

Introduzione ai compilatori C/C++

Esistono molti compilatori del linguaggio C/C++ gratuiti sia sotto Linux che sotto Windows. Un compilatore C/C++ che gira sotto Linux molto noto e diffuso è **gcc** (<http://gcc.gnu.org>). A pag. 66 del libro di F. Scorzoni, trovate una breve spiegazione delle principali opzioni del compilatore gcc (un'opzione controlla il comportamento del compilatore). Il compilatore **Dev-C++** comprende anche un comodo **IDE**, Integrated Development Environment, cioè un ambiente di sviluppo integrato, con il quale è possibile non solo compilare il programma ma editarlo, associarlo a un progetto, farne il debug (è l'attività con la quale si correggono gli errori del programma), osservare il contenuto delle variabili, sospendere l'esecuzione a un certo punto ecc ecc. Esso si basa sul compilatore gcc.

In Dev-C++, il file oggetto (il risultato del processo di compilazione, si veda la figura in fondo al presente documento) solitamente ha il nome del file sorgente (un file sorgente contiene il codice, cioè i comandi che saranno eseguiti dal processore ovviamente in linguaggio C) seguito dall'estensione **.o** (o sta per **oggetto**), mentre il file eseguibile (il risultato del processo di linking, che rappresenta il programma **eseguibile**) ha invece il nome del progetto seguito dall'estensione **.exe**.

Utilizzo del compilatore C/C++ (gratuito) della Borland (<http://www.borland.com>)

Un altro compilatore anch'esso molto diffuso è il compilatore C/C++ della Borland (<http://www.borland.com>). Il compilatore a riga di comando noto col nome di **bcc32** nella versione 5.5.1 è anch'esso gratuito.

Installazione del compilatore

1. Scaricate il file **bcc32_setup.exe** sul Desktop del vostro computer
2. Fate doppio click sul file (farete partire l'installazione)
3. Scegliete come destinazione la cartella **C:\Borland\BCC55**
4. Posizionatevi nella cartella **Bin** (sottocartella di BCC55) col comando
`cd C:\Borland\BCC55\Bin`
5. Create all'interno di questa cartella un file di nome **bcc32.cfg** contenente le seguenti righe
`-I"c:\Borland\Bcc55\include"`
`-L"c:\Borland\Bcc55\lib"`

che imposterà le opzioni -I e -L del compilatore riguardanti i percorsi Include e Lib

6. Create all'interno della cartella precedente un file di nome **ilink32.cfg** contenente la seguente riga
`-L"c:\Borland\Bcc55\lib"`

che imposterà l'opzione del linker per il percorso Lib

7. Aggiungete al file **autoexec.bat** i due seguenti percorsi (path)
`C:\Borland\BCC55; C:\Borland\BCC55\Bin`

Se ciò non fosse possibile (il file **autoexec.bat** molte volte risulta protetto dal sistema), conviene aggiungere i due percorsi alla **variabile di ambiente Path** solitamente disponibile nei sistemi Windows (tasto destro del mouse sull'icona Risorse del computer > Proprietà del sistema > scheda Avanzate > Pulsante Variabili d'ambiente > riquadro Variabili di sistema > Nuovo > Path oppure Modifica > (copiare i due percorsi in coda a quanto è già scritto).

Testare il compilatore

Per testare il compilatore, scrivete il seguente programma che chiameremo **main.c**, salvandolo in una propria cartella di lavoro (ad esempio C:\MaurizioCozzetto). Per costruire la cartella, dal prompt di sistema (pulsante Start > Programmi > Accessori > Prompt di sistema), scrivere

```
md C:\MaurizioCozzetto
```

(ovviamente sostituire con il proprio nome e cognome).

Segue programma

```
#include <stdio.h>

int main() {
    // Pongo a=0 per evitare comportamenti indesiderati del programma
    int a=0;

    // Introduco un valore di a
    printf("Introduci a ");
    scanf("%d",&a);

    // Calcolo il doppio del numero intero
    printf("Il doppio di %d è %d\n",a,2*a);

    return 0;
}
```

Per compilare il programma, entriamo nella cartella con nome e il cognome che abbiamo deciso (nel mio caso MaurizioCozzetto) col comando

```
Cd C:\MaurizioCozzetto
```

e scriviamo dal prompt

```
bcc32 main.c
```

(Il compilatore crea un file di nome **bcc32.obj**, richiama automaticamente il linker e crea un eseguibile di nome **main.exe**, più altri file di supporto)

Per eseguire il programma, occorre caricarlo in memoria e scrivere dal prompt dei comandi

```
main
```

(seguito da invio)

Riutilizzo del codice

Se ci venisse richiesto di riutilizzare il codice che calcola il doppio di un numero intero (ovviamente l'esempio è banale), probabilmente potremmo avere dei problemi (al più potremmo usare il copia-incolla ma sappiamo benissimo che possiamo facilmente sbagliare se usiamo il copia-incolla più volte magari ripetutamente).

Abbiamo allora due possibilità:

- A) Possiamo scrivere una funzione che calcola il doppio di un numero intero e richiamarla nei nostri progetti nel modo seguente

```
#include <stdio.h>

// Prototipo di funzione (in Java si direbbe segnatura); nel compilatore Borland è obblig.
int doppio(int);

int main() {
    // Pongo a=0 per evitare comportamenti indesiderati del programma
    int a=0;

    // Introduco un valore di a
    printf("Introduci a ");
    scanf("%d",&a);
    // Calcolo il doppio del numero intero
    printf("Il doppio di %d è %d\n",a,doppio(a));
    return 0;
}
```

```
// calcola il doppio di un generico numero intero x (parametro formale)
int doppio(int x) {
    return 2*x;
}
```

Tutto il listato è contenuto all'interno di un file unico di nome **main2.c** (attenzione che il riquadro tratteggiato separa il main dalla funzione solo perché c'è un salto pagina qui nel documento; in realtà tutto il codice è contenuto in un solo file).

Per compilare procediamo nello stesso modo

```
bcc32 main2.c
```

Per eseguire il programma, dal prompt scriviamo

```
main2
```

(viene caricato in memoria e lanciato il programma main2.exe)

B) L'approccio precedente si rivela problematico in quanto la funzione è contenuta nel file main2.c

Allora decidiamo di scrivere il codice della funzione doppio in un file che chiameremo **double.c**

```
// calcola il doppio di un generico numero intero x (parametro formale)
int doppio(int x) {
    return 2*x;
}
```

Dobbiamo solo compilare, per cui utilizziamo l'opzione -c

```
bcc32 -c double.c
```

otteniamo in tal caso il file **double.obj**

In questo caso, il programma diventa (**main3.c**)

```
#include <stdio.h>
int doppio(int);

int main() {
    // Pongo a=0 per evitare comportamenti indesiderati del programma
    int a=0;

    // Introduco un valore di a
    printf("Introduci a ");
    scanf("%d",&a);

    // Calcolo il doppio del numero intero
    printf("Il doppio di %d è %d\n",a,doppio(a));

    return 0;
}
```

Se lo compilassi e facessi il link (il compilatore richiama sempre il linker a meno che non usi l'opzione -c)

otterrei un messaggio di errore:

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland
main3.c:

Turbo Incremental Link 5.00 Copyright (c) 1997, 2000 Borland

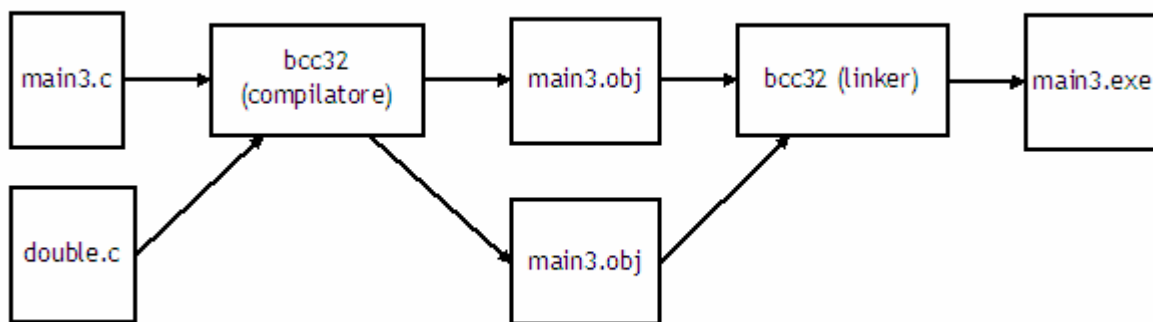
Error: Unresolved external '_doppio' referenced from C:\BORLAND\BCC55\BIN\MAIN3.OBJ

Questo perchè il riferimento alla funzione doppio è **esterno** (all'interno di main3.c non c'è alcun riferimento alla funzione doppio).

Risolviamo compilando nel seguente modo (la compilazione precedente ha già creato un file di nome main3.obj):

```
bcc32 main3.obj double.obj
```

Il seguente schema dovrebbe chiarire tutte le fasi precedentemente descritte:



Otteniamo il file `main3.exe` (il nome del file eseguibile è il primo indicato nell'elenco) che lanciato dal prompt verrà eseguito correttamente

Conclusioni

Conviene progettare i programmi in C in maniera modulare (questo approccio si chiama approccio top-down). Occorre individuare le funzioni necessarie al funzionamento del programma, collaudarle e compilarle in formato oggetto (.obj). Il main program richiama le funzioni all'interno del programma. Per risolvere i riferimenti esterni ed ottenere un unico file eseguibile, conviene procedere come visto precedentemente.

