

Introduzione a Ingresso e Uscita Compilazione

Come si termina ...

- ... un programma in esecuzione (***processo***)?
- **Ctrl + C**
- In UNIX ci si basa sul concetto di *terminale*
- Anche da GUI, quello che si apre è un terminale (Terminal, Konsole, xterm, ...)
- In seguito a determinate combinazioni di caratteri il terminale spedisce speciali **segnali** ai processi

Ingresso e uscita

- **Input/Output**
 - Ingresso di informazioni (da elaborare) all'interno di un processo
 - Uscita di informazioni (elaborate) da un processo
- **Esempio:** stampa di informazioni sullo schermo, lettura di valori da tastiera

Ingresso e Uscita in C++

- Il linguaggio C++ non prevede istruzioni per l'ingresso/uscita
- Implementato mediante *oggetti di libreria* chiamati *stream*
 - *stream*: *flusso di caratteri*
 - *ostream*: *flusso di caratteri in uscita, output formattato*
 - *istream*: *flusso di caratteri in ingresso, input formattato*

- Flusso di caratteri:
 - successione di righe,
 - ciascuna costituita da zero o più caratteri, e
 - terminata dal carattere speciale newline '\n'

cin, cout, cerr 1/2

- Quando un programma inizia la propria esecuzione ci sono tre flussi di caratteri già aperti
 - *cin*: flusso standard di ingresso
 - *cout*: flusso standard di uscita
 - *cerr*: flusso standard di uscita per comunicare messaggi di errore

cin, cout, cerr 2/2

- Se il programma è invocato da una shell Unix senza *redirezionamenti*
 - Lettura da *cin*:
 - Lettura dei caratteri immessi dal terminale in cui gira la shell (tastiera)
 - Scrittura su *cout* o su *cerr*:
 - Visualizzazione sul terminale in cui gira la shell

Operatore di uscita << 1/2

- Scrittura formattata su *cout*
- Forma più semplice
 - *cout<<stringa ;*
 - ove *stringa* è una sequenza di caratteri delimitata da doppi apici “
 - “*esempio di stringa*”

File sorgente

- File sorgente (unità di traduzione): file di testo che contiene (parte del) il programma scritto nel linguaggio (di alto livello) di partenza

- Ogni file sorgente che contenga riferimenti ad oggetti della libreria di ingresso/uscita deve contenere le direttive

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

- Devono precedere il primo punto in cui viene utilizzato uno stream di ingresso/uscita

Anatomia programma C++

// direttive

main()

{

// istruzioni

}

Primo esercizio 1/4

- Sulle tastiere italiane:
- { **Alt + 123** sul tastierino numerico
- } **Alt + 125** sul tastierino numerico
- { **Alt Gr + Shift + è**
- } **Alt Gr + Shift + +**
- Scrivere un programma che stampi *Ciao mondo* sul terminale

Primo esercizio 2/4

```
#include <iostream>  
using namespace std;  
main()  
{  
    cout<<"Ciao mondo!" ;  
}
```

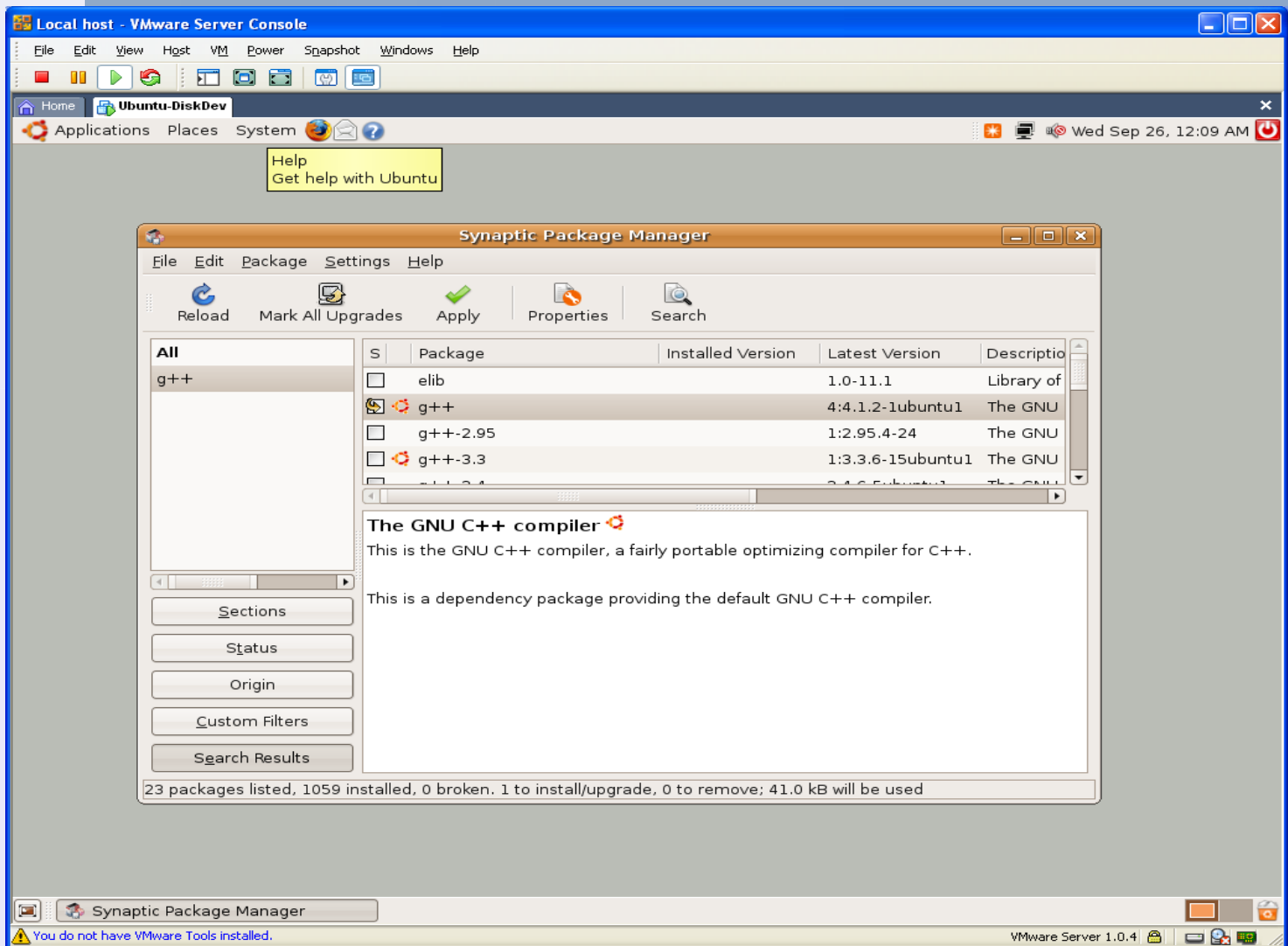
Compilazione

- File sorgente (unità di traduzione): file di testo che contiene il programma scritto nel linguaggio (di alto livello) di partenza
- Per ottenere un programma eseguibile a partire dal nostro sorgente possiamo utilizzare un compilatore per il linguaggio C++
- Schema:
 - Sorgente->Compilazione->Esecuibile

Compilatore gcc 1/2

- gcc: GNU Compiler Collection
- g++: front end al gcc per compilare sorgenti C++
- Tutte le informazioni sul compilatore:
 - <http://www.gnu.org/software/gcc/>
 - *man g++*
- Progetto GNU:
 - <http://www.gnu.org/>

Installazione g++



- <http://informatica.scienze.unimo.it/corso-pc>
- Appendice_Linux.txt

Compilatore gcc 2/2

- Sintassi più semplice per generare un programma eseguibile da un file sorgente:
- *g++ nome_sorgente.cc*
 - Assegna un nome predefinito al programma eseguibile (a.out)
- *g++ -o nome_eseguibile nome_sorgente.cc*
 - Permette di scegliere il nome del programma eseguibile

Proviamo ...

- ... a compilare ed eseguire il nostro programma ...

Messaggi di errore 1/2

- Può darsi che la compilazione non sia andata a buon fine
- In questo caso il compilatore ci ha sicuramente dato dei messaggi

Messaggi di errore 2/2

- Se ci sono problemi, il compilatore può comunicare
 - **Warnings** (avvisi): c'è qualcosa di 'sospetto' nel codice, ma si può comunque generare un eseguibile
 - **Errors**: ci sono errori che impediscono la conclusione della compilazione

Se tutto ha funzionato ...

- Forse il *prompt* appare appiccicato al nostro messaggio ...
- Non siamo andati a capo!
- Bisognerebbe poter stampare il carattere *a capo* (*newline*)

Sequenze di controllo

- I caratteri non visualizzabili (caratteri speciali) possono essere rappresentati mediante sequenze di controllo (*escape sequence*)
- `\n` newline
- `\t` tabulazione
- `\\` barra inversa
- `\'` apice
- `\"` virgolette

Primo esercizio 3/4

- Modificare il programma affinché vada anche a capo

Primo esercizio 4/4

```
#include <iostream>  
using namespace std;  
main()  
{  
    cout<<"Ciao mondo!\n" ;  
}
```


Accodamento operatori

- E se volessimo stampare il valore di una variabile?
- Gli operatori `<<` possono essere accodati l'uno all'altro
- Esempio: `cout<<"Ciao " <<"mondo\n";`
- Gli argomenti verranno stampati l'uno di seguito all'altro

Manipolatori

- Possono essere passati all'operatore di uscita
- Modificano in qualche modo la formattazione dell'ingresso/uscita
- Esempio:
 - *endl*: equivalente alla sequenza di controllo `\n`

Esercizio 2 1/2

- Usare il manipolatore *endl* per andare a capo nel precedente programma

Esercizio 2 2/2

```
#include <iostream>  
using namespace std;  
main()  
{  
    cout<<"Ciao mondo!"<<endl ;  
}
```