Esercitazione 12

17/12/2010

Es. Makefile

Si vuole creare un `Makefile` per il programma contenuto dell'archivio `makefile.zip`, in modo tale che digitando `make` sulla linea di comando il programma contenuto possa essere compilato correttamente.
Si seguano i seguenti passi

1. Scrivere il `Makefile` nel formato più semplice, utilizzando quindi chiamate del tipo:

```
edit : main.o kbd.o command.o display.o
        g++ -o edit main.o kbd.o command.o display.o

main.o : main.cpp defs.hpp
        g++ -c main.cpp

kbd.o : kbd.cpp defs.hpp command.hpp
         g++ -c kbd.cpp

clean :
         rm edit main.o kbd.o command.o
```

   utilizzando per la compilazione di ciascun file l'opzione `-Wall`. Fare attenzione che i files sono contenuti nella sotto cartella `src`, alle dipendenze dei files e alle eventuali chiamate "a vuoto" di `clean`.

2. Modificare il `Makefile` precedente introducendo le seguenti variabili

   - `CXX` per indicare il compilatore utilizzato;
   - `OPTFLAGS` per indicare le opzioni di ottimizzazione;
   - `CXXFLAGS` per indicare le opzioni non di ottimizzazione;
   - `EXEC` il nome dell'eseguibile;

   Per usare una variabile digitare `$(NOMEVARIBILE)`. Utilizzare inoltre `.PHONY` per indicare quali tag non sono associati a files. Abilitare l'opzione di ottimizzazione `-O2` per tutti i files. La variabile `CXXFLAG` conterrà anche quanto definito nella variabile `OPTFLAG`, ma senza ripeterlo.

3. Vogliamo gestire in maniera semplificata i files. Consideriamo le seguenti flag del compilatore

   - `-MM` stampa una regola utile per il `Makefile` che descrive le dipendenze del file. Per un esempio provare a digitare:

```
g++ -MM main_integration.cpp
```

- `-MF` associata alla flag `-MM` scrive l'output su il file indicato. Per un esempio provare a digitare:

  ```
  g++ -MM main_integration.cpp -MF make.dep
  ```

Introdurre le seguenti variabili

- `EXESRCS` il file sorgente contenente il main;
- `EXEOBJS` il file oggetto associato al main, generato automaticamente dalla variabile `EXESRCS`. Un modo per farlo è tramite il seguente comando

  ```
  EXEOBJS = $(EXESRCS:.cpp = .o)
  ```

- `FOLDER` per indicare la cartella dove sono contenuti i files sorgenti;
- `SRCS` in cui è presente la lista di tutti i files sorgenti presenti nella cartella `FOLDER`. Lista generata automaticamente;
- `OBJS` in cui è presente la lista dei files oggetto, generata automaticamente partendo dalla variabile `SRCS`;
- `HEADERS` in cui è presente la lista dei files `.hpp`, generata automaticamente partendo dalla variabile `SRCS`;
- `DEPEND` il nome del file su cui vengono scritti gli output associati ad `-MM`;

Seguire i seguenti passi

(a) quando viene chiamato il tag `all` inserire come prima dipendenza la generazione del file delle dipendenze;

(b) inserire il tag per generare il file delle dipendenze, associato sia ai files in `SRCS` sia a quello in `EXESRCS`;

(c) inserire come secondo tag di `all` la lista dei file oggetto, in questo modo vengono automaticamente creati i corrispondenti files oggetto con le flags indicate in `CXXFLAGS`;

(d) attraverso la keyword `include` inserire il file con le dipendenze, facendo attenzione al caso in cui esso non è presente;

(e) inserire come terzo tag di `all` il file oggetto contenente il main.

4. Introdurre la gestione delle librerie. Vogliamo quindi inserire tutti i files oggetto associati ai files sorgenti in `SRCS` all'interno di una libreria. I passi sono i seguenti

   (a) introdurre la seguente variabile locale `LIBNAME` in cui viene memorizzato il nome della libreria. Si ricorda che una libreria statica in Linux ha la seguente sintassi

   ```
   libnumint.a
   ```

   ovvero è la libreria `numint` statica;

   (b) introdurre la variabile locale `LDLIBS` che contiene la lista delle librerie esterne;

   (c) introdurre un nuovo tag uguale al nome della libreria in cui viene creata la libreria, tale tag dipende dai files oggetto. Si ricorda che la sintassi è la seguente

   ```
   ar rv libPROVA.a pippo.o pluto.o
   ```

   (d) cambiare il secondo tag di `all` in modo tale che generi la libreria;

   (e) aggiungere la libreria generata alla variabile `LDLIBS`;

   (f) specificare nella compilazione del main la dipendenza dalle variabili esterne.

Es. factory