DevC++ vers. 4.9.9.2

Manuale d’uso

Liceo Scientifico “N. TRON” di SCHIO (VI)

Sommario

1 INTRODUZIONE 3

2 SETUP 5

2.1 Procedura di installazione 5

2.2 Configurazione 7
2.2.1 Opzioni dell'ambiente 7
2.2.2 Opzioni dell'editor 8
2.2.3 Scaricare le librerie ed eventuali aggiornamenti 9

3 SESSIONE DI LAVORO 11

3.1 Realizzare una Applicazione 11
3.1.1 Creazione di un nuovo programma (file sorgente) 12
3.1.2 Apertura di un file esistente 12
3.1.3 Modifica del codice sorgente 13

3.2 Esecuzione del programma 14
3.2.1 Compilazione 14
3.2.2 Compilazione, link ed esecuzione 16
3.2.2 Esecuzione con parametri 18

3.3 Salvataggio del codice 18

4 ALCUNI RIFERIMENTI UTILI 20

4.1 Help in linea 20

4.2 Bloodshed.net 20

4.3 SourceForge.NET 20

4.4 Forum di sviluppatori 20

1 Introduzione

DevC++ è un ambiente di sviluppo integrato (IDE, ovvero Integrated Development Environment) che utilizza il compilatore GCC che è pienamente compatibile con lo standard C89 o ANSI C.

Fig. 1.1: *Informazioni su DevC++*

ANSI C è uno standard proprio perchè la maggior parte dei compilatori soddisfa le specifiche da esso previste per il linguaggio.

Grazie alla sua interfaccia grafica, un ambiente IDE consente di utilizzare in modo semplice gli strumenti offerti dal compilatore senza dover ricorrere necessariamente alle istruzioni da riga di comando (command-line).

L'ultima versione ad oggi disponibile è la 4.9.9.2.

L'IDE della 4.9.9.2 si presenta come in fig. 1.2.

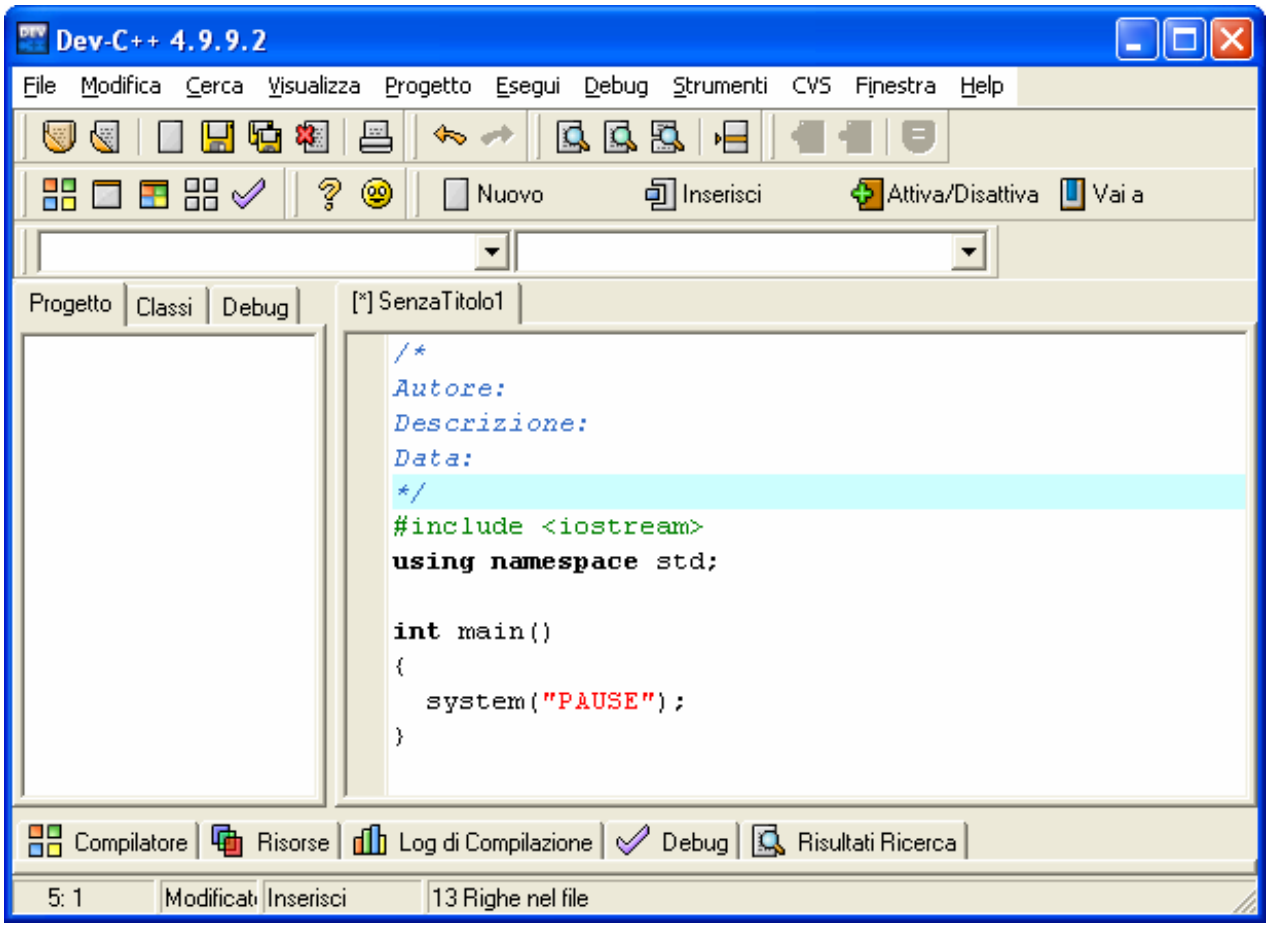


Fig. 1.2: *L'IDE DevC++ 4.9.9.2*

2 Setup

1. Per installare e configurare DevC++ 4.9.9.2 è necessario:
2. installare DevC++ versione 4.9.9.2 mediante doppio clic sul file *devcpp-4.9.9.2_setup.exe*;
3. configurare DevC++ per prepararlo all'utilizzo.

2.1 Procedura di installazione

Prevede i passi di seguito riportati (wizard).
Dapprima il licence agreement: DevC++ è gratuito.

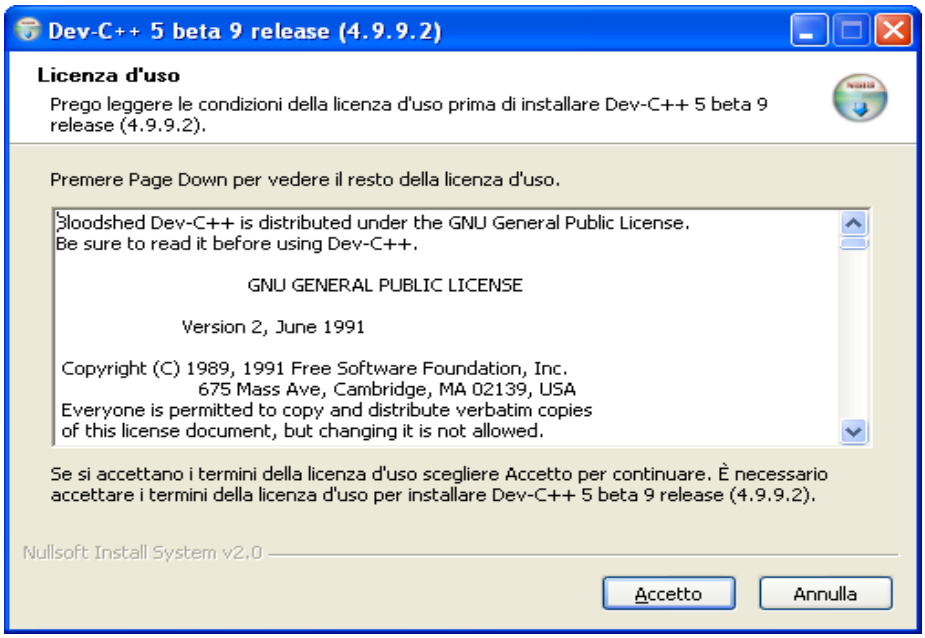


Fig. 2.1: License Agreement

Poi le opzioni di installazione: l'installazione tipica è di default e prevede la selezione automatica dei componenti necessari.

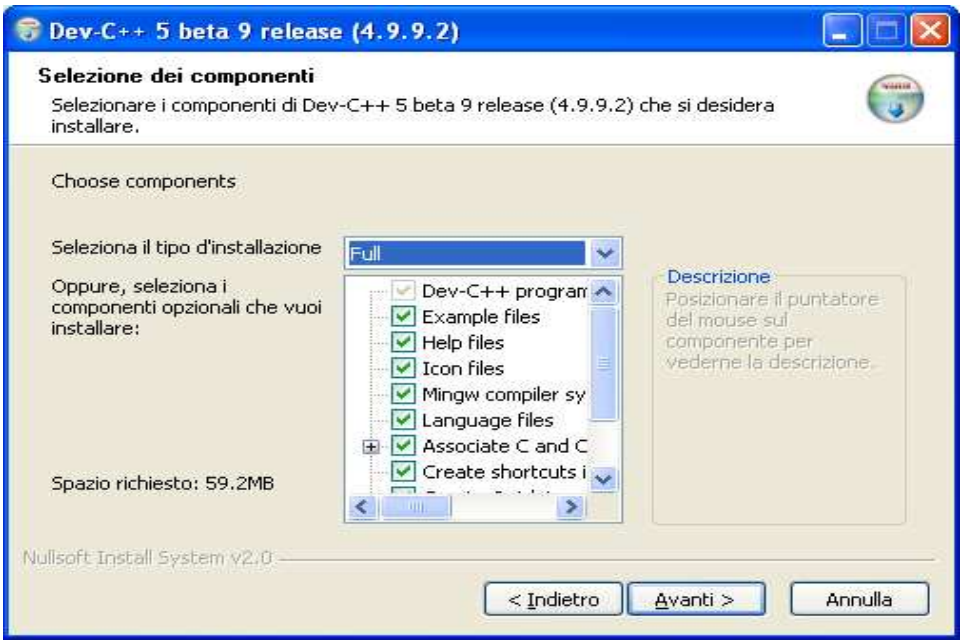


Fig. 2.2: Installazione completa

Si sceglie quindi la directory di installazione, preferibilmente la cartella Programmi.

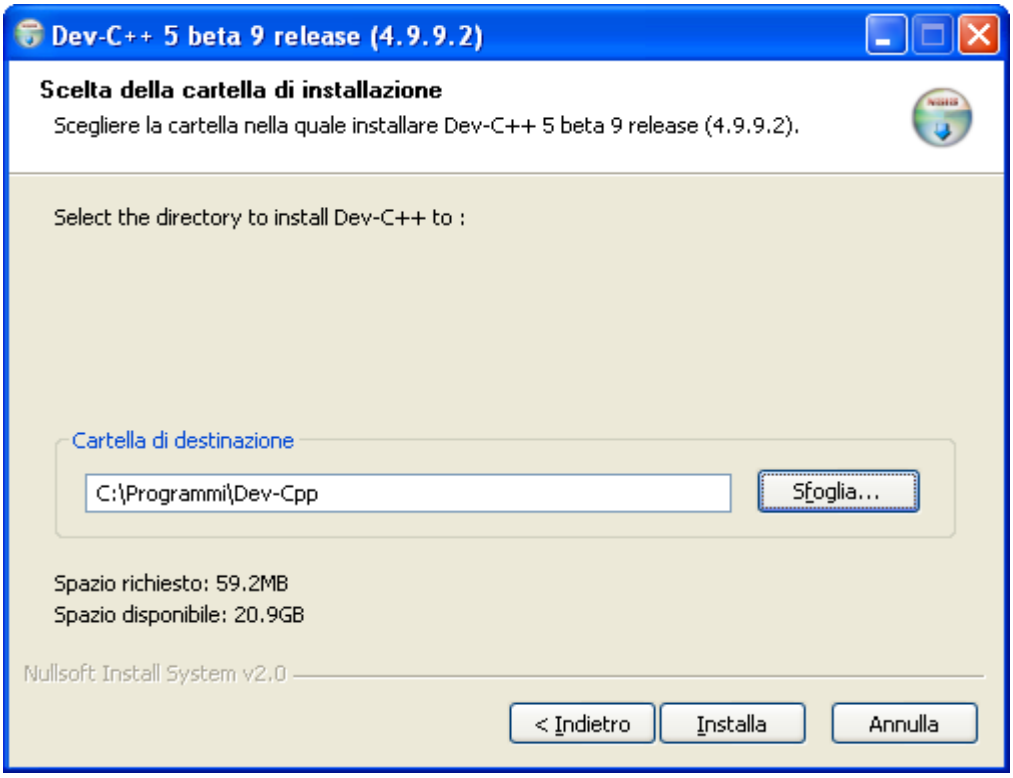


Fig. 2.3: Directory di installazione

Vengono estratti nella directory specificata i file compressi contenuti nel file di installazione e alla fine se tutto è andato bene compare la seguente finestra:

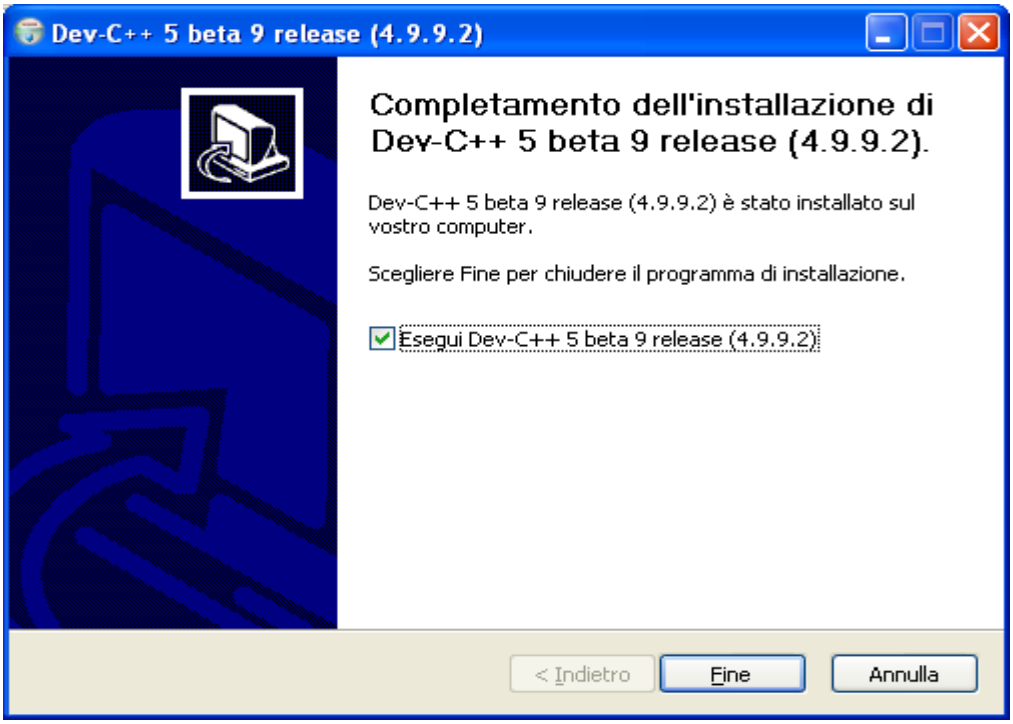


Fig. 2.4: Installazione completata

2.2 Configurazione

La configurazione iniziale di DevC++ consente di scegliere la lingua e l'aspetto grafico.

È inoltre possibile configurare i parametri di compilazione, di ambiente e personalizzare l'editor del codice sorgente.

2.2.1 Opzioni dell'ambiente

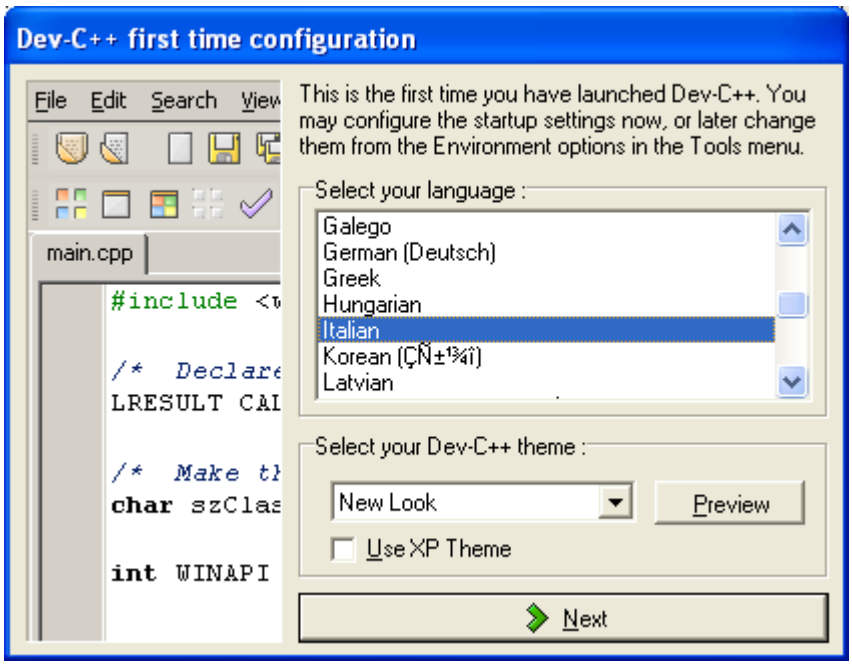


Fig. 2.5 *Configurazione dell'interfaccia durante l'installazione*

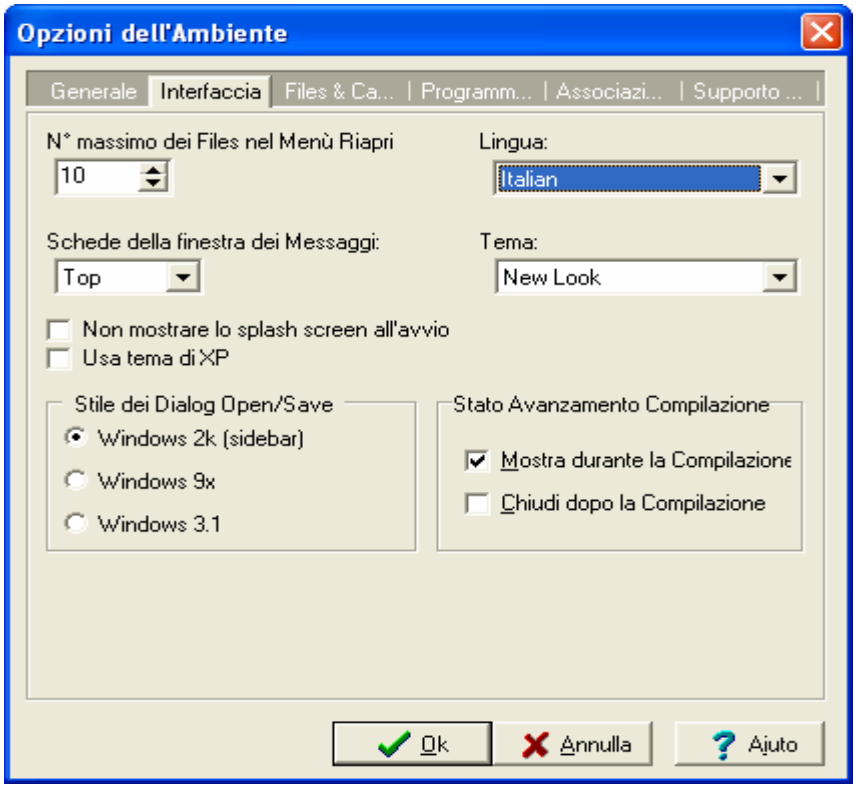


Fig. 2.6 *Configurazione dell'interfaccia successivamente all'installazione*

2.2.2 *Opzioni dell'editor*

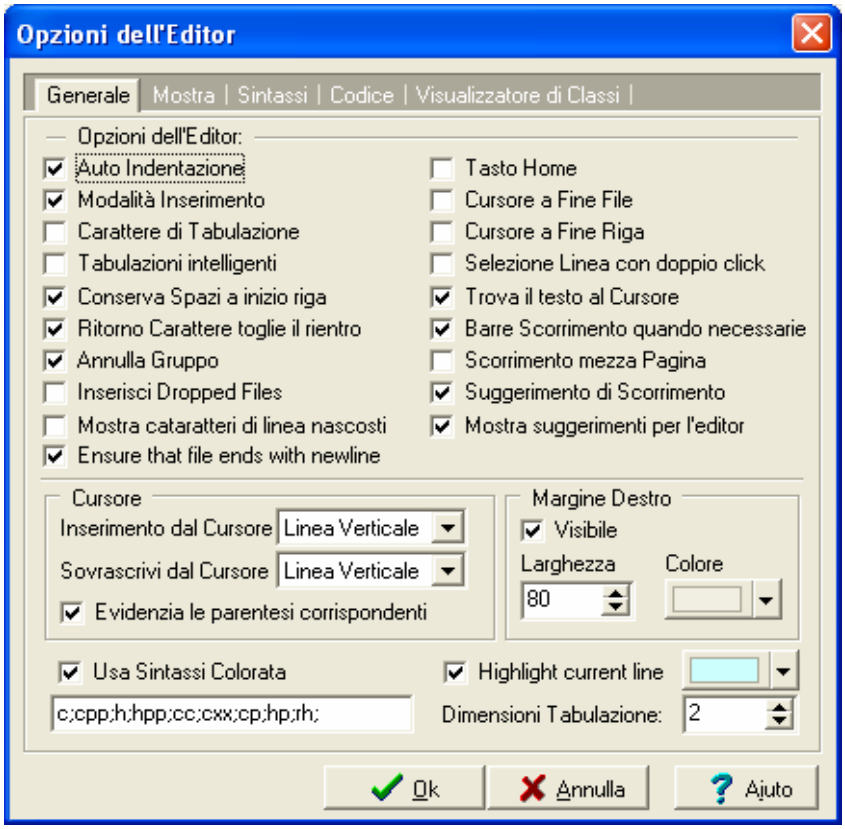


Fig. 2.7: Opzioni dell'editor (Riquadro Generale)

Preferibilmente:
Selezionare l'opzione "Evidenzia le parentesi corrispondenti" (come in figura)
Impostare la "Dimensione Tabulazione" con il valore 2 (come in figura)

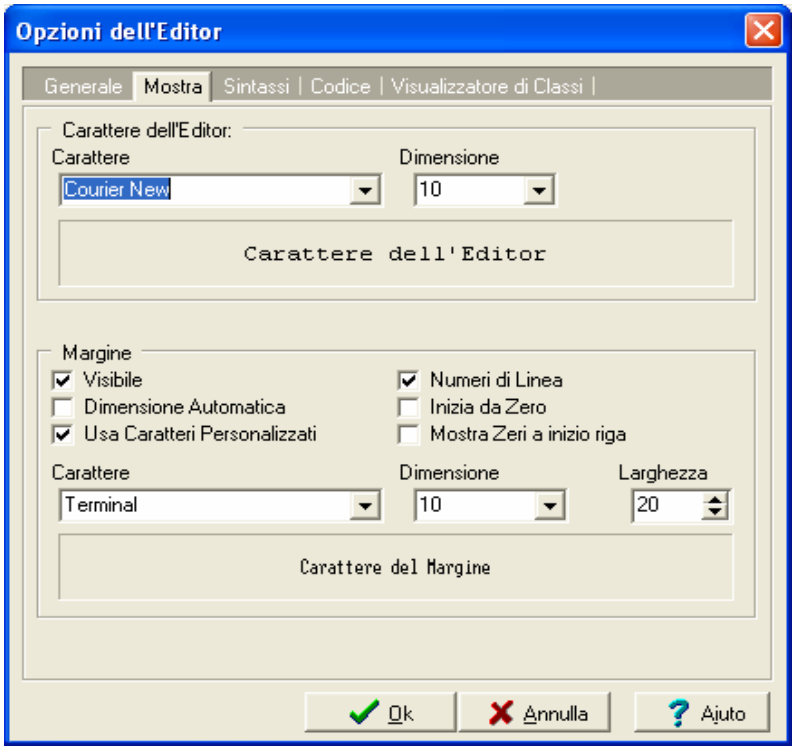


Fig. 2.8: Opzioni dell'editor (Riquadro Mostra)

Preferibilmente:
Selezionare "Numeri di linea"

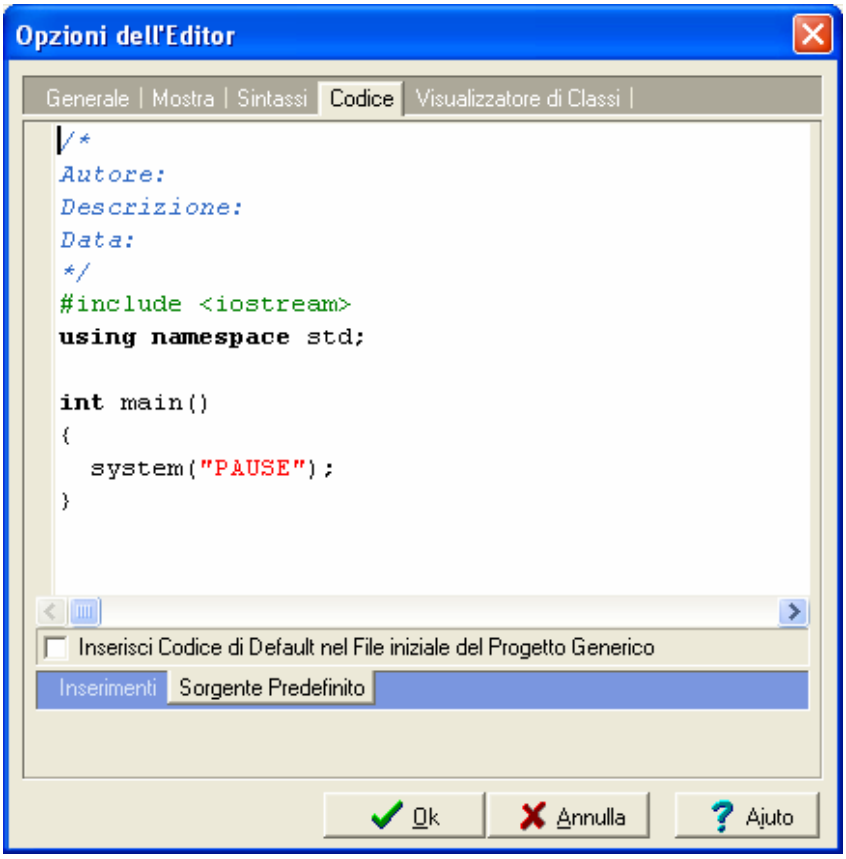
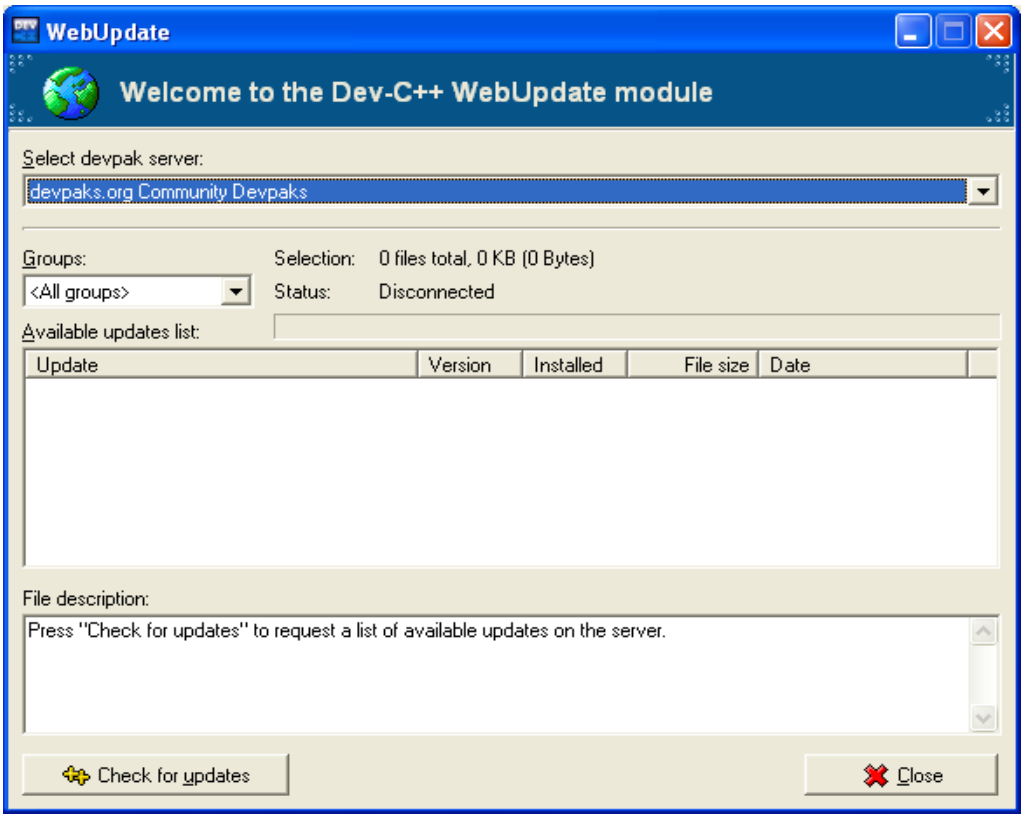


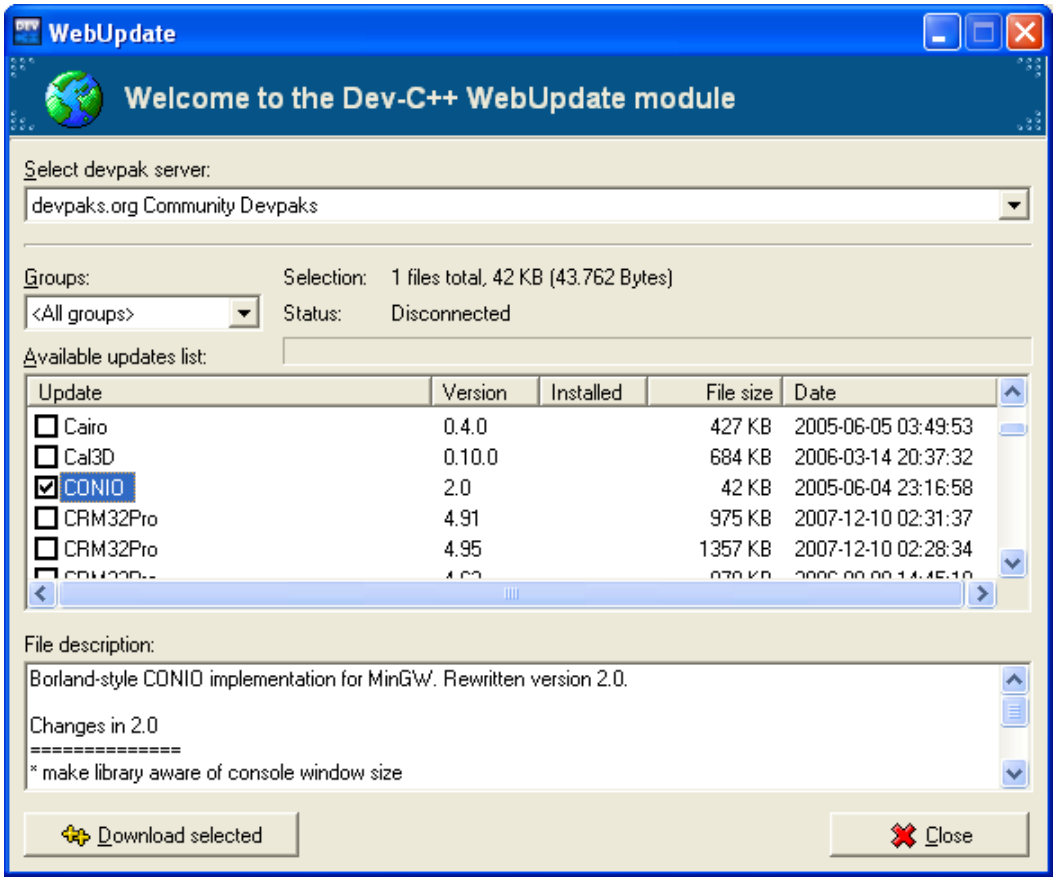
Fig. 2.9: Opzioni dell'editor (Riquadro Codice e Sorgente Predefinito)

Preferibilmente:
Scrivere il codice predefinito che apparirà ogni volta che si crea un nuovo file.

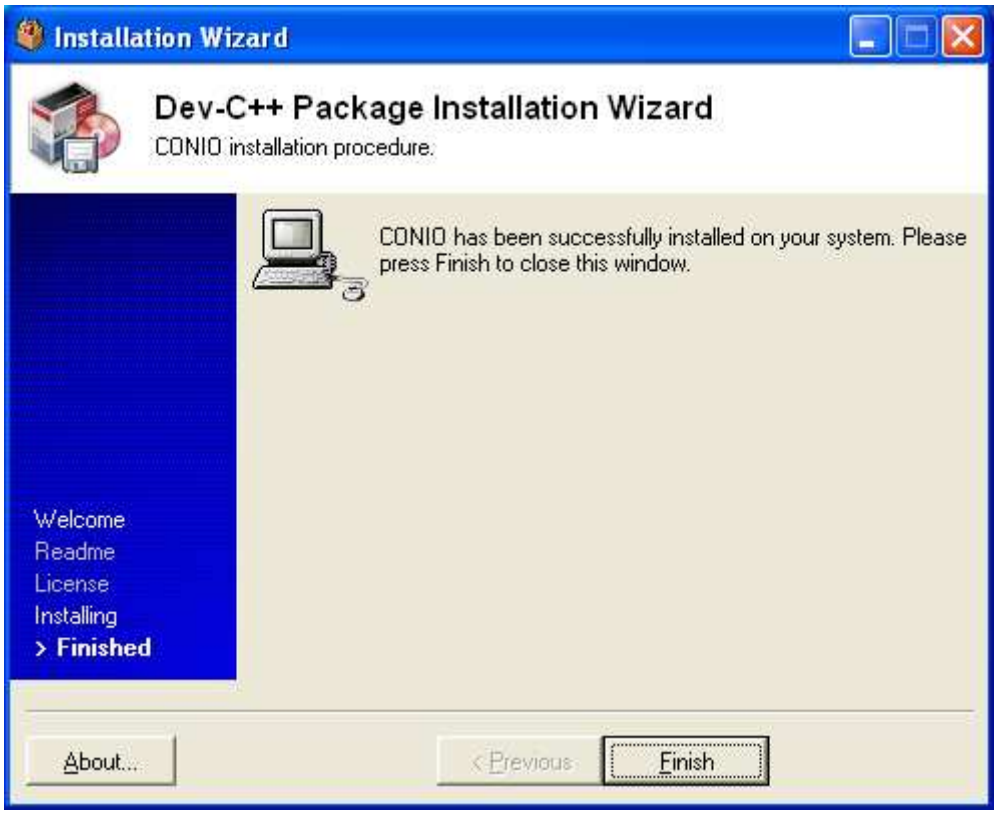
2.2.3 Scaricare le librerie ed eventuali aggiornamenti



scegliere il devpack server devpack.org e selezionare il pulsante "Check for updates"



selezionare la libreria CONIO e selezionare il pulsante “Download selected”
verrà scaricata la libreria e seguirà la procedura di installazione automatica.



3 Sessione di lavoro

Una tipica sessione di lavoro con un ambiente di sviluppo prevede una serie di azioni, spesso svolte in modo ciclico.

Durante una sessione di lavoro:

1. si crea un nuovo file (con estensione .cpp) o si apre un file esistente (con estensione .cpp);
2. si modifica il codice sorgente;
3. si salvano le modifiche apportate;
4. si effettua la compilazione;
5. in caso di esito positivo il programma viene eseguito, altrimenti si ritorna al passo 2.

Dal momento che un programma complesso, può essere costituito da più file, l'IDE prevede il concetto di progetto, ossia un gruppo di file di codice sorgente in linguaggio C o C++ (.c o .cpp) che costituisce il programma da compilare ed eseguire.

3.1 Realizzare una Applicazione

Nei casi più semplici si crea un nuovo file sorgente che conterrà il programma. Nei casi più complessi è necessario creare un progetto che contenga i vari files che costituiscono il programma.

3.1.1 *Creazione di un nuovo programma (file sorgente)*

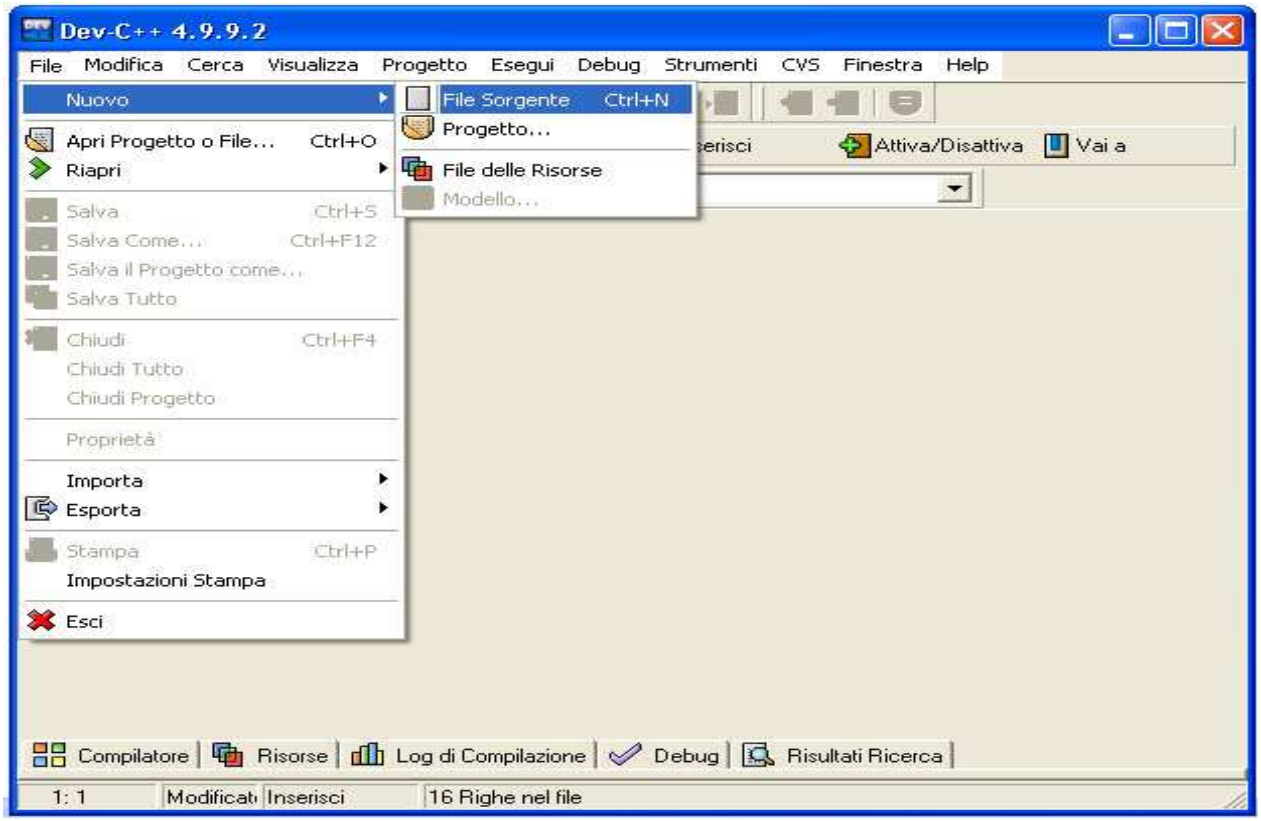


Fig. 3.1: Creazione di un nuovo programma

3.1.2 *Apertura di un file esistente*

L'apertura di un file o progetto esistente consente di modificare il codice sorgente.

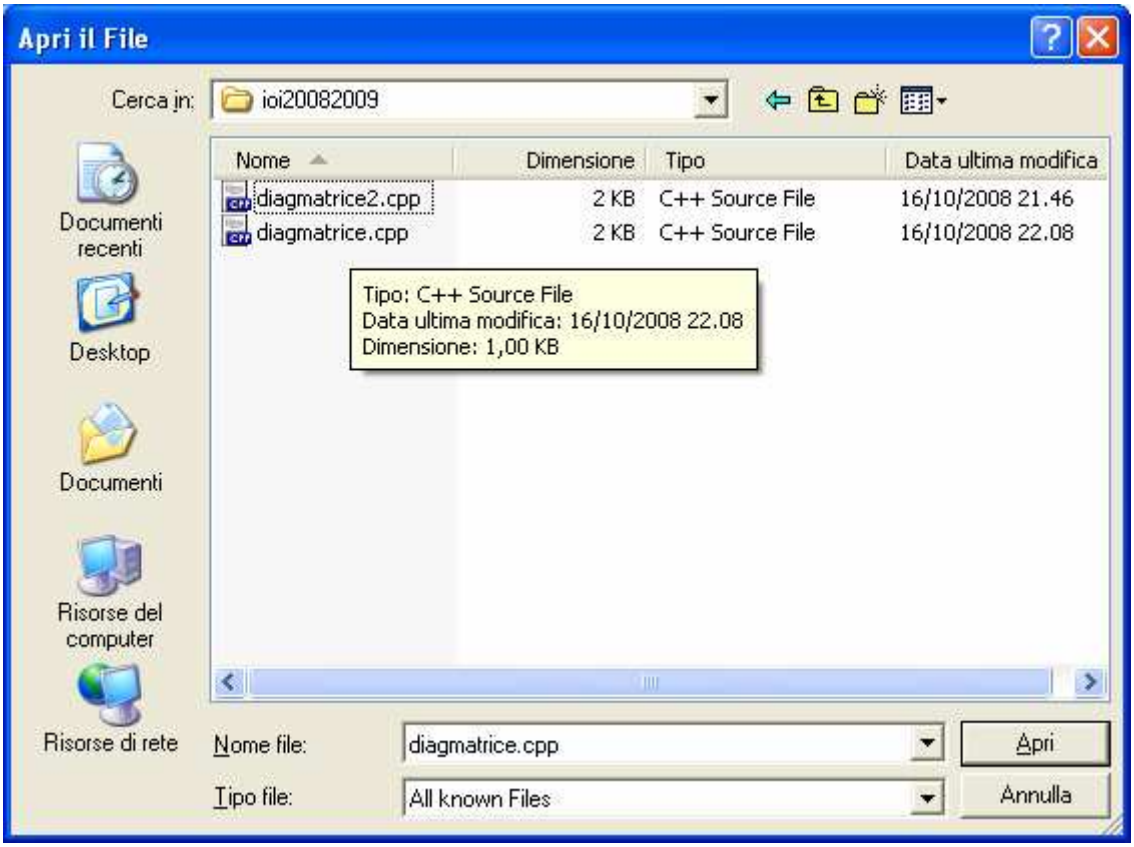


Fig. 3.3: *Apertura di un file esistente*

3.1.3 *Modifica del codice sorgente*

L'IDE dispone di un editor che, oltre ad offrire l'evidenziazione automatica delle parole chiave (keyword), permette di:

- effettuare ricerche e sostituzioni di testo;
- di spostarsi ad una determinata funzione del programma;
- di spostarsi ad una determinata riga (linea) del programma.

Queste funzionalità di ricerca/sostituzione testo sono accessibili dal menu *Cerca*.

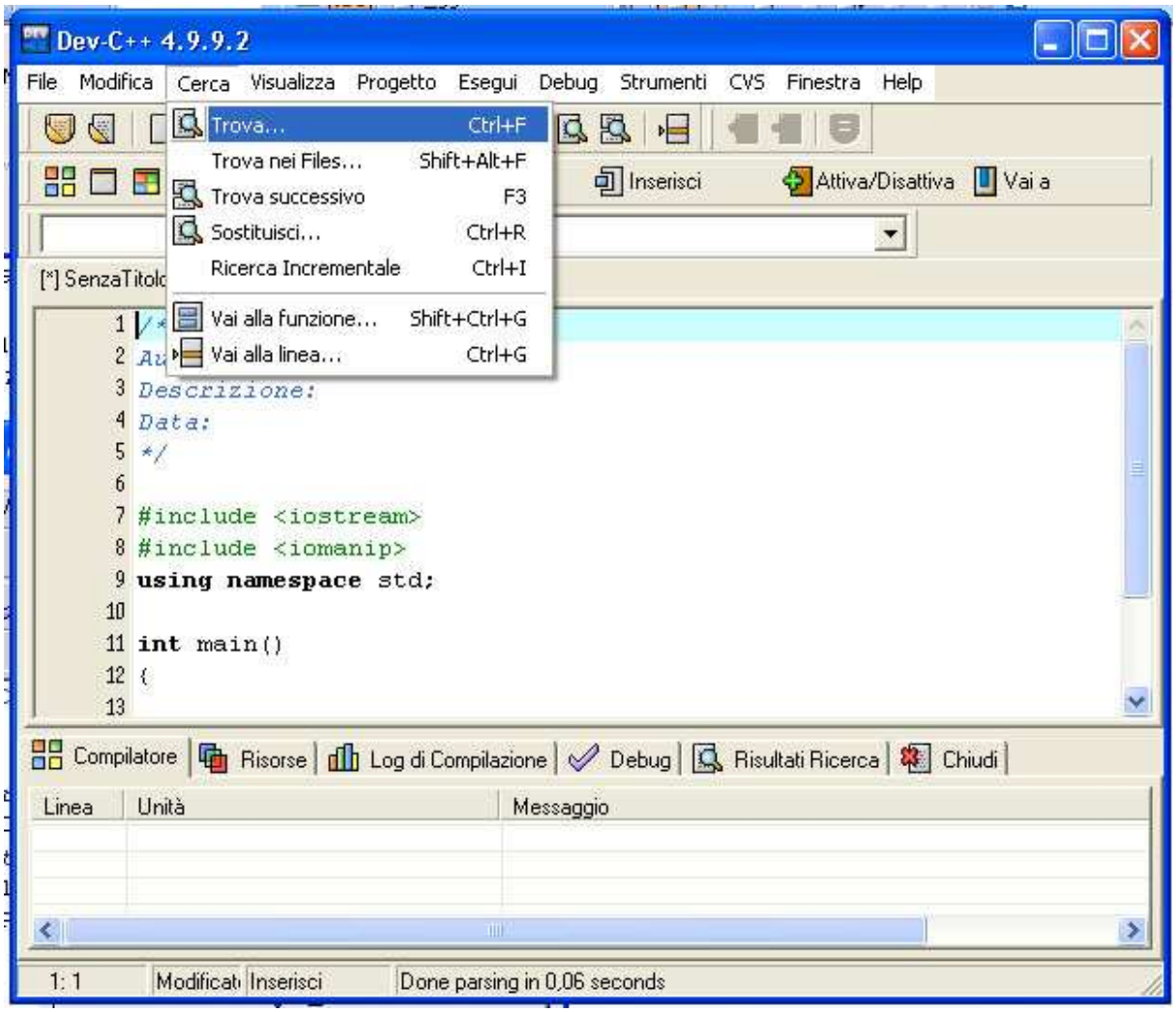


Fig. 3.4: *Ricerca di testo*

I risultati della ricerca in più files del progetto si trovano nella apposita scheda (tab) della finestra di output.

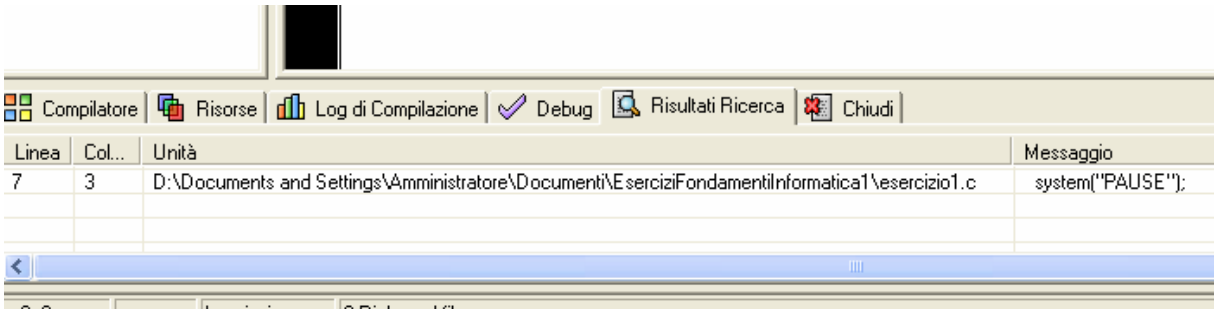


Fig. 3.5: *Risultati ricerca su più file*

3.2 Esecuzione del programma

3.2.1 Compilazione

La voce *Compila* consente di effettuare la compilazione ed il link del progetto, ossia dell'insieme di file di codice sorgente di cui è costituito.

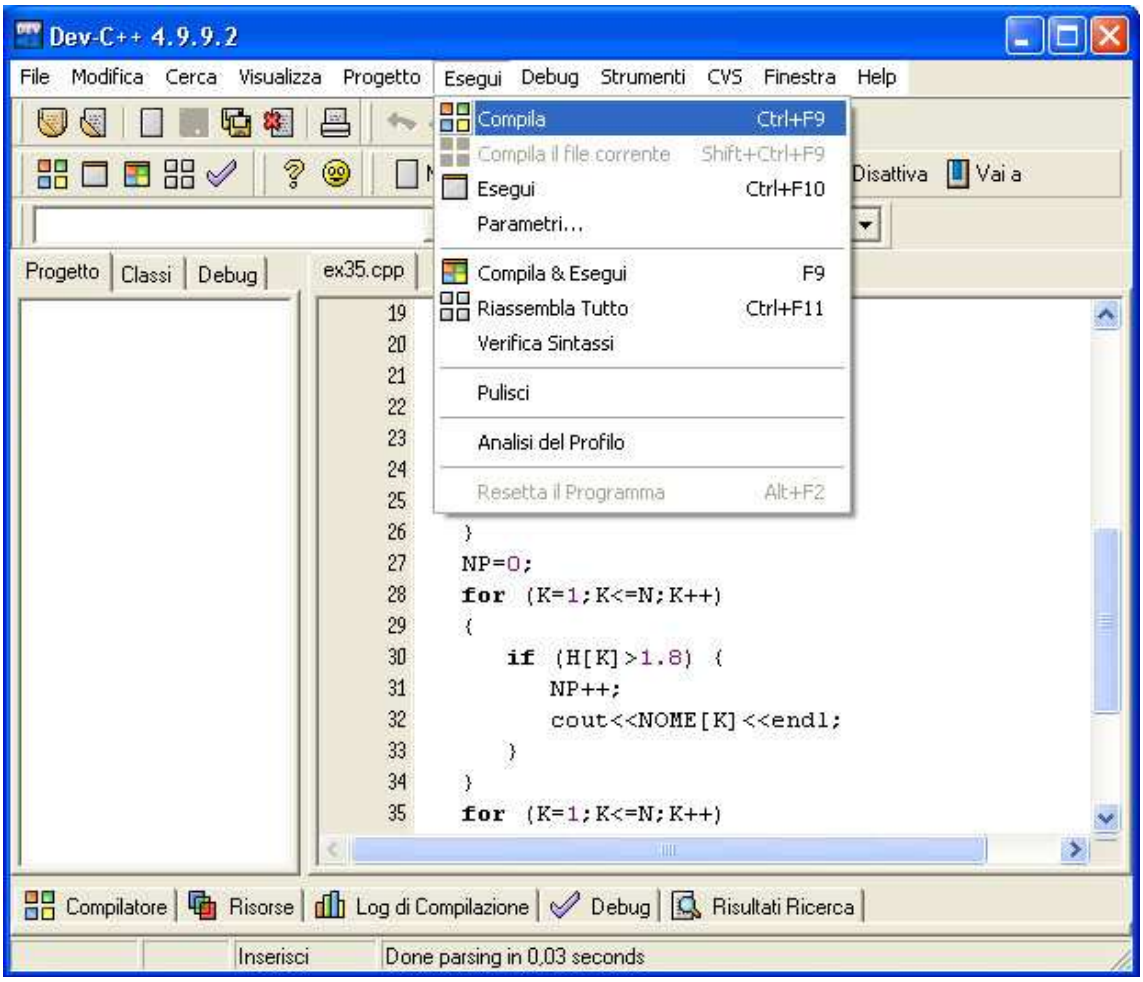


Fig. 3.6: *Compilazione*

Il comando comporta il salvataggio automatico del progetto.

Se la compilazione è avvenuta con successo compare la finestra di dialogo *Compile Progress* con status "Done.".

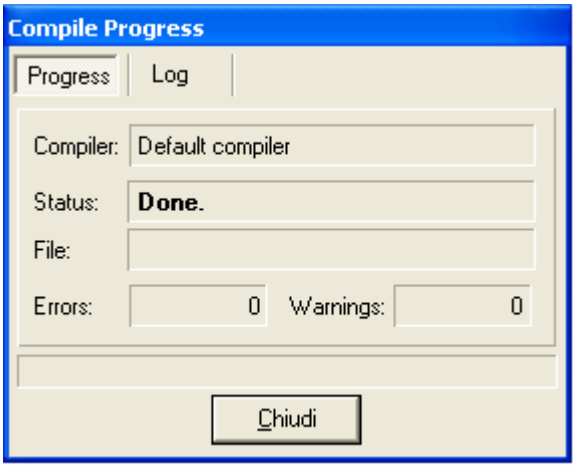


Fig. 3.7: *Progetto compilato*

Gli eventuali errori di compilazione sono presentati nella scheda (tab) *Compilatore*.

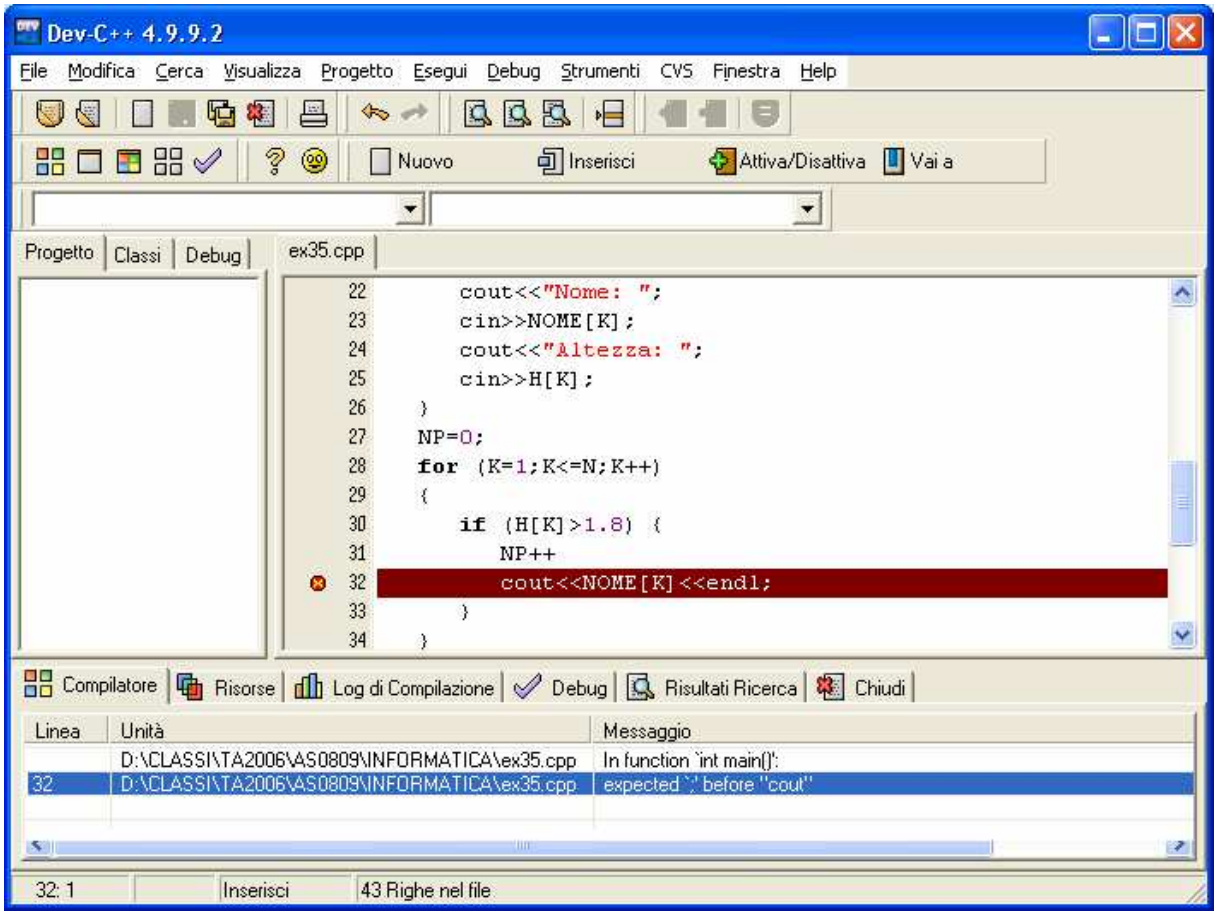


Fig. 3.8: *Scheda "Compilatore" con errori di compilazione*

Nella scheda (tab) *Log di compilazione* si possono vedere, indipendentemente dalla presenza di errori o meno, i risultati (log) dell'attività di compilazione.

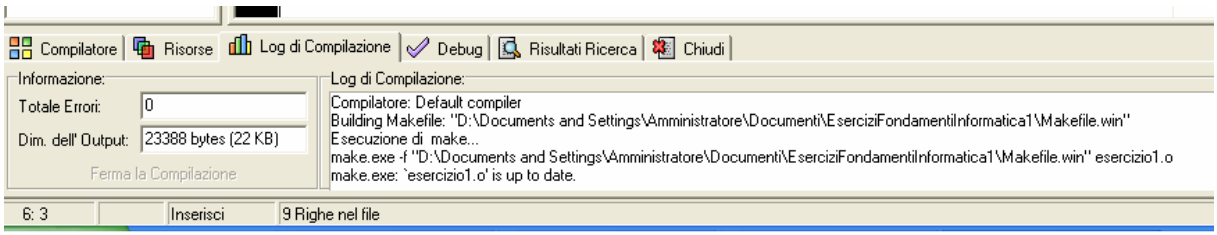


Fig. 3.9: *Scheda "Log di compilazione"*

3.2.2 *Compilazione, link ed esecuzione*

L'esecuzione del comando *Compila & Esegui* comporta nell'ordine:

1. compilazione del file sorgente es. *main.cpp* (generazione del file di codice oggetto *main.o*);
2. qualora siano presenti, compilazione dei rimanenti file sorgenti .cpp (o .c) componenti il progetto (generazione dei rispettivi file di codice oggetto *.o*)
3. link del progetto (generazione del file *progetto.exe*).

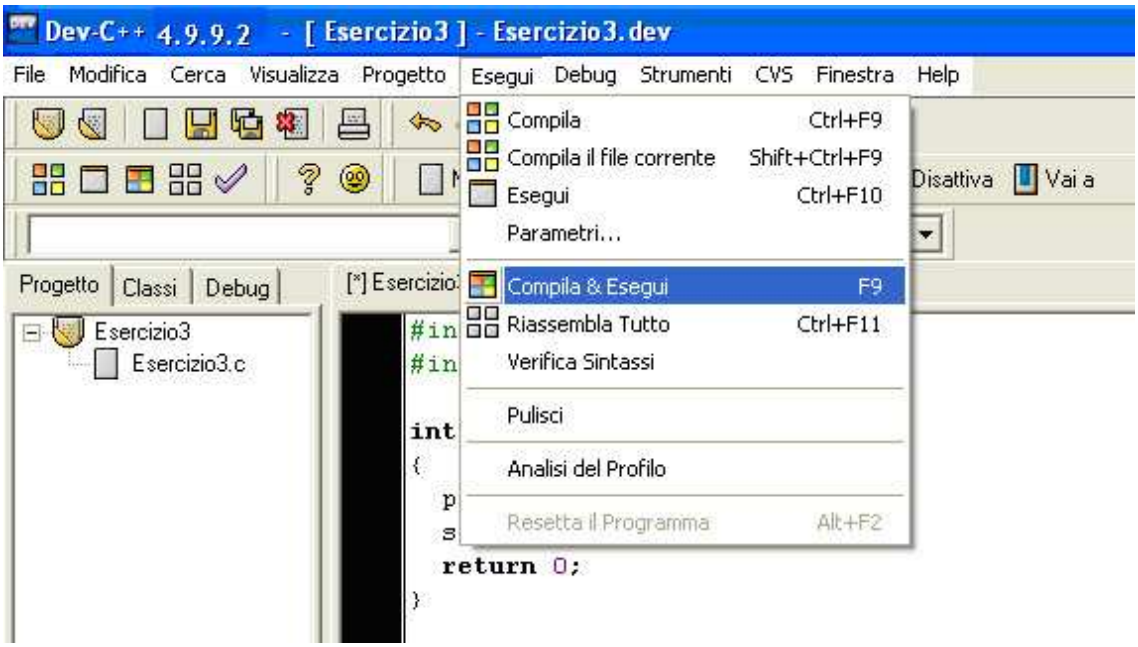


Fig. 3.10: *Compilazione, link ed esecuzione*

Il comando comporta il salvataggio automatico del progetto.

In presenza di errori il programma non viene eseguito.
In presenza di errori, nella scheda (tab) *Compilazione* si possono vedere le tipologie di errore riscontrate (compilazione, link e build).

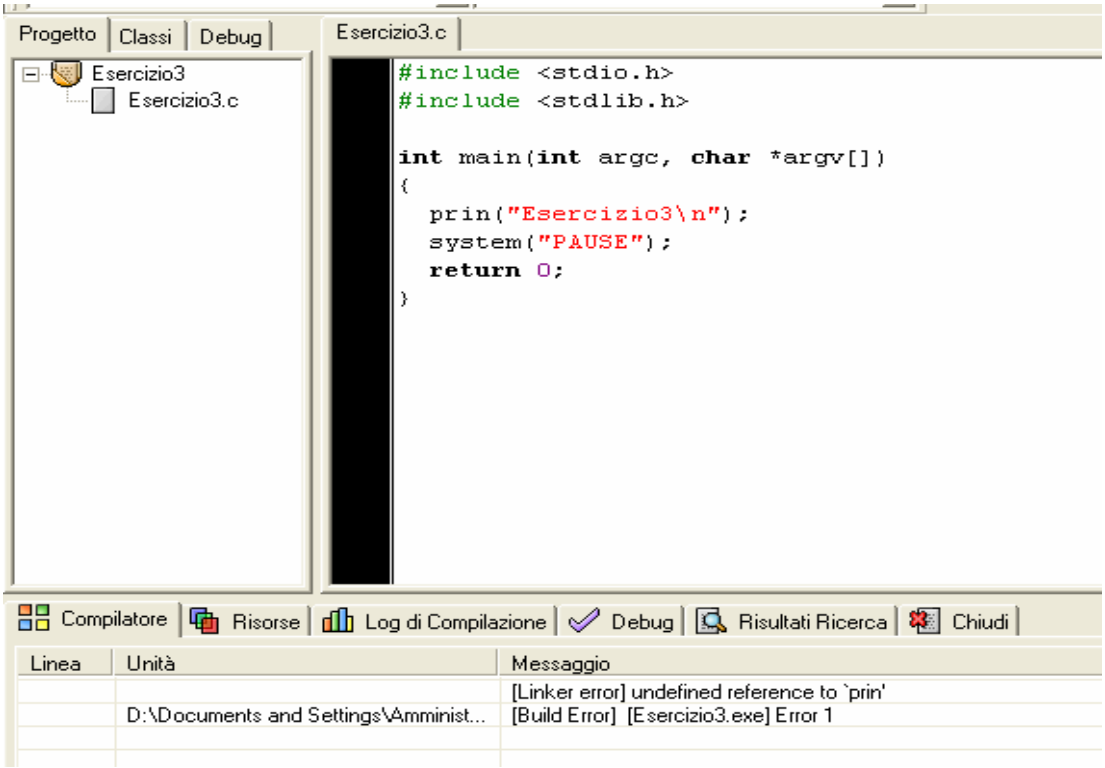


Fig. 3.11: *Scheda "Compilatore" con errori di link e build*

Nella scheda (tab) *Log di compilazione* si possono vedere i risultati (log) delle attività di compilazione e di link. In essa sono riportati gli eventuali errori generati.

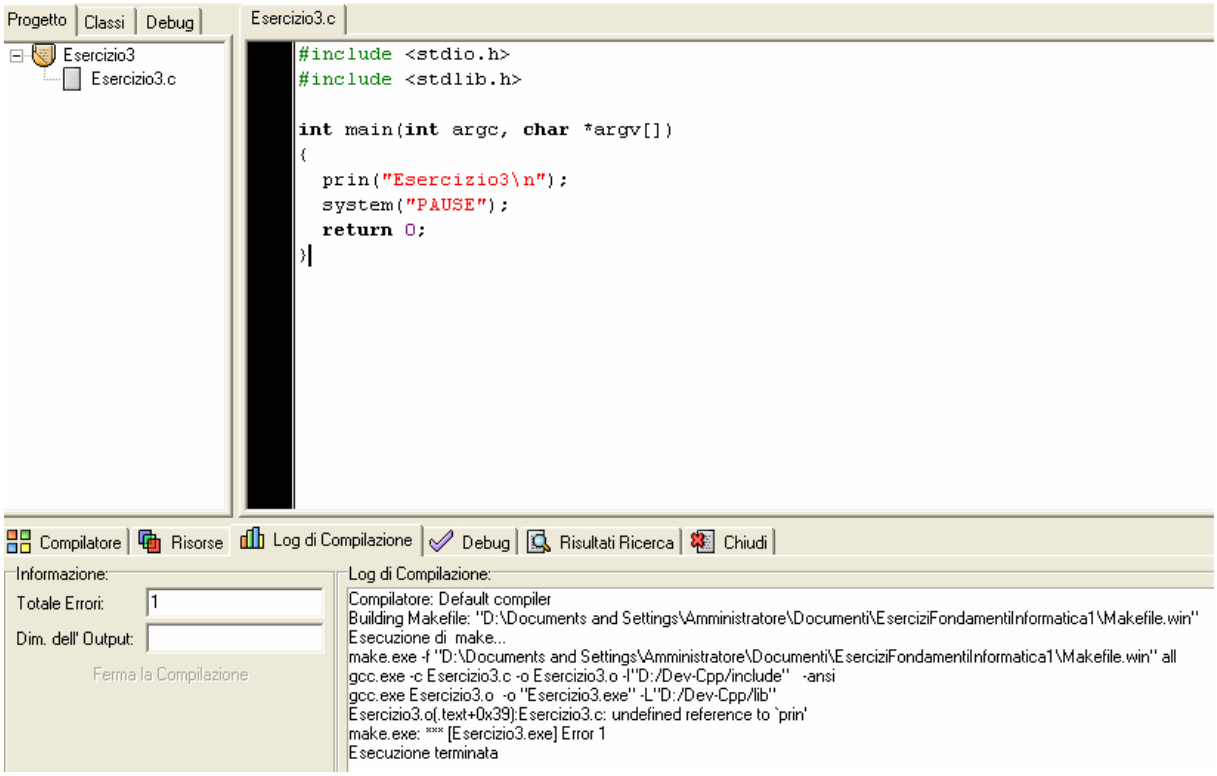


Fig. 3.12: *Scheda "Log di compilazione" con errori*

Con il comando *Pulisci* si elimina dalla directory corrente il file *.exe* generato. Si consiglia di utilizzare questo comando prima di una nuova azione di compilazione ed esecuzione.

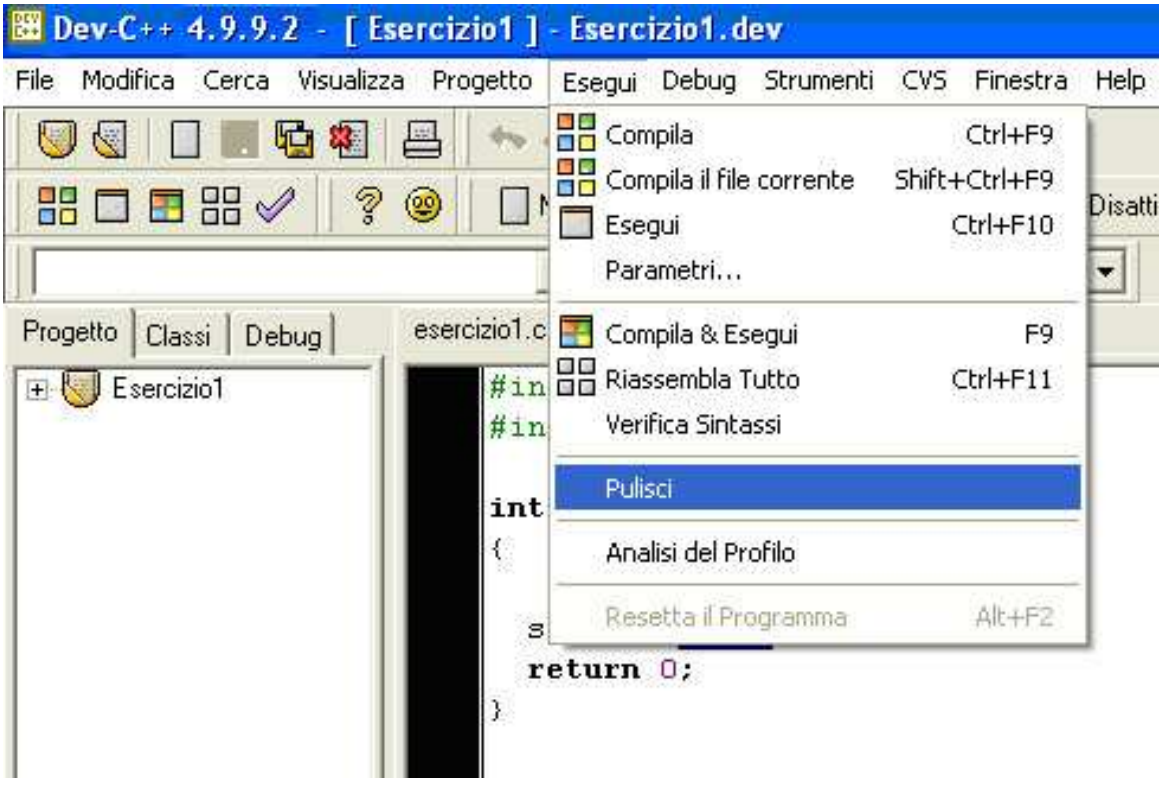


Fig. 3.13: *Eliminazione dell'eseguibile generato*

3.2.2 Esecuzione con parametri

È possibile parametrizzare l'esecuzione di un programma attribuendo un valore ai parametri della funzione main.

int main (int argc, char *argv[])

Per specificare il valore di tali parametri si deve richiamare il comando *Parametri...* dal menu *Esegui*. I parametri (stringa) devono essere separati da spazio.

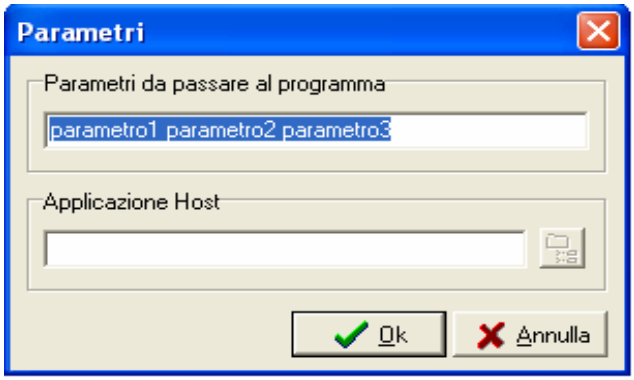


Fig. 3.14: *Specifica di parametri nella funzione main()*

3.3 Salvataggio del codice

I comandi di salvataggio sono contenuti nel menu *File*.

Le opzioni di salvataggio sono 4:

- salvataggio del file sorgente .C corrente (comando *Salva*);
- salvataggio con rinomina del file sorgente .C corrente (comando *Salva Come*);
- salvataggio con rinomina del progetto (.dev) corrente (comando *Salva il Progetto come...*);
- salvataggio di tutti i file aperti (*Salva Tutto*).

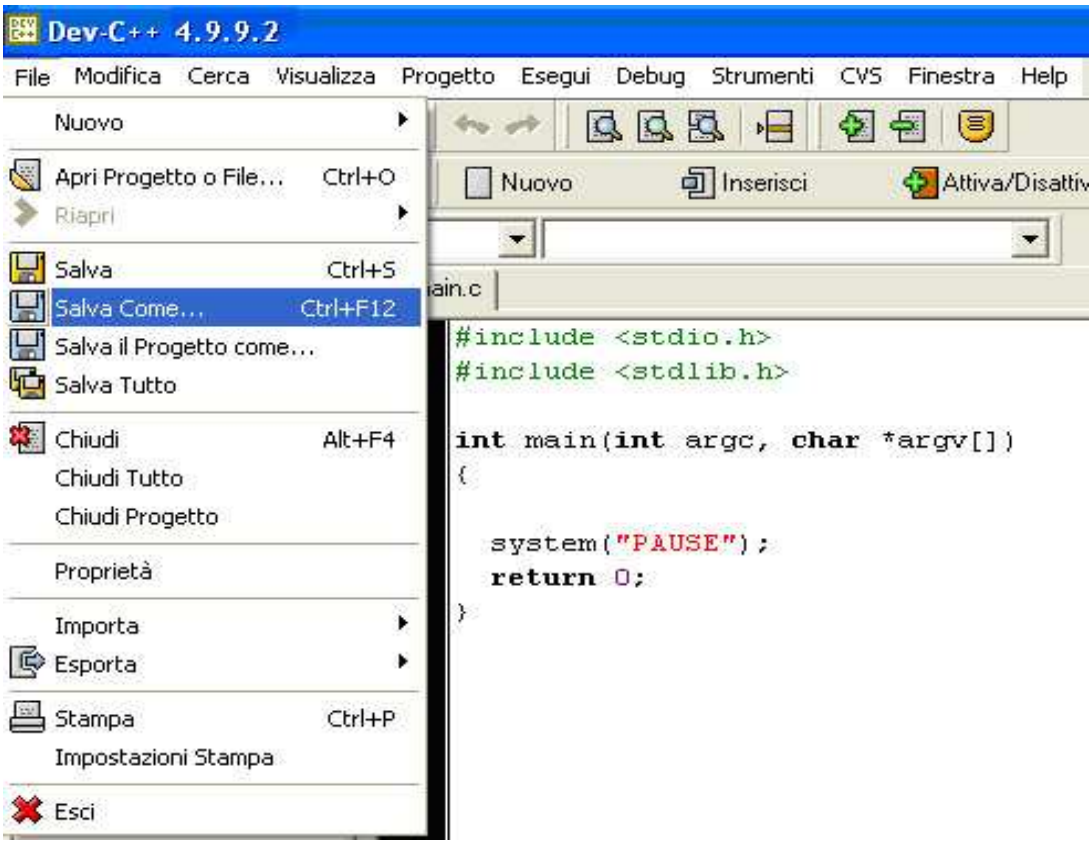


Fig. 3.15: *Salvataggio*

Il salvataggio con nome prevede di specificare la directory in cui il file deve essere salvato.

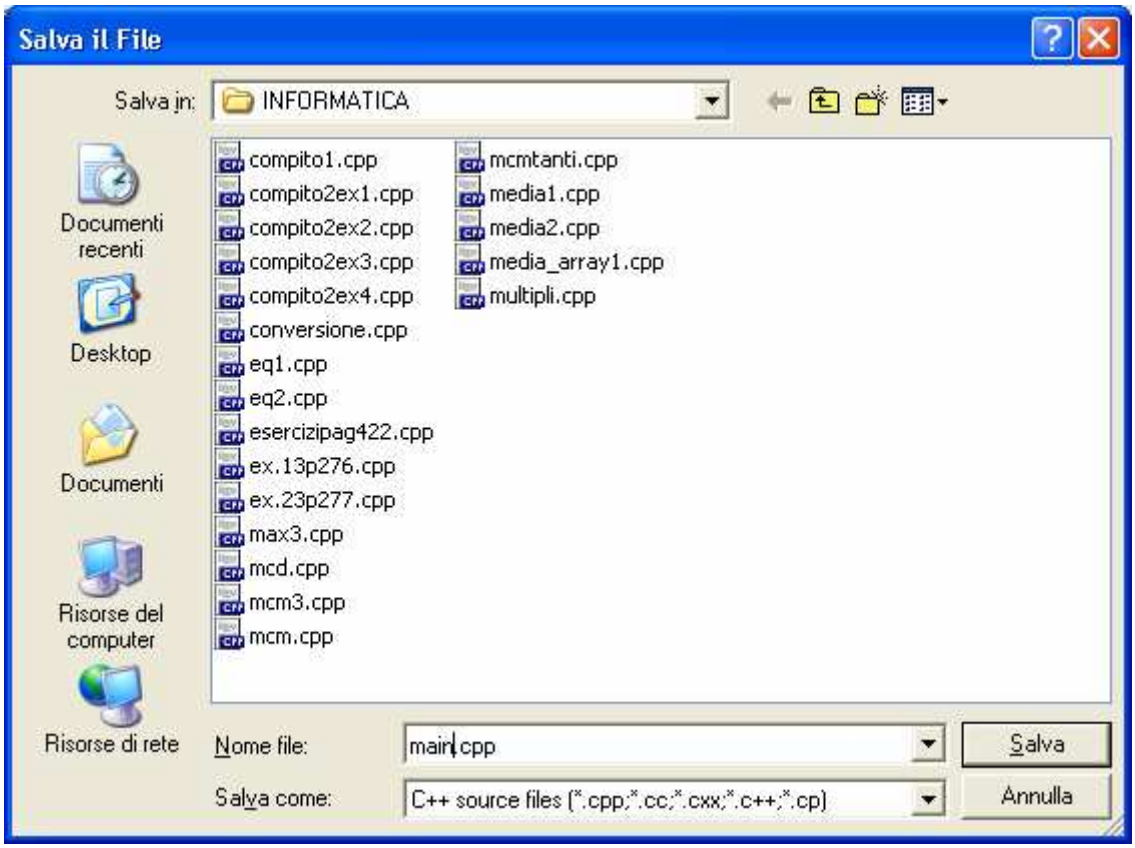


Fig. 3.16: *Salvataggio con nome di un file sorgente .cpp*

4 Alcuni riferimenti utili

4.1 Help in linea

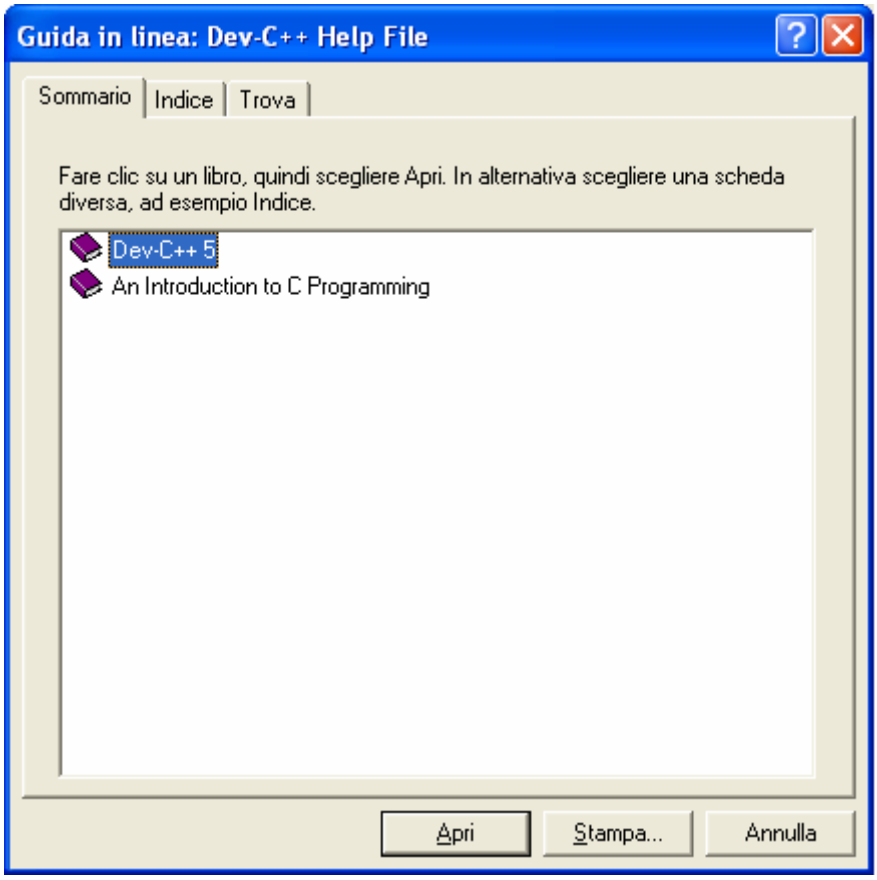


Fig. 4.1: *Help in linea*

4.2 Bloodshed.net

È il sito ufficiale di presentazione del compilatore DevC++ in lingua inglese.

http://www.bloodshed.net/dev/devcpp.html

4.3 SourceForge.NET

È un sito di sviluppo in inglese che tratta anche del compilatore DevC++.

Dalla pagina http://sourceforge.net/projects/dev-cpp/ è possibile:

- scaricare il compilatore;
- scaricare aggiornamenti (patches & bug fixes);
- scaricare informazioni sulle precedenti versioni del compilatore (release notes);
- consultare forum;
- consultare l'elenco dei bugs;
- consultare news sullo stato della documentazione, sulle versioni, sulla soluzione di problemi, ecc..

4.4 Forum di sviluppatori

È uno forum di sviluppatori DevC++ in lingua italiana.

http://forum.redangel.it/