  
    return 0;  
}
```



- Secondo voi il seguente codice *compila*?

```
#include<stdio.h>

int main(void){
    printf( "Hello world
");
    return 0;
}
```

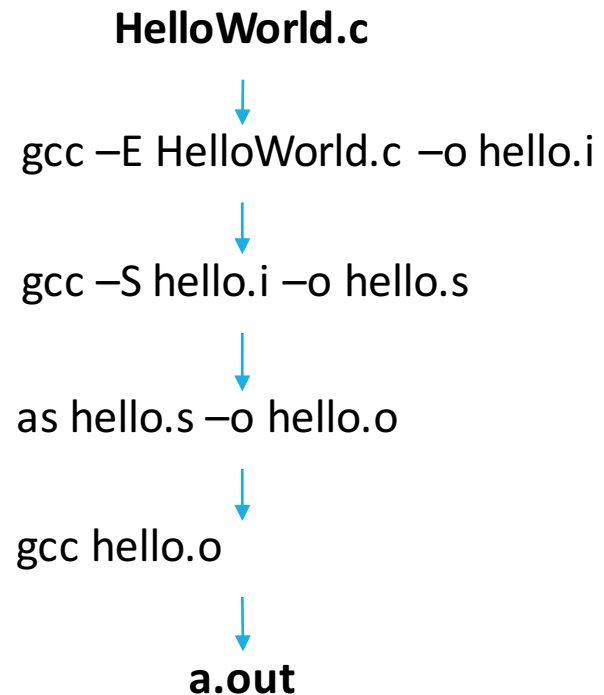
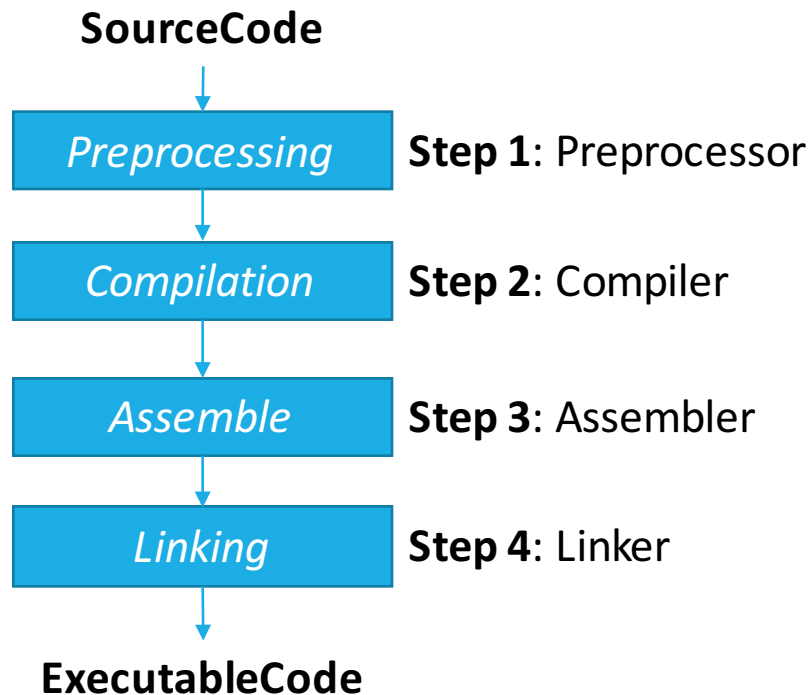
```
main.c:5:9: warning: missing terminating '"' character [-Winvalid-pp-token]
    printf("Hello world
      ^
main.c:5:9: error: expected expression
main.c:6:3: warning: missing terminating '"' character [-Winvalid-pp-token]
    ");
      ^
main.c:11:1: error: expected '}'
^
main.c:3:15: note: to match this '{'
int main(void){
      ^
2 warnings and 2 errors generated.
[Finished in 0.2s with exit code 1]
```

HOMeworks

FONDAMENTI DI PROGRAMMAZIONE

HOMeworks

- Il processo di compilazione di gcc è mostrato in figura.



- Utilizzando la riga di comando, vedere gli output del processo di compilazione (schema a destra).

HOMEWORKS

- Scrivere un programma C che:
 - Stampi il messaggio “Questo è un programma C.” su *una linea*.
 - Stampi il messaggio “Questo è un programma C.” su *due linee*, in modo tale che la prima linea termini con “*un*”.
 - Stampi il messaggio “Questo è un programma C.” su *più linee*, in modo che ogni parola del messaggio sia stampata su una linea diversa.

HOMEWORKS

- Le ***escape sequences*** maggiormente adoperate in C sono mostrate nella tabella seguente:

Escape sequence	Description
<code>\n</code>	Newline. Position the cursor at the beginning of the next line.
<code>\t</code>	Horizontal tab. Move the cursor to the next tab stop.
<code>\a</code>	Alert. Produces a sound or visible alert without changing the current cursor position.
<code>\\</code>	Backslash. Insert a backslash character in a string.
<code>\"</code>	Double quote. Insert a double-quote character in a string.

- Scrivere un semplice programma C che utilizzi le *escape sequences* mostrate nella tabella e che stampi in output il risultato finale.