Strumenti di sviluppo ed esecuzione di programmi C++

Corso di Fondamenti di Programmazione
Corso di Laurea in Informatica
Università della Calabria
https://www.mat.unical.it/informatica/FondamentiDIInformatica

Strumenti di sviluppo

Si richiede l'installazione di una distribuzione di Linux, come Ubuntu

- Linux ha già installato di base il software di sviluppo necessario per questo corso
- Si tratta di un sistema operativo completo
  - Ai fini del corso è sufficiente avere una shell (terminale) Linux
- Ci sono diverse alternative per avere un terminale Linux sul proprio pc
  - Installare il Windows Subsystem per Linux (WSL)
  - Creare una macchina virtuale, ad esempio con VMWare o VirtualBox
  - Creare una partizione aggiuntiva e installarci una distribuzione Linux
  - Etc.

Per compilare ed eseguire programmi C++ ci servono:

- Compilatore **g++**
- Editor di testo che *non sia Word Processor*, come **gedit**

Shell Linux

- shell è un software che riceve comandi da tastiera e li inoltra al sistema operativo per essere eseguiti
  - In passato, era l'unico strumento a disposizione degli utenti sui sistemi Linux
- Shortcut per aprire una shell su Linux: **Ctrl + Alt + T**

Comandi principali

Vedere l'elenco dei file contenuti nella cartella corrente

`ls`

«ls» è un'abbreviazione dell'inglese **«list»** → elencare

Creazione di cartella

`mkdir` ***`nome_cartella`***

«mkdir» è un'abbreviazione dell'inglese **«make directory»** → creare una cartella

Esempio:

`mkdir mia_cartella` → crea la cartella ***`mia_cartella`*** all'interno della cartella corrente

Comandi principali

Spostarsi tra le cartelle

`cd` ***`path_cartella`***

«cd» è un'abbreviazione dell'inglese **«change directory»** → cambiare cartella

Esempio:

`cd mia_cartella` → entro in ***`mia_cartella`***

`cd ..` → entro nella cartella a livello superiore

Possiamo specificare sia path assoluti che relativi

- **Path assoluto**: inizia con / e indentifica il path completo nel sistema operativo

  Ad esempio: `cd` ***`/home`***

- **Path relativo**: indentifica il path in riferimento alla cartella corrente

  Ad esempio: `cd` ***`mia_cartella`*** → ***`mia_cartella`*** si deve trovare nella cartella corrente

Comandi principali

Vedere il path assoluto della cartella corrente

`pwd`

«pwd» è un'abbreviazione dell'inglese **«print working directory»** → letteralmente, stampa la cartella di lavoro ovvero quella corrente

Per avere maggiori informazioni su un comando

`man` *`nome_comando`*

«man» è un'abbreviazione dell'inglese **«manual»** → manuale

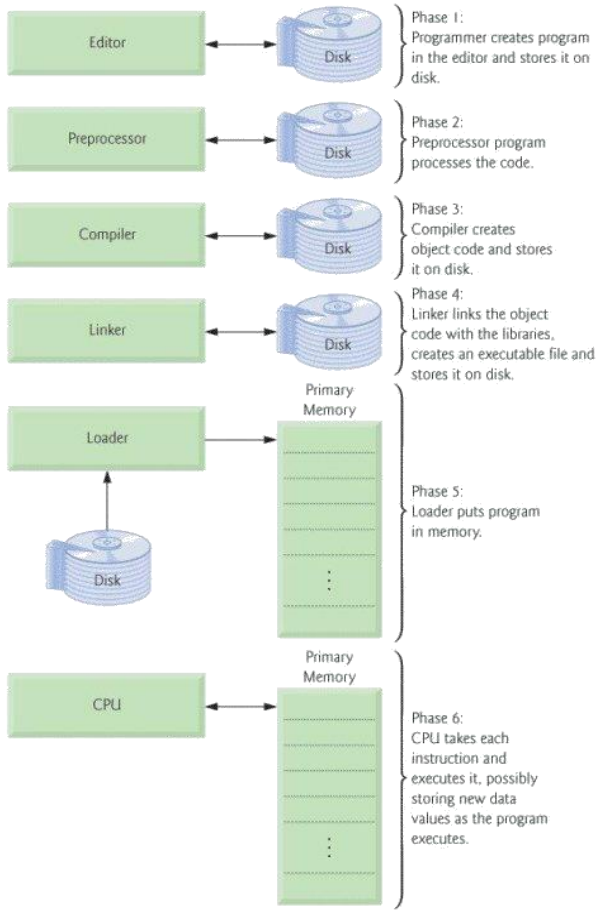
Esempi:

`man ls` → manuale del comando **`ls`**

`man cd` → manuale del comando **`cd`**

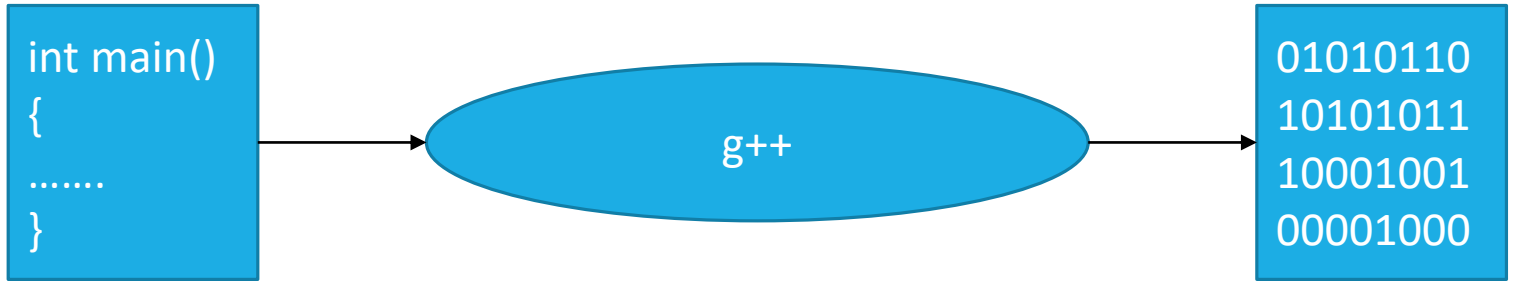
Fasi di un programma C++

1. **Edit**: Scrittura del programma
2. **Preprocess**: Preprocessamento delle inclusioni
3. **Compile**: Compilazione
4. **Link**: Collegamento con le «librerie» esterne
5. **Load**: Caricamento in memoria
6. **Execute**: Esecuzione



Compilazione

- A seguito delle fasi di compilazione e linking, il programma C++ è trasformato in un programma ***eseguibile*** equivalente, scritto in linguaggio macchina
- Il programma ***eseguibile*** può essere eseguito più volte senza dover tradurre nuovamente
- La compilazione è assimilabile al processo di traduzione, da una lingua ad un'altra, di un libro



Uso del compilatore g++

Source File contenente il main -> main.cpp

```
#include <iostream>
using namespace std;

int main(){
    cout<<"La lezione di oggi è
            in lab";
    return 0;
}
```

Per compilazione e linking:

```
g++ main.cpp
```

Per default viene generato un file eseguibile di nome **a.out**

Per eseguire:

```
./a.out
```

L'opzione **–o** consente di specificare il nome dell'eseguibile:

```
g++ main.cpp –o prova
```

Per eseguire:

```
./prova
```

Tipi di Errore

Errori Sintattici

- facili da individuare
- segnalati automaticamente dal compilatore

Errori Semantici

- implicano un comportamento inaspettato del programma
- nascosti all'interno del codice
- non segnalati dal compilatore

Debugger

- analizza il programma in corso d'esecuzione
- verifica risultato singole operazioni

Attenzione…

- Si consiglia di creare una cartella per ogni esercizio
- Compilare frequentemente, a seguito di ogni modifica significativa
- Leggere con attenzione gli errori segnalati dal compilatore a partire dal primo errore segnalato

Esercizio

Scrivere un programma c++ in cui viene stampato il proprio nome e cognome.