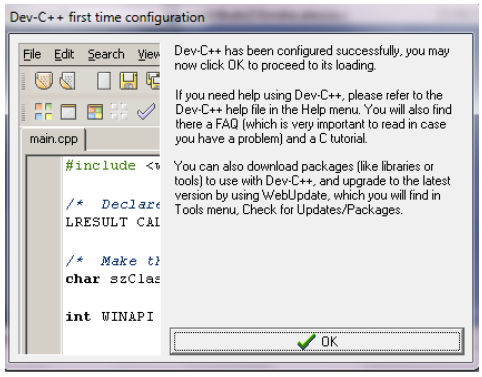
Tecniche della Programmazione: Note sull'uso del DEVC++

PRIMA COSA:
DEVC++ permette di programmare in C++ ed in C. Non e' esattamente la medesima cosa, come vedremo durante il corso.
Ricordiamoci che noi programmiamo in C!

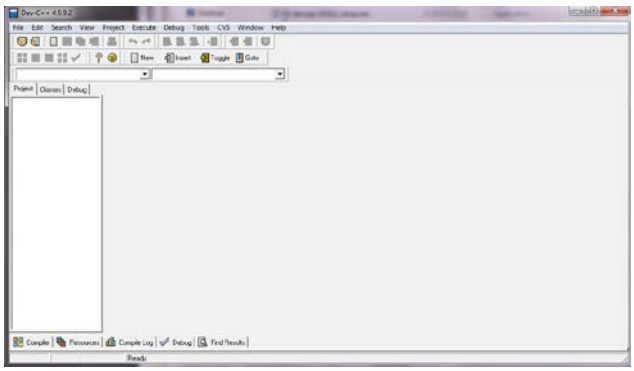
Dev-C++ e' un Ambiente Integrato di Sviluppo (Integrated Development Environment – IDE) per C (e C++). Sito Web: http://www.bloodshed.net/dev/devcpp.html.
Funziona in ambiente Windows. Il programma di istallazione e' scaricabile da:
http://www.dis.uniroma1.it/~marte/homepage/didattica/tdp-latina/esercitazioniGuidate/



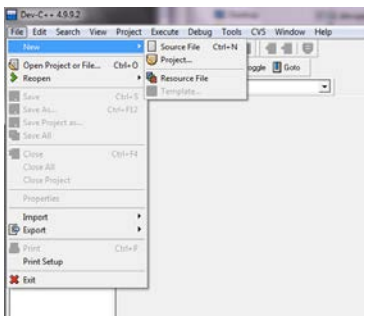
Dopo l'istallazione (durante la quale vengono chiesti alcuni permessi e proposte alcune alternative … si puo' dire "Yes" a tutto senza pericoli).
La prima cosa richiesta e' in quale lingua vogliamo sia scritta l'interfaccia (cioe' siano scritti i vari nomi che vediamo nelle finestre e nei menu': facciamo inglese?
Finita l'istallazione, quando si avvia DEVC++ per la prima volta, viene eseguita una configurazione … anche qui diciamo "yes" se richiesti.

Il DEV permette la creazione di progetti, cioe' di programmi complessi composti da diversi file prodotti dal programmatore. Non parliamo qui di progetti perche' all'inizio non serve. Piu' avanti ne parleremo, per permettere lo sviluppo di librerie di strutture dati.
Quel che ci interessa ora e' che avviando DEV si entra in un ambiente che permette

- Editing di un programma (per ora, come detto sopra, ci limitiamo ad un programma contenuto in un singolo file)
- Compilazione del programma
- Linking del programma con le funzioni di libreria
- Esecuzione del programma eseguibile



Questo e' quel che si vede avviando il DEV:
il menu' FILE permette di gestire i file che contengono i programmi (aprire un file, salvare le correzioni fatte editando il file, salvare il programma).
Il menu' Tools contiene alcune funzionalita' di configurazione che ci saranno utili …

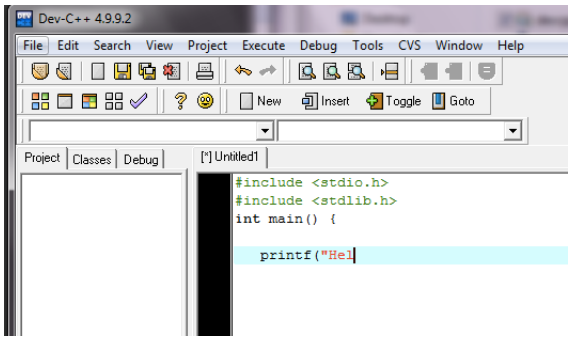


Si puo' scrivere il primo nostro programma scelgliendo NEW nel menu' FILE, e specificando SOURCE FILE (cioe' file che conterra' un "programma sorgente" – sniff, "sorgente", "programma sorgente", "source code" sono vecchi modi romantici di chiamare il programma …)

Scelta questa opzione, il DEV ci mostra l'area di lavoro, in cui scriveremo il programma; come si vede in una linguetta in alto a sinistra, il nome del file in cui potrebbe essere salvato il programma, al momento e' Untitled1 (in realta' si sottindende Untitled1.cpp: non ci va bene nulla di tutto cio', adesso vediamo).

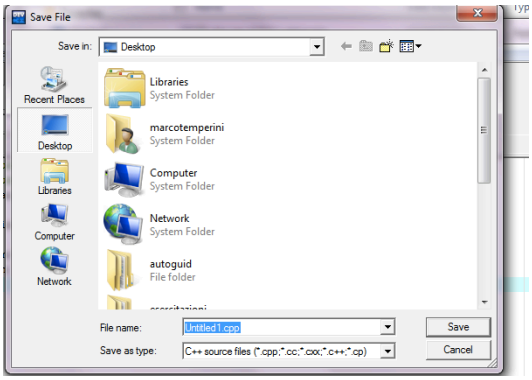
Cominciamo ad editare il nostro primo … e' un classico: useremo questo codice:

```
#include <stdio.h>
#include <stdlib.h>
int main() {

   printf("Hello World!");

system("PAUSE");
return 0;
}
```

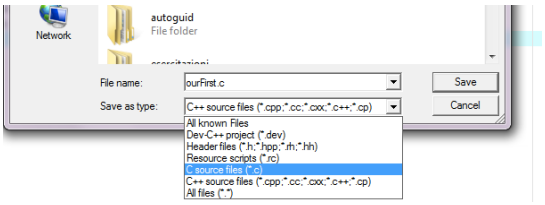


Per generare le parentesi graffe { e } nelle tastiere italiane …
- <Alt> 123 genera {
- <Alt> 125 genera }
- <Alt> 126 genera ~
- In generale, <Alt> numeri_del_codice_ASCII_di_un_carattere, genera la stampa del carattere …

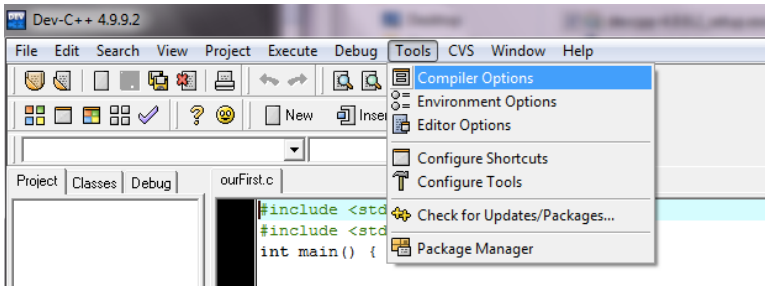
In figura sopra, il programma non e' completo, ma facciamo lo stesso l'operazione di salvataggio:
- menu' FILE,
- funzionalita' SAVE AS,
- In "File Name", specifichiamo un nome serio per il file: ourFirst.c
- notiamo che stiamo dando a questo file un nome con estensione ".c" perche' questa e' l'estensione dei programmi C.
- In "Save as Type" scegliamo "C source File"

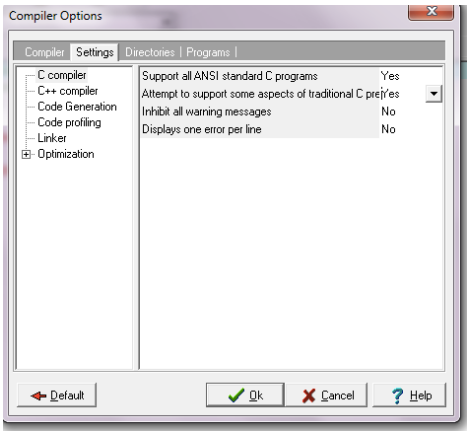


Riguardo all'estensione, se lasciassimo fare al DEV, verrebbe usata l'estensione C++, il che condurrebbe, dopo, il sistema a pensare che il programma sia scritto in C++: NO, questo non lo vogliamo.



Un'altra cosa: noi programmiamo in C, quindi vogliamo che il DEV sia configurato in modo da accettare, compilare ed eseguire bene i nostri programmi, secondo quanto prescritto dallo standard C.
Per eseguire questa configurazione dobbiamo andare nel menu' TOOLS, scegliere le opzioni di compilazione e selezionare quelle che ottengano il suddetto obiettivo.



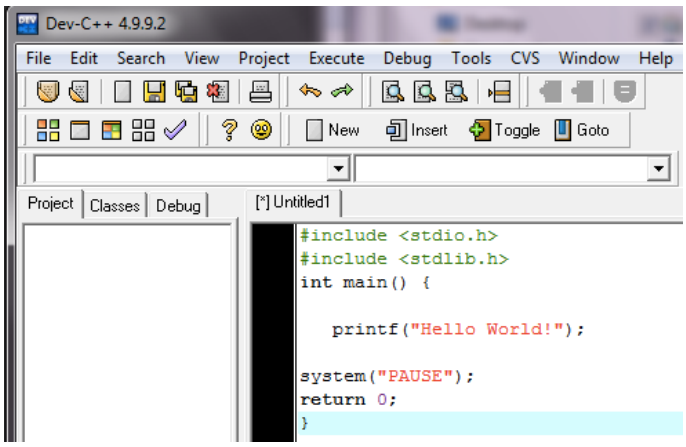


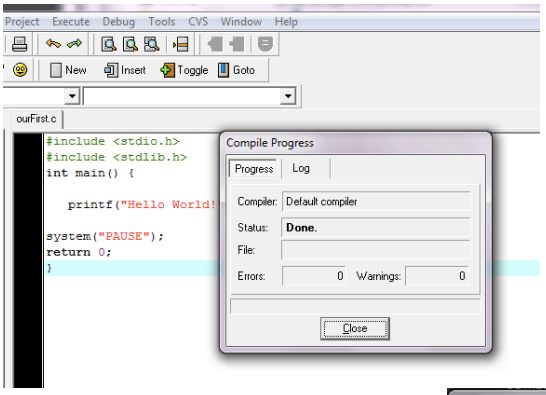
Riguardo al programma, si noti che

- Usiamo la direttiva include per includere nel programma le funzioni contenute nella libreria di I/O standard del C (stdio.h). Queste sono le funzioni che permettono di inserire dati (INPUT) da tastiera, e vedere l'OUTPUT del programma sul video (in una bella schermata nera vecchio stile). Lo facciamo perche' nel programma vogliamo usare una funzione di standard output (printf()).
- Usiamo la direttiva include per includere nel programma le funzioni contenute nella libreria stdlib.h. Lo facciamo perche' nel programma vogliamo usare una funzione system(), con parametro "PAUSE", per fare in modo che la finestra di output (quella nera menzionata prima) rimanga aperta per farci vedere l'output e non si chiuda appena terminata l'esecuzione del programma.
- Il programma principale (cioe' l'unico programma su cui stiamo lavorando) e' rappresentato dalla speciale funzione main. Si scrive cosi' come mostrato: main e' una funzione intera, priva di parametri (per quanto di nostro interesse) il cui bocco di codice contiene le istruzioni del programma.
- Notare come elegantemente la '{' di apertura del blocco e' sulla stessa linea di int main(), mentre la chiusura '}' e' isolata nell'ultima riga.
- L'istruzione `printf("Hello World!");` esegue la funzione printf, definite nella libreria stdio.h, con il risultato di emettere in output ("stampare", cioe' visualizzare, sullo schermo) la sequenza di caratteri `Hello World!`

Terminata la scrittura del programma si passa alle fasi successive, che conducono all'esecuzione, anche per vedere se il programma e' scritto bene e funziona.

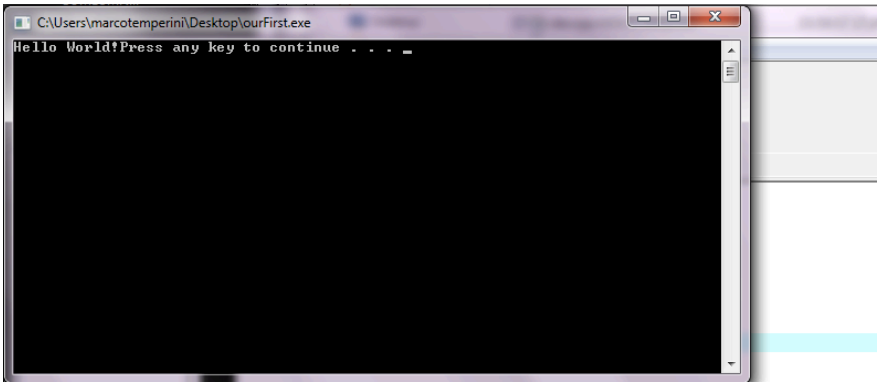
Nella seconda riga di icone, la prima corrisponde a "COMPILE". Clickandola si avvia la compilazione del programma. Se la compilazione va a buon fine (Done), viene eseguito anche il linking e prodotto un file ourFirst.exe, eseguibile (anzi, rilocabile per essere eseguito …).





La seconda icona e' quella "RUN" e permette di eseguire il programma. (La terza icona e' "COMPILE AND RUN", che esegue i due precedenti passi in sequenza, se possibile, cioe' se non ci sono errori di compilazione).

Con "RUN, si apre la finestra di output e si vede l'esecuzione del programma …

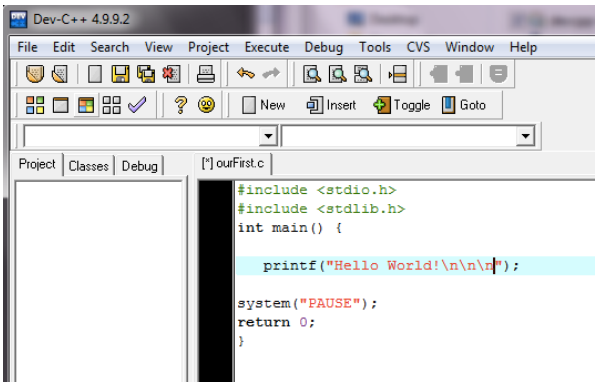


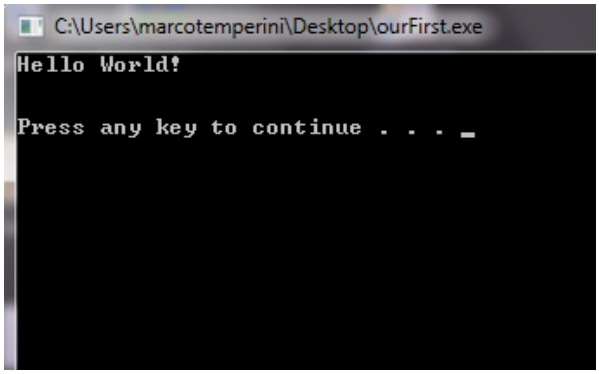
L'output di `printf("Hello World!");` appare nella prima riga della finestra (prima non c'era altro output prodotto dal programma.

L'output di `system("PAUSE");` viene emesso subito dopo (ed e' la comunicazione implicita che il programma e' stato sospeso, in attesa di un qualunque input da parte dell'utente. L'utente e' chi sta guardando l'esecuzione del programma … il "qualsiasi input" e' qualsiasi cosa (anche niente) seguita da RETURN. RETURN e' il modo che ha l'utente per dire che dalla tastiera e' in arrivo un input per il programma. Quando viene premuto RETURN, il programma riceve l'input che stava aspettando ed esegue l'istruzione successiva (cioe' la terminazione del programma).

… acc. Viene tutto attaccato. Che schifo!!

Modifichiamo il programma in modo che vengano "stampate" tre andate a capo dopo "Hello World!", e che quindi l'output sia un po' piu' leggibile.





Ora l'esecuzione e' perfetta.

E abbiamo finito.