

# Lab Info

A.A. 2024/25



**POLITECNICO**  
MILANO 1863

# Organizzazione

5 Lab venerdi mattina **8:30-11.15 AULA: 21.0.1**

vene 18/10 lab1

vene 8/11 lab2

vene 22/11 lab3

vene 29/11 lab4

vene 6/12 lab5

# Obiettivi

Comprendere e risolvere problemi con gli strumenti dell'informatica

→ Implementare programmi in linguaggio C

Lo scopo del laboratorio è mettere in pratica quello che avete imparato.

# Ambienti di sviluppo

## windows

- CodeBlock
- Clion
- Visual studio

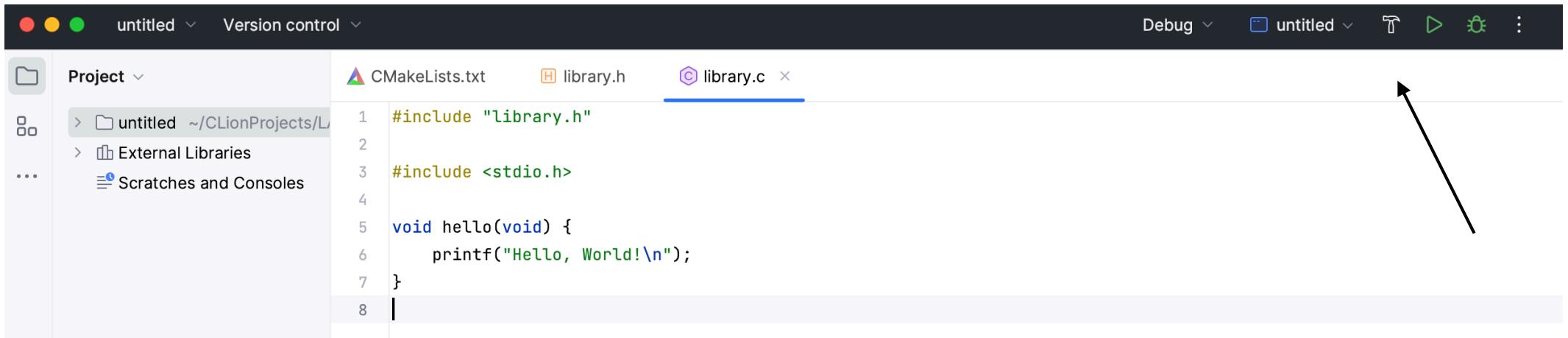
## Mac

Xcode

Clion

*In Allegato a fine slide le singole installazioni*

# Ambiente di sviluppo CLion



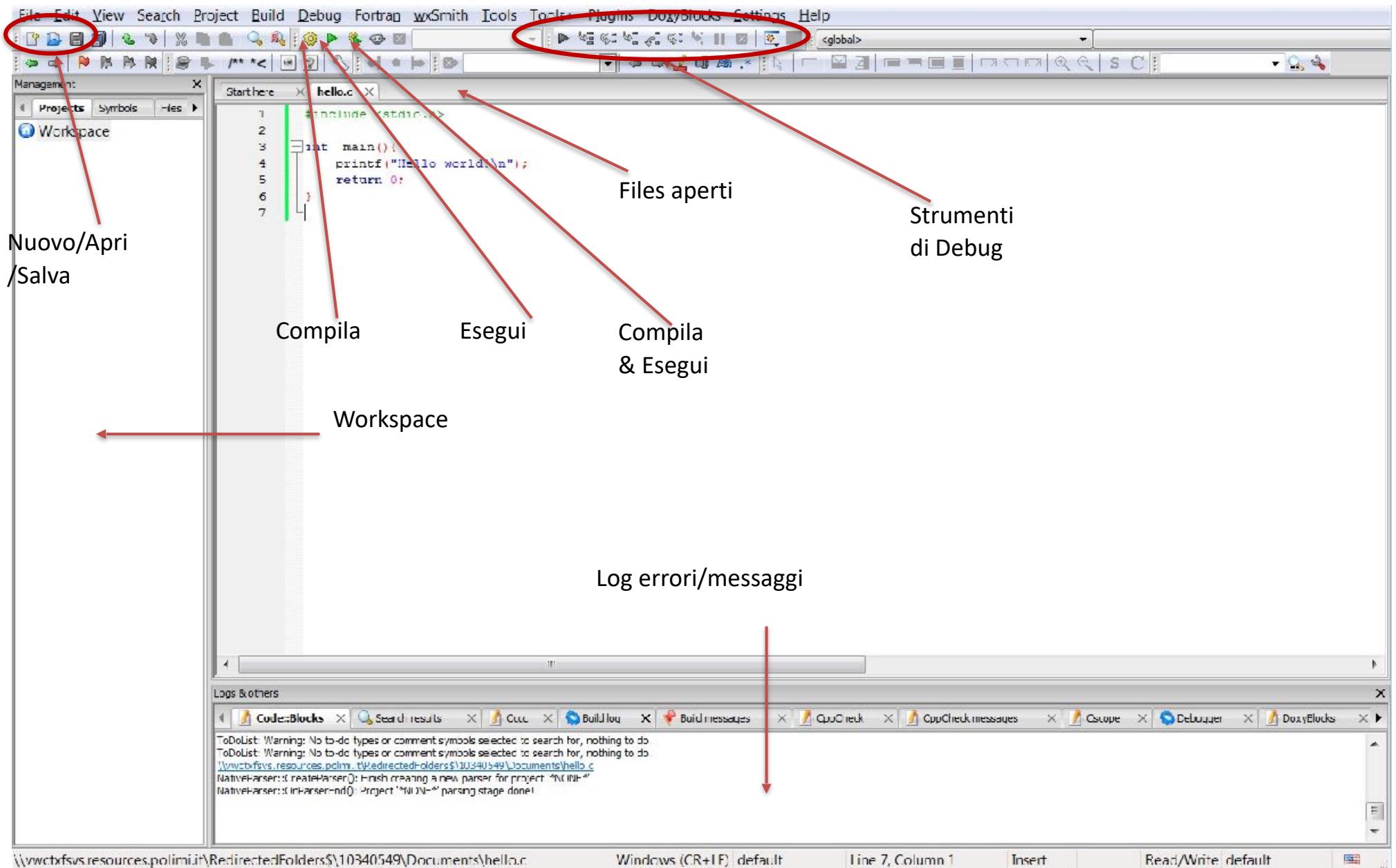
The screenshot shows the CLion IDE interface. The top bar includes icons for switching between windows, a "untitled" tab, "Version control", "Debug" mode, and other toolbars. The left sidebar has sections for "Project", "untitled ~/CLionProjects/L", "External Libraries", and "Scratches and Consoles". The main area displays a code editor with three tabs: "CMakeLists.txt", "library.h", and "library.c". The "library.c" tab is active, showing the following C code:

```
1 #include "library.h"
2
3 #include <stdio.h>
4
5 void hello(void) {
6     printf("Hello, World!\n");
7 }
8
```

A black arrow points from the text "Identico x Mac e win" below to the CLion interface.

Identico x Mac e win

# Ambiente di sviluppo CodeBlocks



# Ambiente di sviluppo CLion

Vedi anche:

attivazioneCLION 2024.pages.pdf

installazione\_CLION\_vers1.1+attivazione.key.pdf

Da qui:

<https://webeep.polimi.it/mod/folder/view.php?id=312964>

# Introduzione al linguaggio C

## Struttura di un programma C

- **Librerie:** Includono funzionalità comuni a molti programmi e si possono dunque importare in maniera standard.
- **main():** è la funzione di ingresso per ogni programma C.
- Deve esserci obbligatoriamente. Il codice al suo interno è il punto di partenza per il programma.
- **Commenti:** le linee precedute da “//” non sono eseguite. È molto importante usare commenti per non perdere traccia di quello che si sta facendo.

```
#include <stdio.h>
#include <stdlib.h>

void main()
{
    //DO SOMETHING
}
```

# Commenti

Altri modi per commentare codice

---

```
printf("hi"); //This is an in-line comment
/*
This is a multi-line comment.
All this text won't be executed.
This goes on until you write:
*/
```

# printf

La funzione **printf** ci permette di stampare a schermo e comunicare con l'utente.

```
printf("hello world!");
```

## Chiamare una funzione

Per chiamare una funzione, la sintassi è:

```
nome_funzione(parametro_1, parametro_2, parametro_n);
```



In C, le istruzioni terminano con un punto e virgola. Dimenticarselo è un errore molto comune, quindi se qualcosa non va, è tra le prime cose da guardare.

# Esercizio 0

Scrivere un programma che stampa a schermo le seguenti informazioni:

**data di oggi**  
**nome, cognome,**  
**matricola**

```
#include <stdio.h>
#include <stdlib.h>
```

```
void main()
{
```

?

}



Per andare a capo, si scrive: \n



Il backslash \ è un carattere di *escape*, ed è usato per indicare i caratteri speciali come \n (new line).

Per poter scrivere \, si può scrivere \\.

Un altro esempio di carattere speciale è il tab, che si scrive con \t.

# Compilazione

## Compilare

In assenza di errori, viene prodotto un file.exe nella stessa cartella in cui è stato salvato il codice sorgente.

**In presenza di errori**, il codice non viene compilato e vengono mostrati dei messaggi relativi agli errori rilevati.

## Correggere gli errori

I messaggi di errore sono importanti!

Correggere sempre gli errori in ordine, poiché alcuni errori potrebbero dipendere da errori precedenti.

# Errori comuni

Errori di compilazione

Controllare di aver salvato sempre i file in formato `.c` e non `.c++`.

Controllare di aver salvato nella cartella remota e non nel disco locale.

Controllare di aver chiuso altre istanze del vostro programma.

# Esercizio 0 - Soluzione

```
void main()
{
    printf("xx/xx/xxxx\n");
    printf("Loris, Giulivi, 874611");
}
```

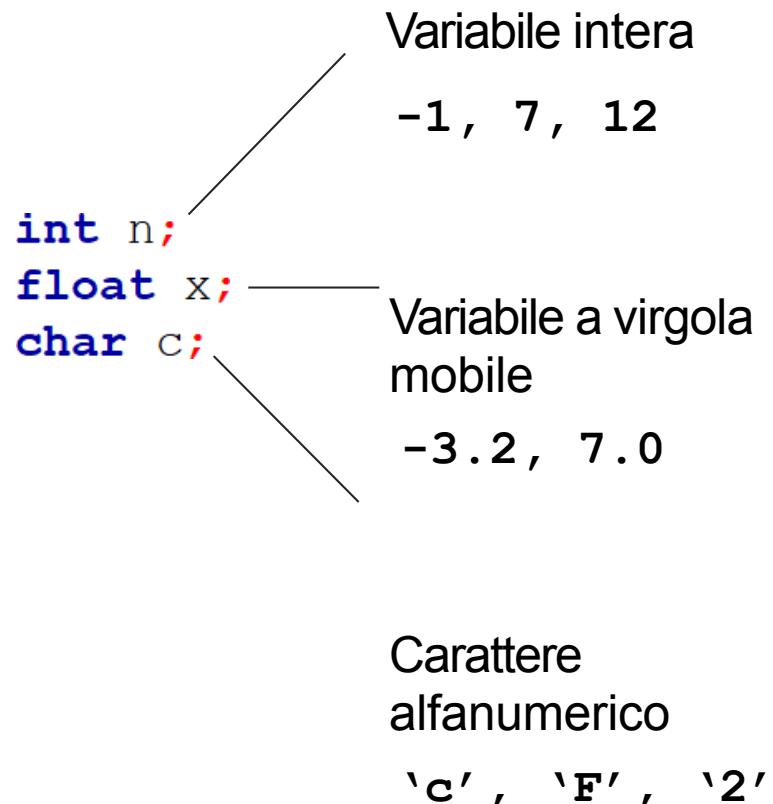
# Variabili

## Salvare un valore

Le variabili sono contenitori per i nostri dati. Al loro interno possiamo salvare informazione di vario tipo.

## Tipi di variabile

In C, le variabili sono *tipizzate*. Ciò significa che il tipo di informazione contenuto nella variabile deve essere specificato durante la definizione della variabile.



# Visibilità delle variabili

## Scope

Il nome delle variabili le identifica all'interno del *blocco di codice* in cui sono state definite.

Ciò significa che possiamo avere più variabili diverse con lo stesso nome, se ridefinite in diversi blocchi.

Lo *scope* più ampio è quello delle variabili globali.

```
int i=3;

void main()
{
    int i=2;
    //IL CODICE QUI VEDE LA i=2
}
```

```
int VARIABILE_GLOBALE_INTEGRA;

void main()
{
    char VARIABILE_LOCALE_CARATTERE;
}
```

# Stampare variabili con printf (1)

## La stringa di formato

La funzione `printf` ci permette anche di stampare i valori presenti nelle variabili.

Per farlo, si scrive una *format string*, che contiene delle sequenze speciali che indicano quali valori prendere e dove mostrarli:

`%d` numero intero

`%c` carattere alfanumerico

`%f` numero decimale

```
int i=3;  
  
void main()  
{  
    printf("The value is: %d", i);  
}
```

La sequenza speciale `%d` indica che qui dovrà essere stampato un intero

Il nome della variabile che vogliamo stampare è poi inserito come parametro nella chiamata a `printf`.

# Stampare variabili con printf (2)

## La stringa di formato

È anche possibile stampare più di un valore nella stessa chiamata a `printf`. In questo caso, aggiungiamo un parametro per ogni valore che vogliamo stampare.

```
int i = 3;  
float f = 2.4;
```

```
printf("The integer is: %d, the float is: %f", i, f);
```



Le sequenze speciali indicano alla `printf` come interpretare i dati in memoria, è quindi importante che siano coerenti con il tipo effettivo delle variabili da stampare.

Cosa succede se chiedo alla `printf` di stampare un `char` utilizzando `%d`?

PROVATE VOI!!

# Esercizio 1

Scrivere un programma che,  
date due variabili *char*,  
stampa una parola  
composta dalle due lettere,  
ripetute due volte.

char\_1 = 'a' char\_2 = 'b'  
→ stampa: 'abab'

```
void main()
{
    char char_1 = 'a'
    char char_2 = 'b'
```



}

# Esercizio 1 - Soluzione

```
#include <stdio.h>
#include <stdlib.h>

void main()
{
    char char_1 = 'a';
    char char_2 = 'b';

    printf("%c%c%c%c", char_1, char_2, char_1, char_2);
}
```

# Operatori aritmetici

Espressioni aritmetiche nel codice

Le normali operazioni aritmetiche usano gli operatori classici:

+ - \* /

Modulo

L'operatore `%` ritorna il resto divisione intera tra gli operandi.

```
int a = 1;  
int b = 2;  
int c = a + b + 3;
```

`12%5 = 2`

Operatori di assegnamento

Operatori speciali che eseguono una operazione aritmetica e assegnano il risultato al primo dei due operandi.

```
a = 2;  
b = 5;
```

`a += b;`

`// a = 7`

`+= -=  
*= /=`



Syntactic sugar

<code>a += 1;</code>	$\leftrightarrow$	<code>a++;</code>
<code>a -= 1;</code>		<code>a--;</code>

# Operazioni su interi

## Implicit casting

Se il tipo degli operandi e del risultato non sono compatibili, questi vengono modificati, se possibile, per permettere l'operazione.

```
float a = 2.7;  
int b = 1;  
  
int c = a + b;  
  
// c = 3
```

- ⚠ La conversione da `float` a `int` viene sempre effettuata troncando la parte decimale, e non per arrotondamento all'intero più vicino.

# Ricevere input dall'utente

Ricevere input dall'utente -

**scanf**

La funzione **scanf** ci permette di ricevere dati dall'utente e salvarli nelle variabili.

La stringa di formato

Come per la **printf**, una stringa di formato viene usata per determinare come ricevere i dati.

Il puntatore alla variabile

L'operatore unario **&** ritorna l'indirizzo di memoria della variabile operando.

Per indicare alla funzione dove salvare l'input dell'utente, durante la chiamata alla **scanf**,

```
int a;  
scanf ("%d", &a);
```

**&a** indica che vogliamo salvare il valore nella cella di memoria dove è situata la variabile **a**.

**i** La **scanf** può salvare anche più di un valore alla volta, ma tipicamente questo non viene fatto, perché richiede che l'utente rispetti il formato specificato nella stringa di formato.

```
int a;  
float b;  
  
scanf ("%d-%f", &a, &b);
```

Nell'esempio, l'utente dovrebbe inserire una stringa del tipo:

## Esercizio 2

Scrivere un programma che chiede all'utente due numeri interi e che ne stampi la somma.

# Esercizio 2 - Soluzione

```
#include <stdio.h>
#include <stdlib.h>

void main()
{
    int num1;
    int num2;

    printf("Insert the first integer: ");
    scanf("%d", &num1);
    printf("Insert the second integer: ");
    scanf("%d", &num2);

    int risultato = num1 + num2;
    printf("The sum is: %d", risultato);

}
```

**i** Si può anche inserire l'espressione come parametro della **printf**

```
printf("The sum is: %d", num1 + num2);
```

## Esercizio 3

Scrivere un programma che chiede in input all'utente la prima lettera del proprio nome e la prima lettera del proprio cognome, e che le restituisce nell'ordine inverso (prima cognome poi nome).

**fflush(stdin) ;**

# Esercizio 3 - Soluzione

```
void main()
{
    char name;
    char sur;

    printf("Please, enter the first letter of your name: ");
    scanf(" %c", &name);
    printf("Please, enter the first letter of your surname: ");
    scanf(" %c", &sur);

    printf("%c%c \n", sur, name);
}
```

# Esercizi aggiuntivi

## Esercizio 4

Scrivere un programma che chiede in input all'utente due numeri interi e che stampa risultato e resto della divisione tra i due numeri.

## Esercizio 5

Scrivere un programma che dato un numero rappresentante un prezzo  $p$  ed un valore  $s$  tra 0 e 100, ritorni il prezzo scontato del  $s\%$ .

## Esercizio 6

Scrivere un programma che, acquisiti tre voti universitari e relativi valori di CFU, restituisca la media pesata.

# Esercizio 7

Scrivere un programma che chiede in input all'utente un numero  $r$  e che restituisca:

1. Il perimetro della circonferenza di raggio  $r$
2. L'area del cerchio di raggio  $r$
3. La superficie della sfera di raggio  $r$
4. Il volume della sfera di raggio  $r$

```
#include <math.h>  
  
pow(a, b); //a^b
```

Wikipedia formule:

<https://it.wikipedia.org/wiki/Sfera>

## Esercizio 8

Scrivere un programma che acquisiti tre valori rappresentanti i coefficienti di una equazione di secondo grado, ritorni le radici reali dell'equazione.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

# Installing tools on win11

MacOS  
A.A. 2023/24



**POLITECNICO**  
MILANO 1863

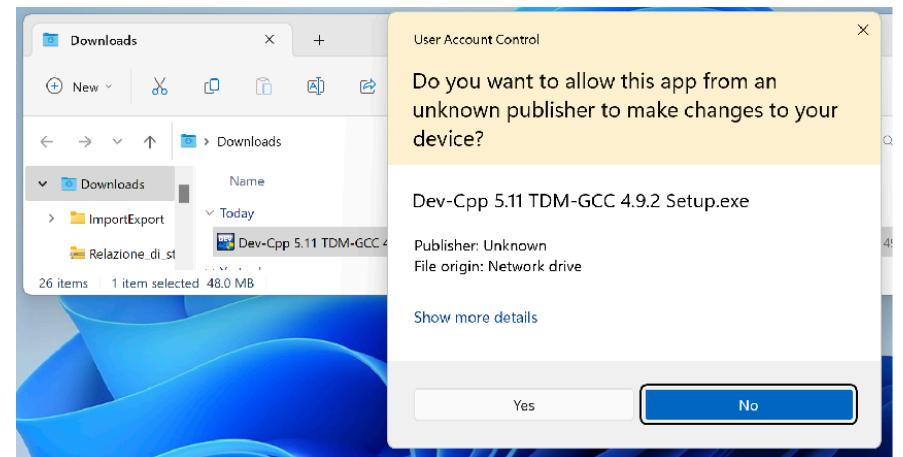
# Dev c++

From:

<https://sourceforge.net/projects/orwelldevcpp/>

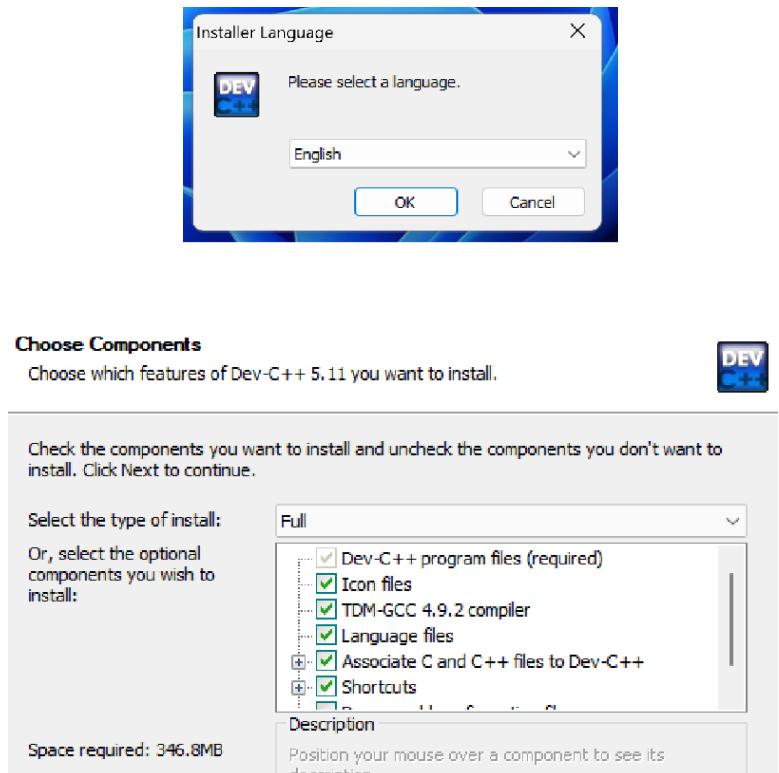


In download folder, open installer,  
"Dev-Cpp 5.11 TDM-GCC 4.9.2 Setup.exe":



# Dev c++

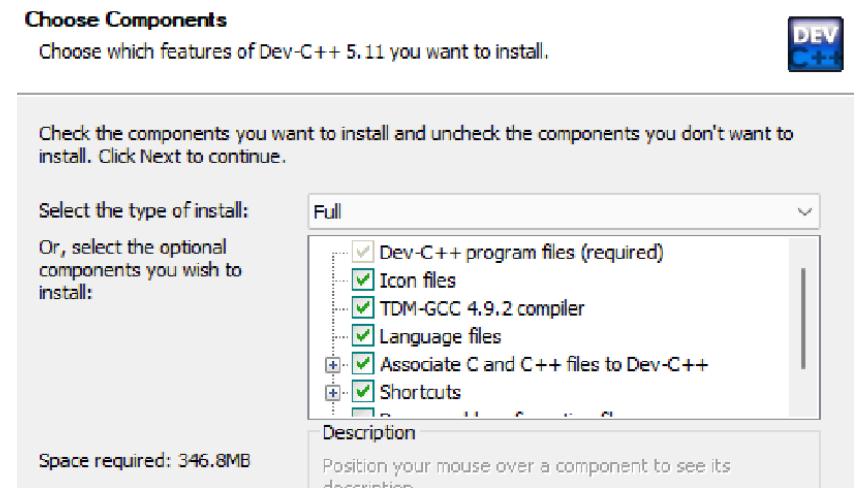
Accept conditions..



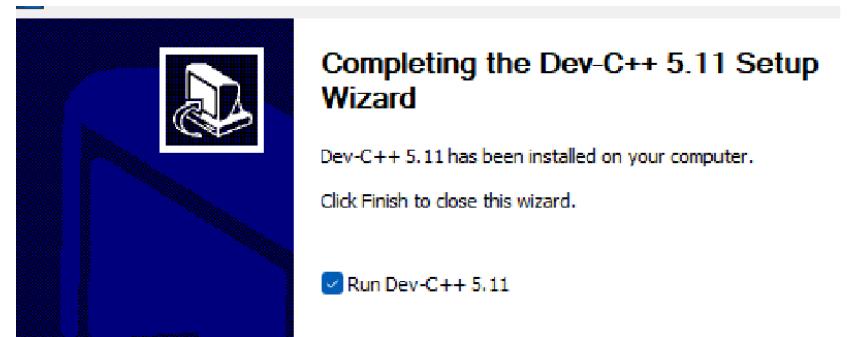
# Dev c++

Default preset Option are fine, Next...

The same for folder, go on.

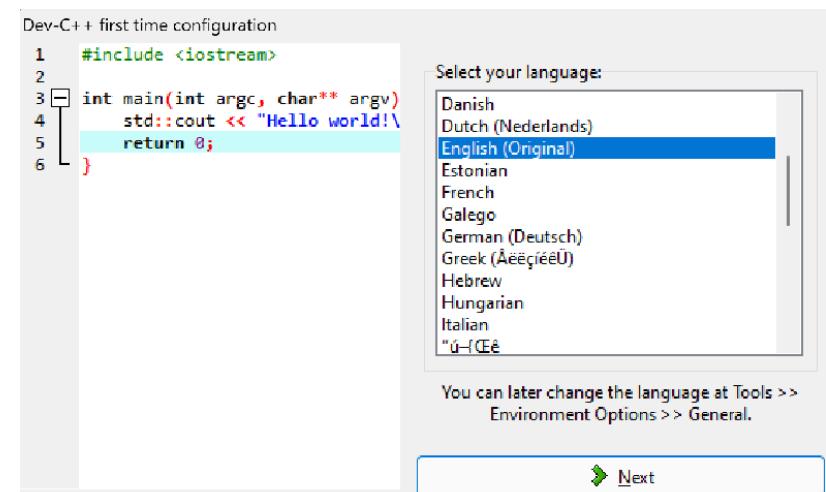


After a while,



# Dev c++

Let's try if works:



**Next:**

(English is better as you will see the same text / messages / alert as you can find on internet / stack Overflow)

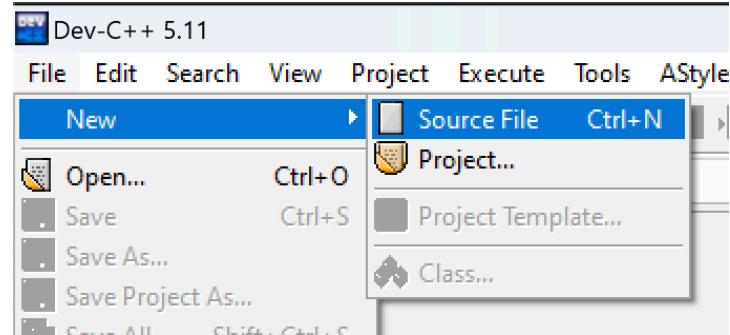
Other dialogs can appear, simply accept (or customise if you know what to choose).

Normally you will have an icon on desktop to run it.

# Dev c++

## Open and RUN

When you open, you can see:



For simpler programs, normally choose "Source File"

You can also choose "Project" if Your program is composed of multiple C file.

Now in editor You can write code.

For example:

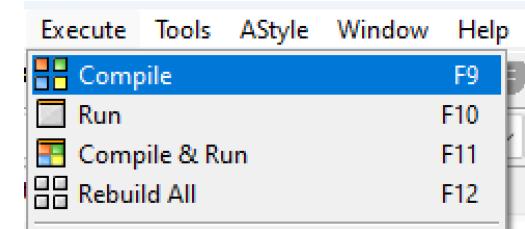
```
#include <stdio.h>

int main() {
    printf("Hello, World!");
    return 0;
}
```

After writing / pasting, you can use menus or shortcuts:

# Dev c++

After writing / pasting, you can use menus or shortcuts:

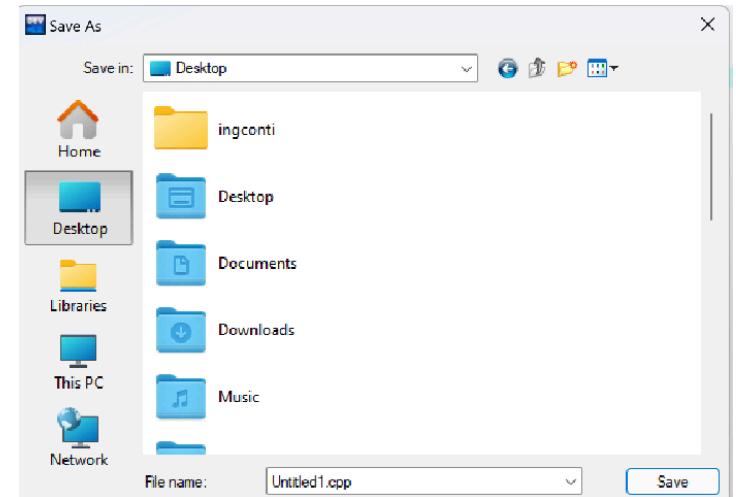


F10 is the more common.

# Dev c++

For newly created files,

You have to save before running:



Pls use a meaningful name, and REMEMBER folder, so in Labs is easier for teachers to help You.

If everything is ok, You should see:

A screenshot of a terminal window titled '\Mac\Home\Desktop\Untitled'. The window displays the following text:

```
Hello, World!
-----
Process exited after 0.5722 seconds with return value 0
'\\Mac\\Home\\Desktop'
CMD.EXE was started with the above path as the current directory.
UNC paths are not supported. Defaulting to Windows directory.
Press any key to continue . . .
```

# Code Blocks (Windows)

<https://www.codeblocks.org/downloads/binaries/>

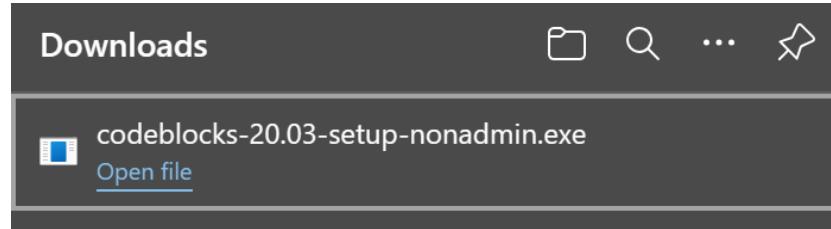
choose:

**codeblocks-20.03-setup-nonadmin.exe**

Will open:

<https://sourceforge.net/projects/codeblocks/files/Binaries/20.03/Windows/codeblocks-20.03-setup-nonadmin.exe/download>

In download You should have:



# Code Blocks (Windows)

Open file...

## Welcome to CodeBlocks Setup

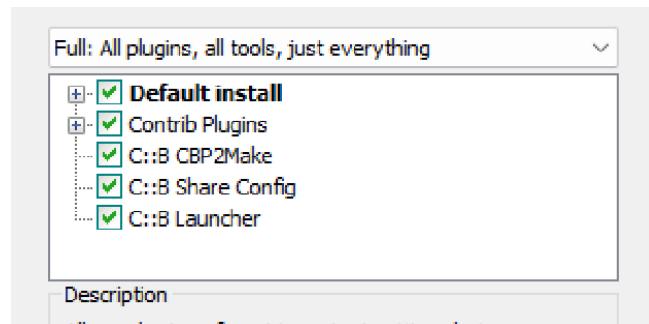
Setup will guide you through the installation of CodeBlocks.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

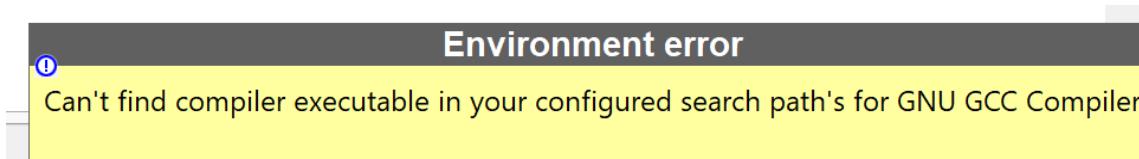
Go on.. "Next".. "Agree"

In options, leave all set:

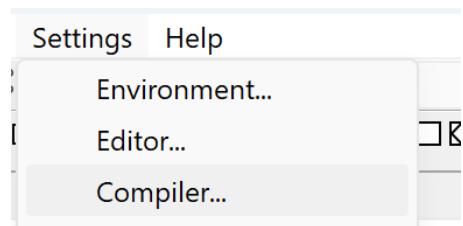


# Code Blocks (Windows)

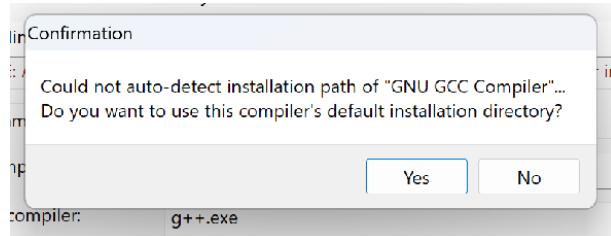
When open, can appear a warning about missing compiler:



Please check under settings:



You can see it is missing.

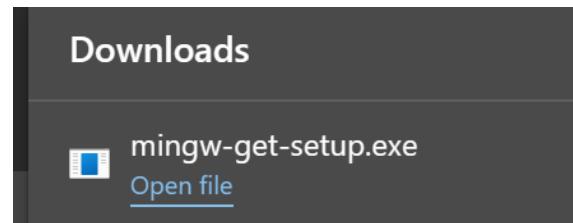


# Code Blocks (Windows)

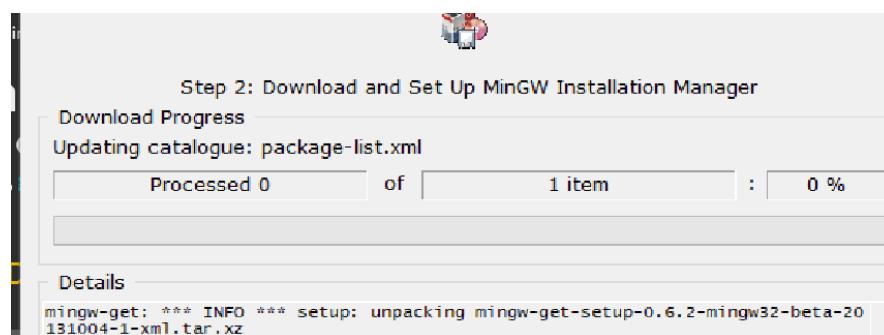
Install MIN GW:

<https://sourceforge.net/projects/mingw/files/latest/download>

Run installer:

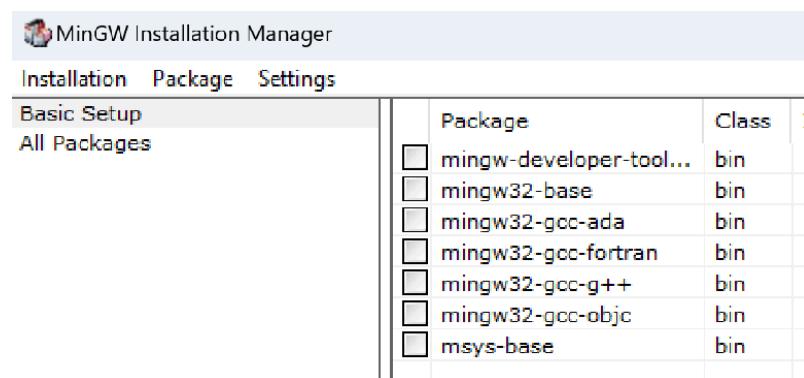


For subsequent view, accept default and go on, download of packets will start:



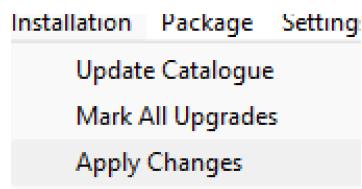
# Code Blocks (Windows)

After installing:



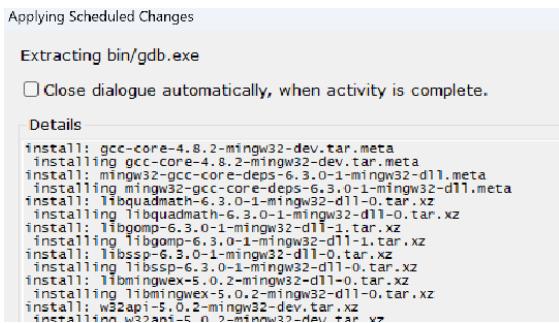
Here flag to install 1' and it will check automatically mngw-gcc.

In Menus, choose Apply:



# Code Blocks (Windows)

Another download will start.



You should have these setting. Note now You have the version column with values.

Package	Class	Installed Version	F
mingw32-base	bin	2013072200	2
mingw32-binutils	bin	2.28-1	2
mingw32-binutils	dev		2
mingw32-binutils	doc		2
mingw32-binutils	info		2
mingw32-binutils	lang		2
mingw32-binutils	man		2
mingw32-gcc	bin	6.3.0-1	6
mingw32-gcc	dev	4.8.2	4
mingw32-gcc	doc		4
mingw32-gcc	info		6
mingw32-gcc	lang		6
mingw32-gcc	lic	6.3.0-1	6
mingw32-gcc	man		6
mingw32-gcc-ada	bin		6
mingw32-gcc-ada	dev		4

# Code Blocks (Windows)

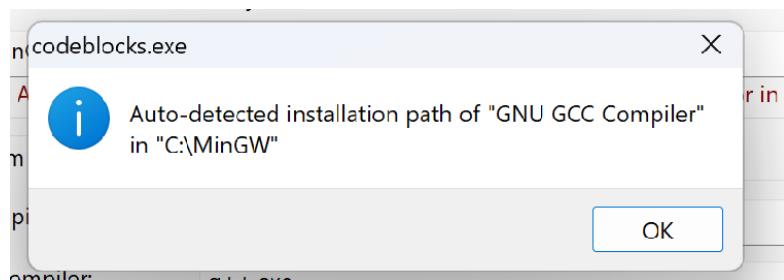
Now again codeBlocks

Go under "settings" you will see:



Click on Autodetect.

You should see:

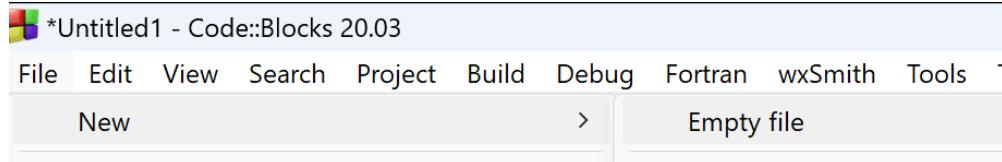


If NO, you made errors in selecting / installing MinGW.

# Code Blocks (Windows)

## Test Run

Under File, New:



Copy/Paste a sample "hello world":

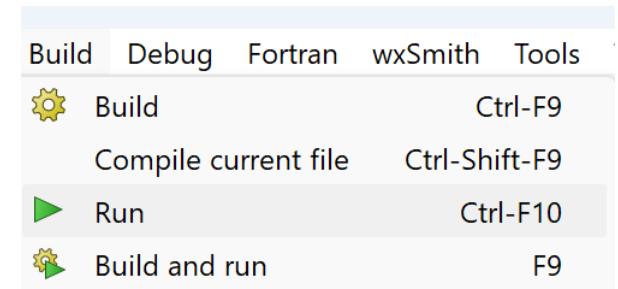
```
#include <stdio.h>

int main() {
    printf("Hello, World!");
    return 0;
}
```

Click on build and run (F9) You will do the same in Labs:

# Code Blocks (Windows)

Click on build and run (F9) You will do the same in Labs:



IDE will ask to save your c file, save as preferred, After a While You will have:

A screenshot of a terminal window titled '\\Mac\\Home\\Documents\\hell'. The window displays the following text:  
Hello, World!  
Process returned 0 (0x0) exec  
Press any key to continue.  
|

So Far so GOOD!

# Installazione IDE



A cross-platform IDE for C and C++

CLION Windows/Linux  
A.A. 2023/24



**POLITECNICO**  
MILANO 1863

# Rationale

- 1) CLion e' multiplataforma
- 2) e' moderno, esiste la "community ed." (Free)
- 3) come Studenti POLI avrete quello full Gratis previa **registrazione**
- 3) e' un IDE, serve pero' la toolchain (compilatore, linker... librerie binarie e header del C)
- 4) su Windows la toolchain NON c'e', su MacOS va installata a parte, su Linux di solito gia presente

*Bonus feature:* programmazione cooperativa!



## **Log Story ... SHORT**

- 1) scaricare CLion (<https://www.jetbrains.com/clion/>)
- 2) Registrarsi (opz. .... Come studenti POLI gratis)
- 3) Lanciare installer... al 99 % fa tutto lui..
- 4) scaricare ed installare la toolchain Mingw (idem)  
<https://sourceforge.net/projects/mingw-w64/>
- 5) configurare CLion x usare la toolchain
- 6) fare progetto di prova (vedere la sezione in fondo..)  
A) da "nuovo Progetto"

**Long story....**

**Ecco tutti i passi..**

# Registrarvi (Opzionale)

The screenshot shows a web browser window for [tbrains.com/licenses](https://tbrains.com/licenses?json=0081341ED71DB89C0DDDA0E7DD5FFCCC6AD88E1DE306849FB2A3D9CA2922C5). The page title is "1 License". It displays a "JetBrains Product Pack for Students" license. The license details are as follows:

- Licensed to: **Gian Conti**
- License restriction: For educational use only
- Valid through: August 10, 2022
- Following products included:
  - AppCode
  - CLion
  - DataGrip
  - dotCover
  - dotMemory
  - dotTrace
  - GoLand
  - IntelliJ IDEA Ultimate
  - PhpStorm
  - PyCharm
  - ReSharper
  - ReSharper C++
  - Rider
  - RubyMine
  - WebStorm

The license ID is XQB8L6POO7.

# Download..

1) scaricare CLion

(<https://www.jetbrains.com/clion/>)

2) salvare

3) eseguire installer

## Download CLion

Windows

macOS

Linux

CLion includes an evaluation license key for

Download

.exe ▾

Windows (.exe)

Windows (.zip)



Get

pp to d

What do you want to do with CLion-2021.2.1.exe (549 MB)?  
From: download-cdn.jetbrains.com

Save

Save as

Cancel

X

CLion-2021.2.1.exe finished downloading.

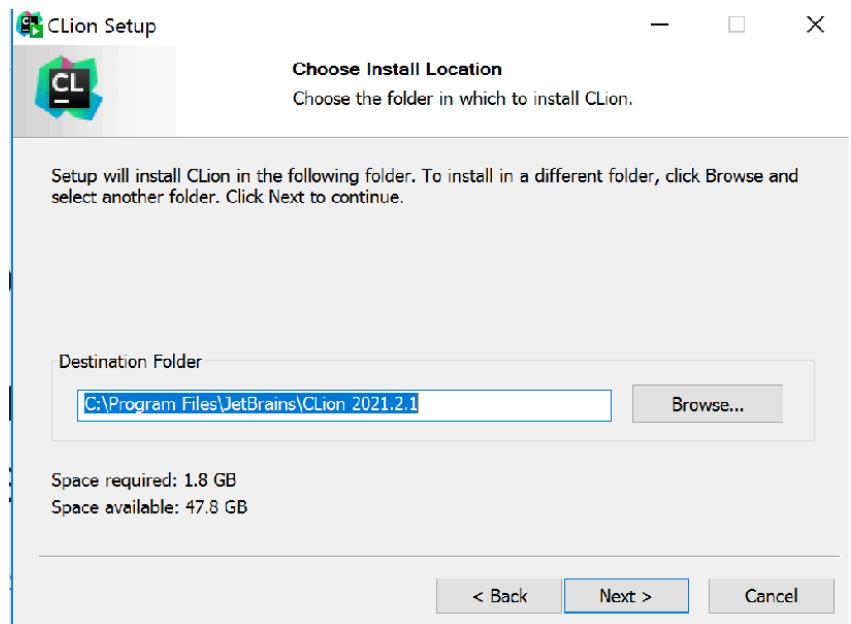
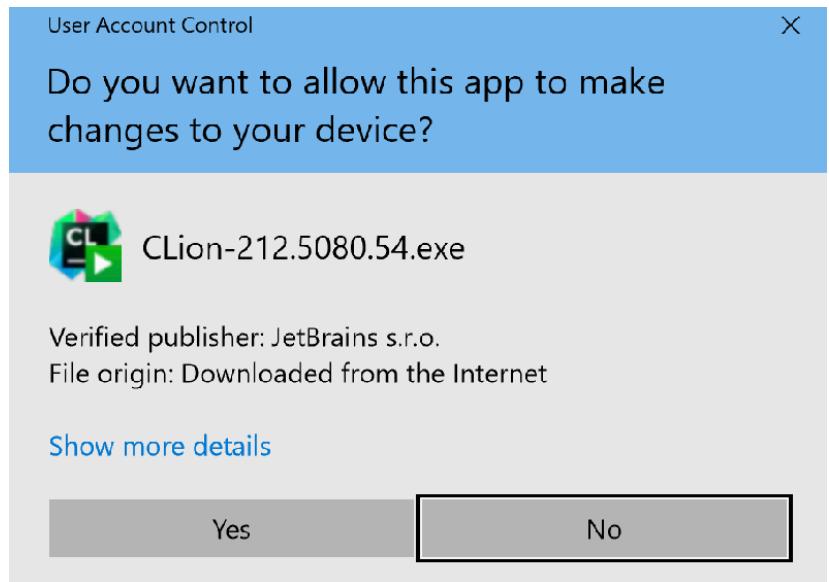
Run

# Installazione.. screen

Accettare...

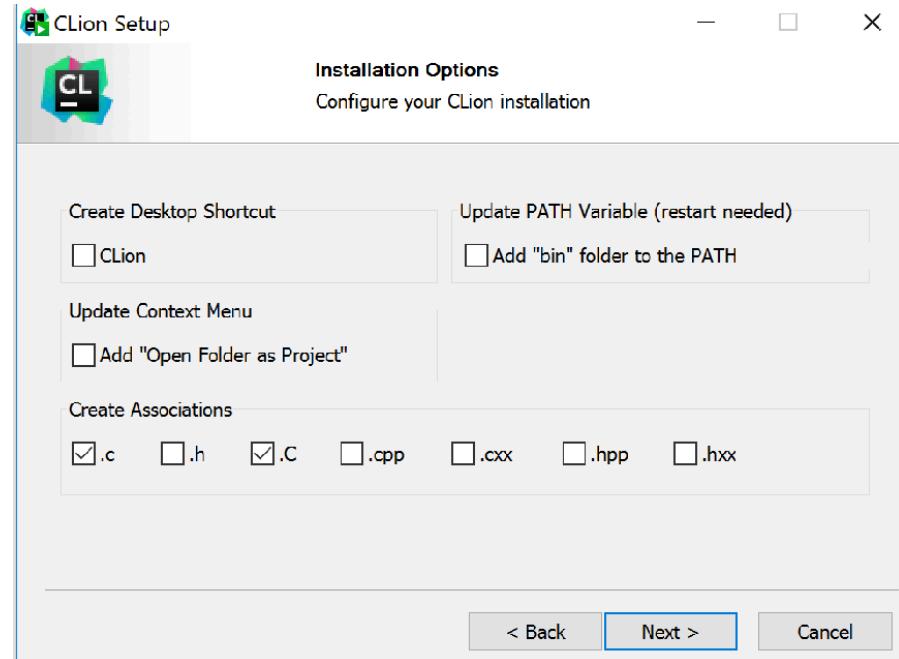
(*Eventualmente*

personalizzare..)



# Opzioni

Useremo CLion solo x C



Dopo install...

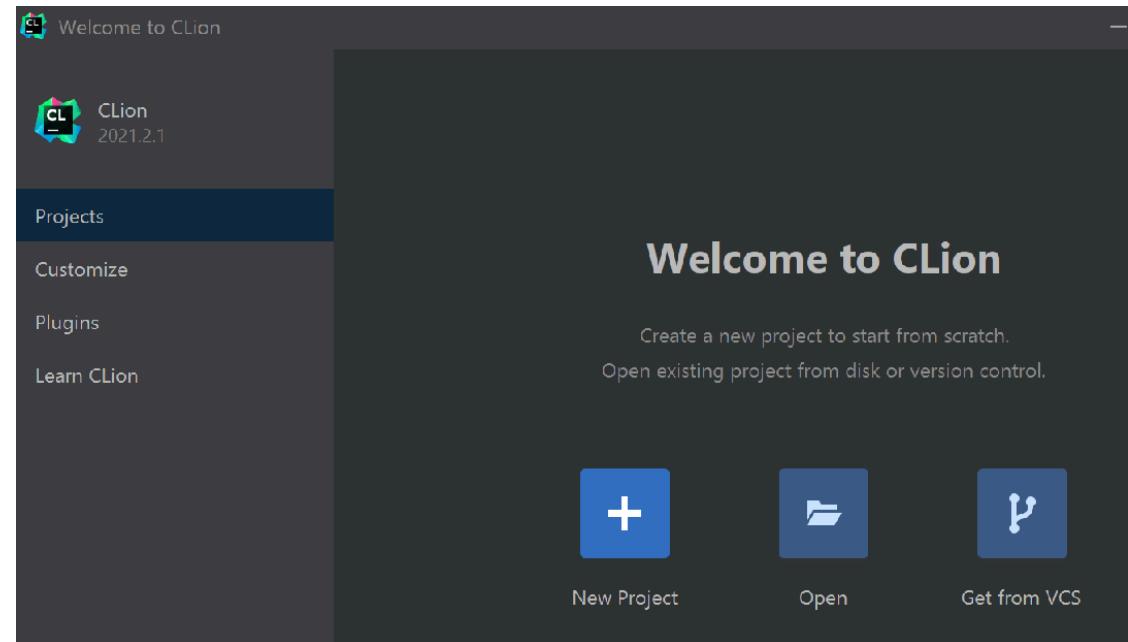
Credenziali

The screenshot shows the activation step of the setup wizard. It offers to 'Activate CLion' or 'Evaluate for free'. Under 'Get license from:', 'JB Account' is selected. The 'Username / email:' field contains '10047232@polimi.it' and the 'Password:' field is masked. A large blue 'Activate' button is at the bottom.

# Done..

CLion e' installato,

Ma mancano i tool...



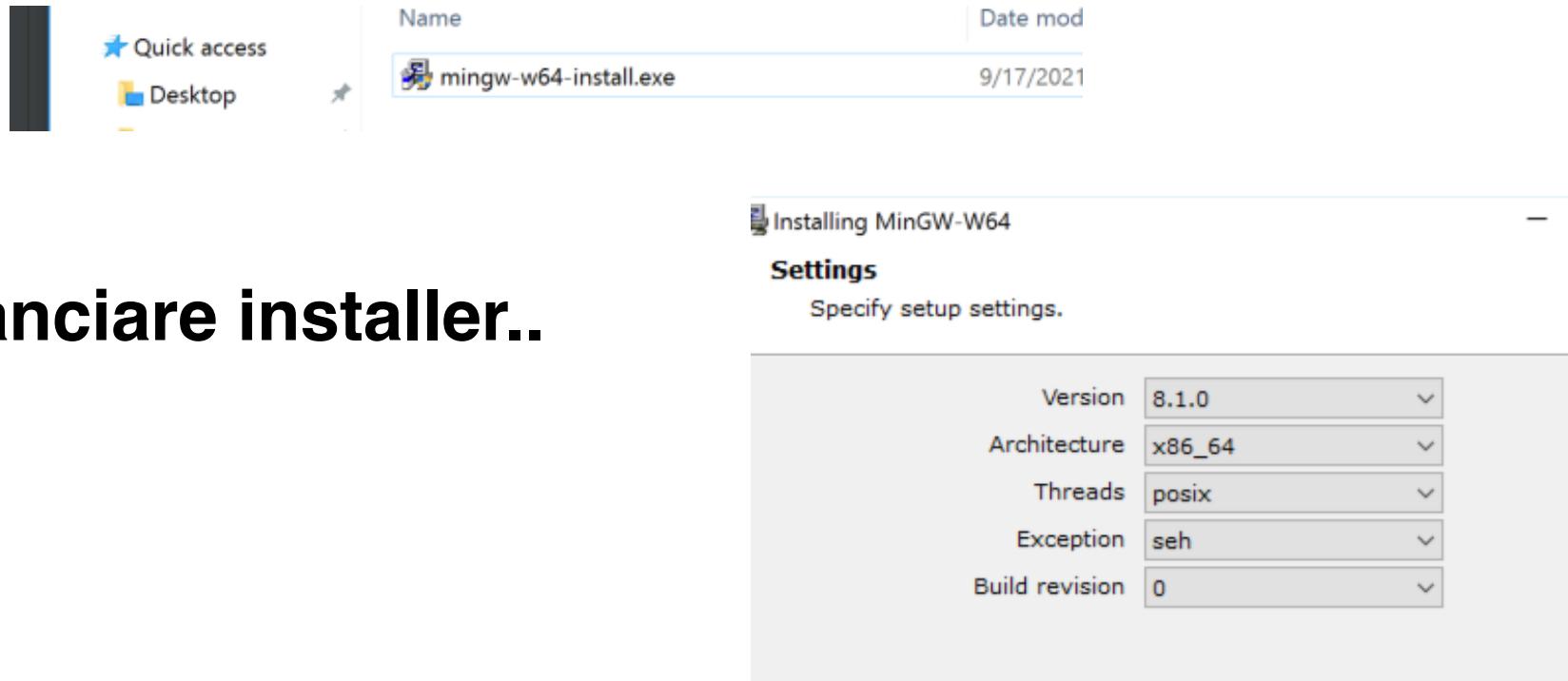
# Useremo Mingw

<https://www.mingw-w64.org/downloads/>

Installation: Sourceforge

<https://sourceforge.net/projects/mingw-w64/>

# Quando scaricato,



# lanciare installer..

# Riaprire CLion

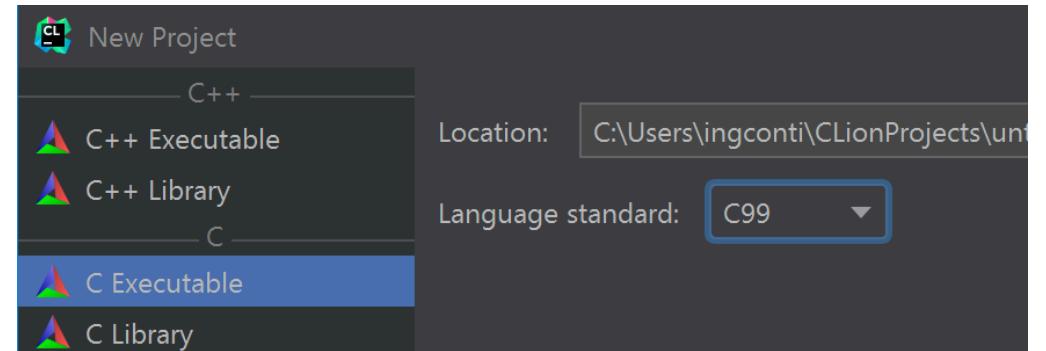
Due opzioni:

- A) partire da nuovo progetto...
- B) impostare ambiente

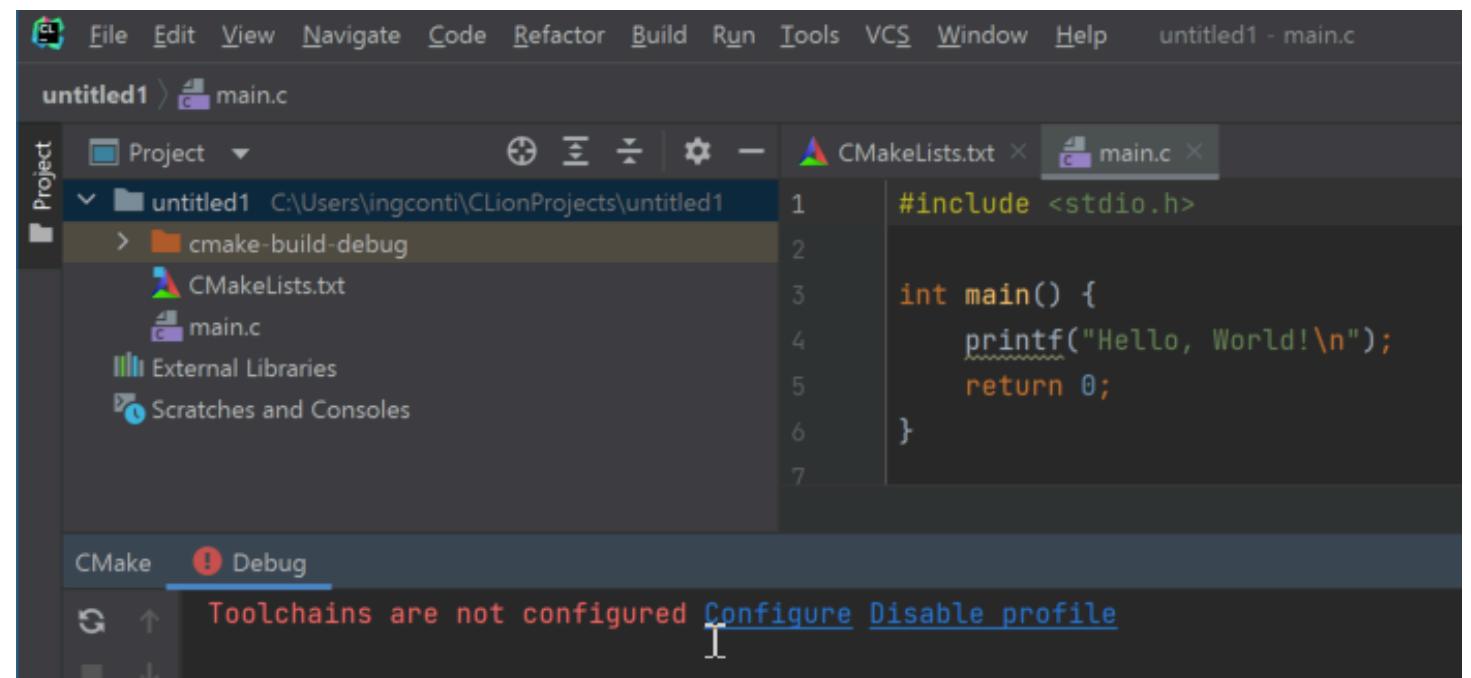
Ps. La impostazione del compilatore va fatta

***una sola volta***, usualmente.

# A) da "nuovo Progetto"



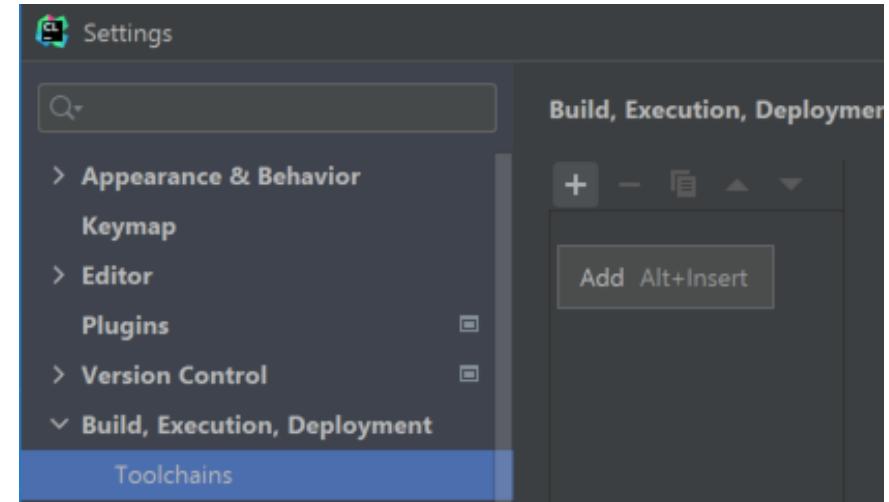
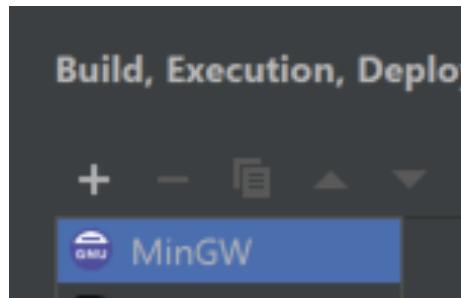
Va configurata la toolchain, ossia va detto dove sono compilatore e linker



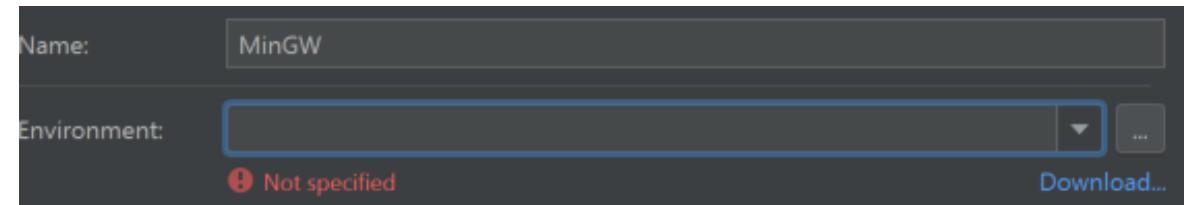
Click su "Configure"

Su "+"

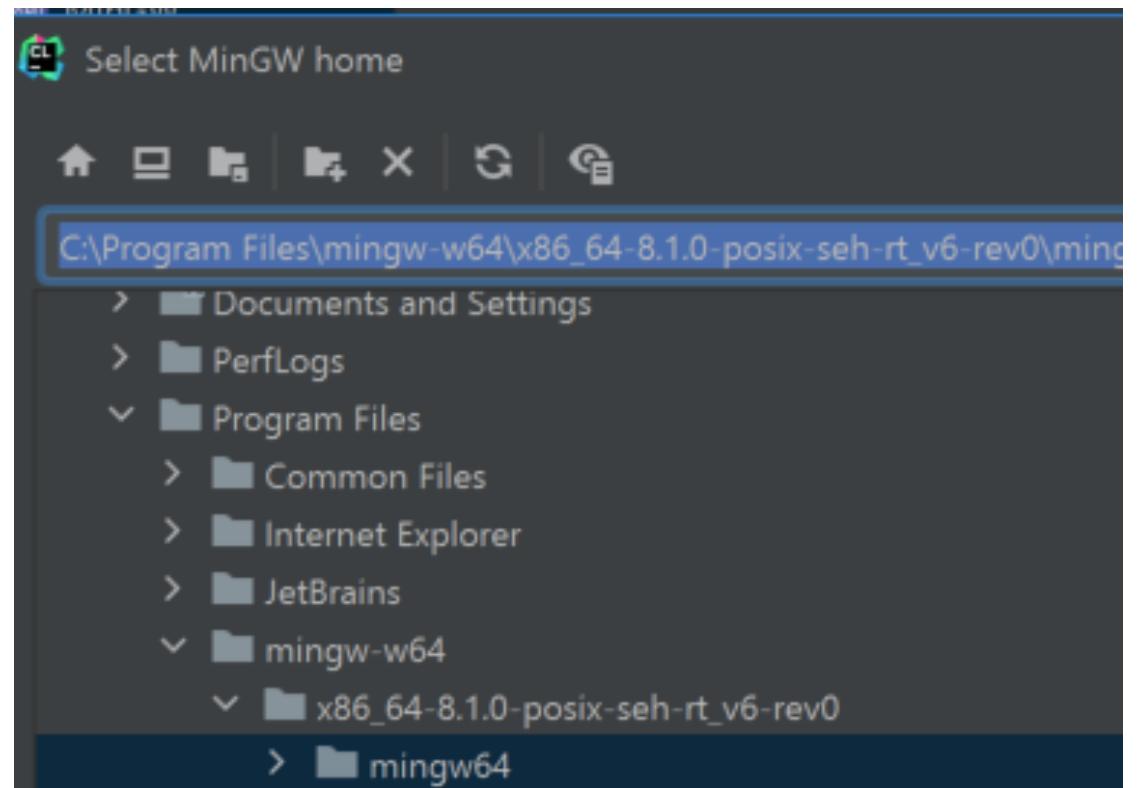
Aggiungere MinGW:



Selezionare cartella:

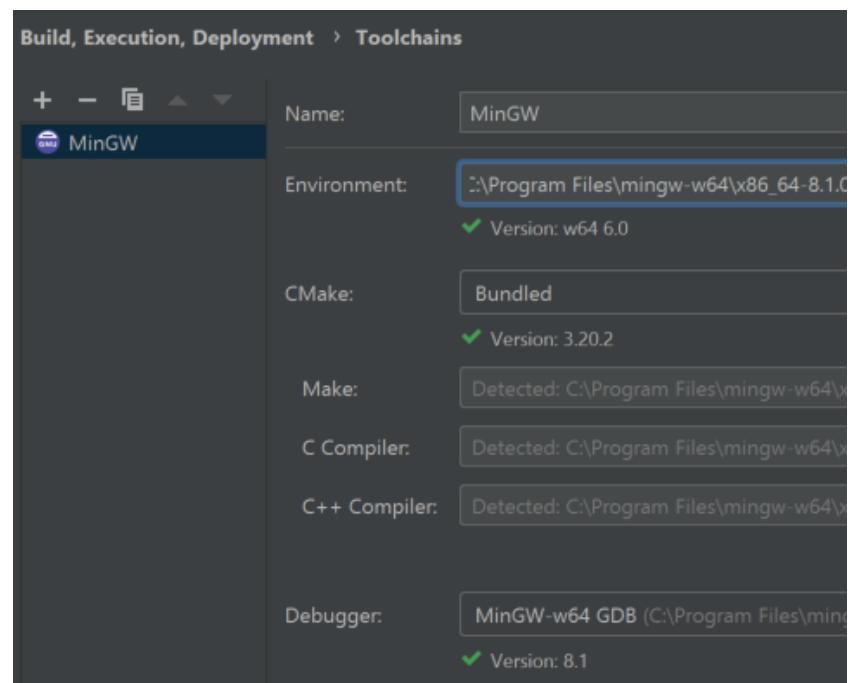


QUI:



Se tutto ok, apparira'

finestra con spunte VERDI:



# 1' progetto

Concetti sottesi:

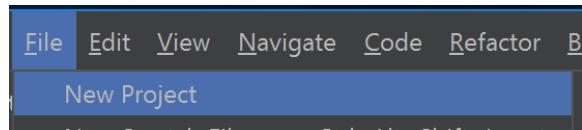


- useremo **SOLO** il C
- ogni attività ingegneristica ha un "progetto"
- creeremo un mini-project
- Clion serve x moltissimi altri tipi di progetti.... Non li vedremo!

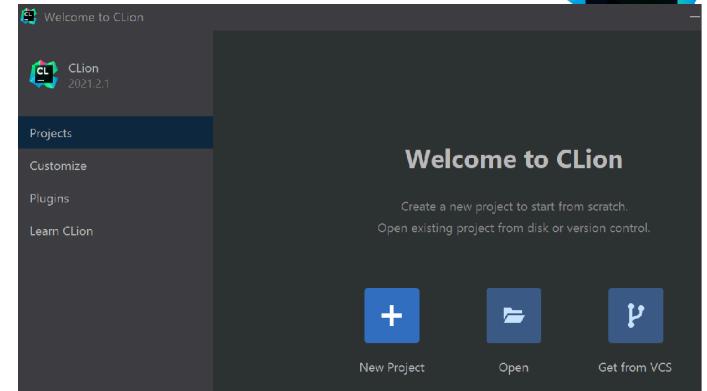


# 1' progetto (2)

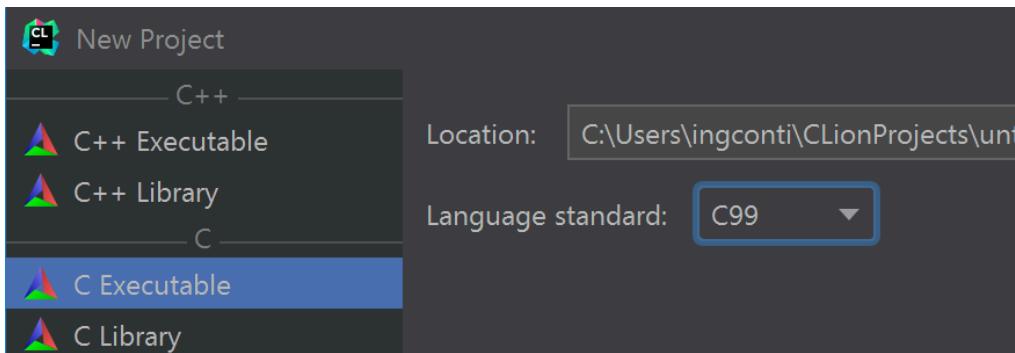
a) Nuovo..



O da:



b) progetto SOLO C, p.es con versione C99

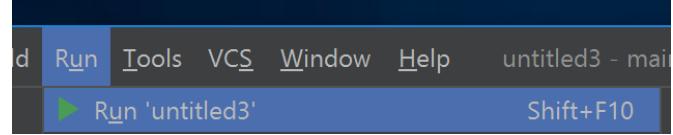


c) NO 3rd step! Avete già "HELLO WORLD"

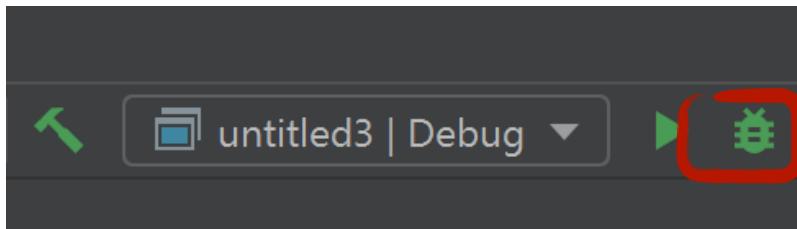
**RUN!**

# 1' progetto (3)

a) RUN (non è "play 😊 )



b) oppure debug



...



# 1' progetto (4)

c) apparira'

The screenshot shows the CLion IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help, and untitled3 - main.c. The title bar shows 'untitled3 > main.c'. The left sidebar has a 'Project' view with a tree structure: 'untitled3' (C:\Users\ingconti\CLionProjects\untitled3), 'cmake-build-debug' (selected), 'CMakeLists.txt', 'main.c', 'External Libraries', and 'Scratches and Consoles'. The main editor area displays the 'main.c' file with the following code:

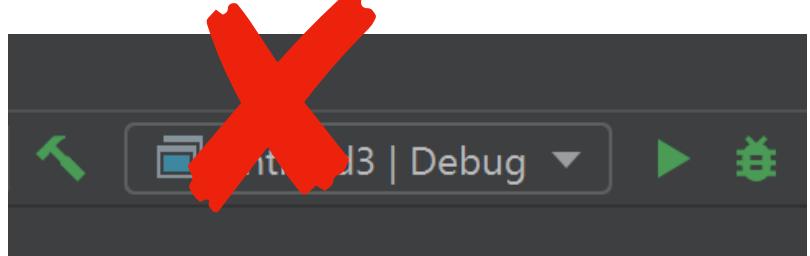
```
2
3 int main() {
4     printf(_Format: "Hello, World!\n")
5     return 0;
6 }
7
```

The bottom terminal window shows the output of the run command:

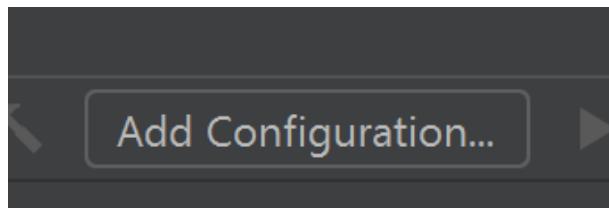
```
Run:  untitled3 ×
▶  C:\Users\ingconti\CLionProjects\untitled3\cmake-build-debug\untitled3.exe
Hello, World!
Process finished with exit code 0
```

# 1' progetto (5) tips and tricks.

1) Non appare



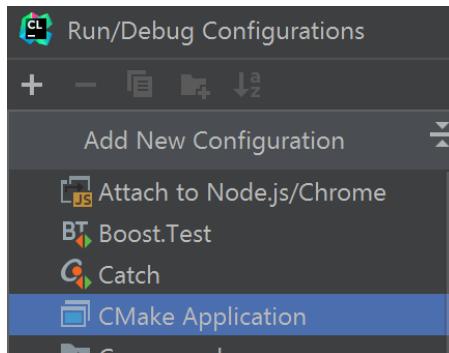
Ma invece:



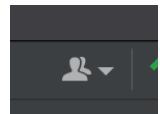
2) fate ADD.. con il "+"

Make

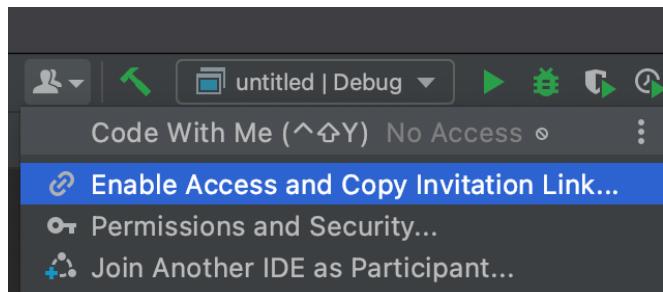
E date ok...



# "Code with Me"



Potete chattare, vedere codice, editare con i vs compagni lo stesso progetto senza altri tool!

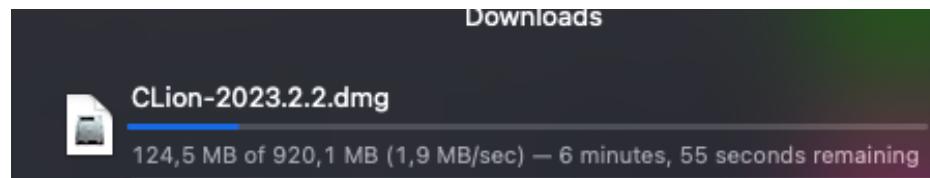




# " And... Mac?"

Easy.

- scaricate la vers. X Mac (safari vi porta già su quella corretta)

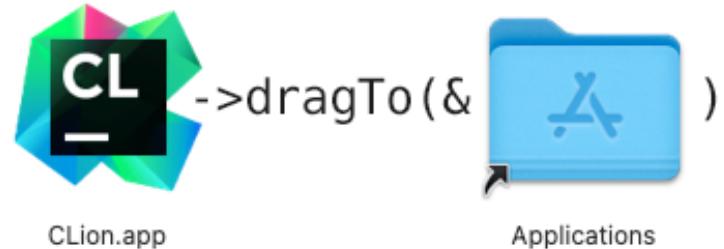




# " And... Mac?"



Aprite DMG...



La parte di registrazione **identica**

**3rd step?**

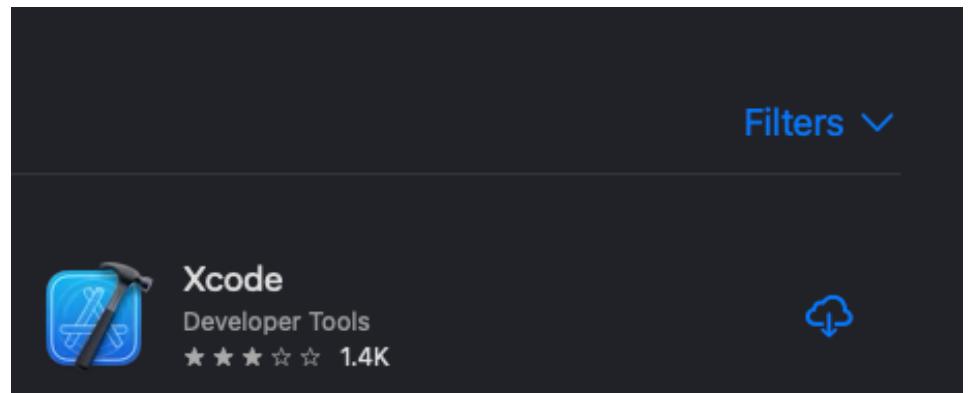


# " And... Mac?"

*No third step!*

Se avete già Xcode, CLion riconosce il compilatore di Apple

Se no.. App Store:



# Installazione IDE

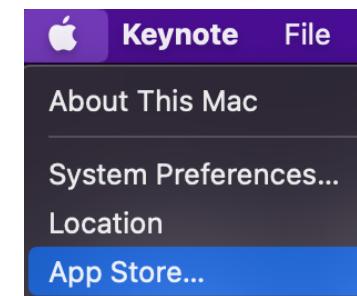
MacOS  
A.A. 2023/24



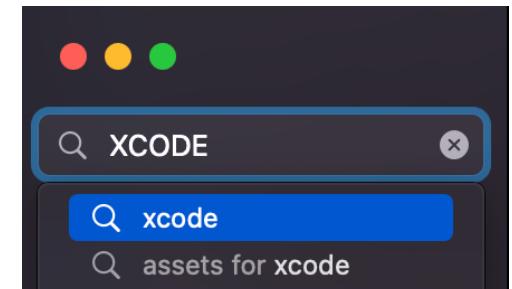
**POLITECNICO**  
MILANO 1863

Xcode e' una "normale" APP, quindi.:

1) andare su AppStore



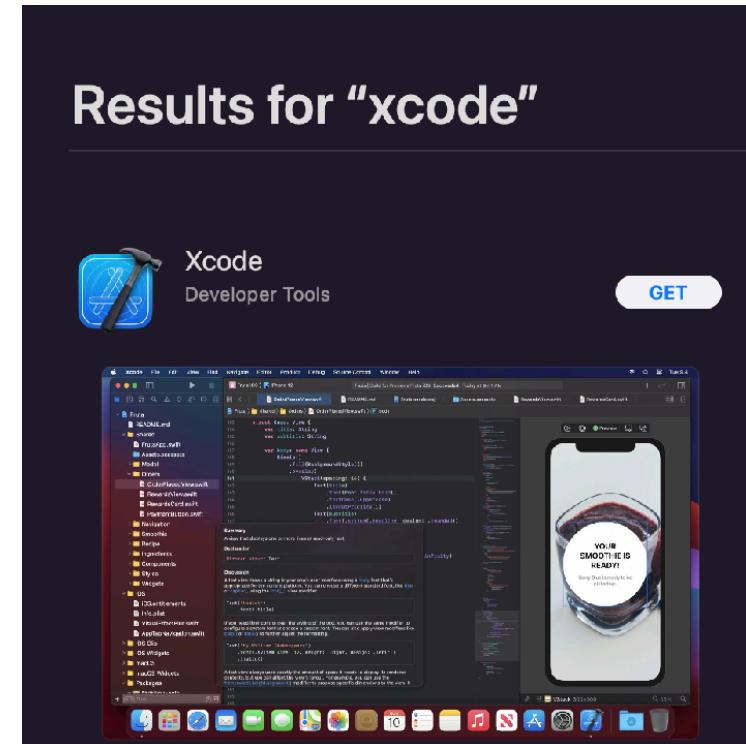
2) cercare...



3) scaricare Xcode..

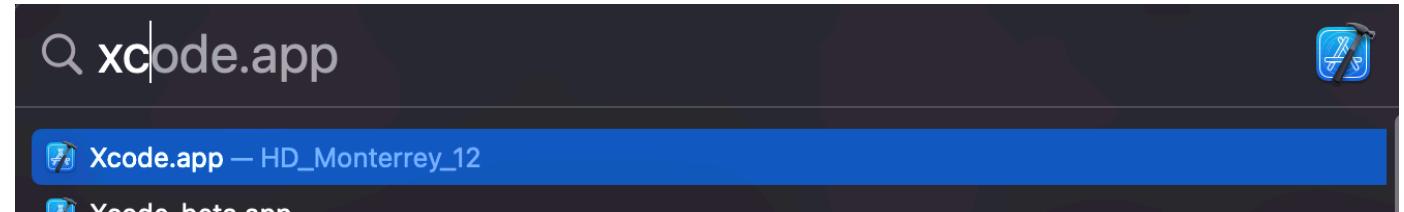
*Nota: richiedera' del tempo..*

*Sono molti GB*



A fine scarico andare nella App (cmd A)

(o solito cmd barra di Spotlight ...)



Aprire Xcode ...

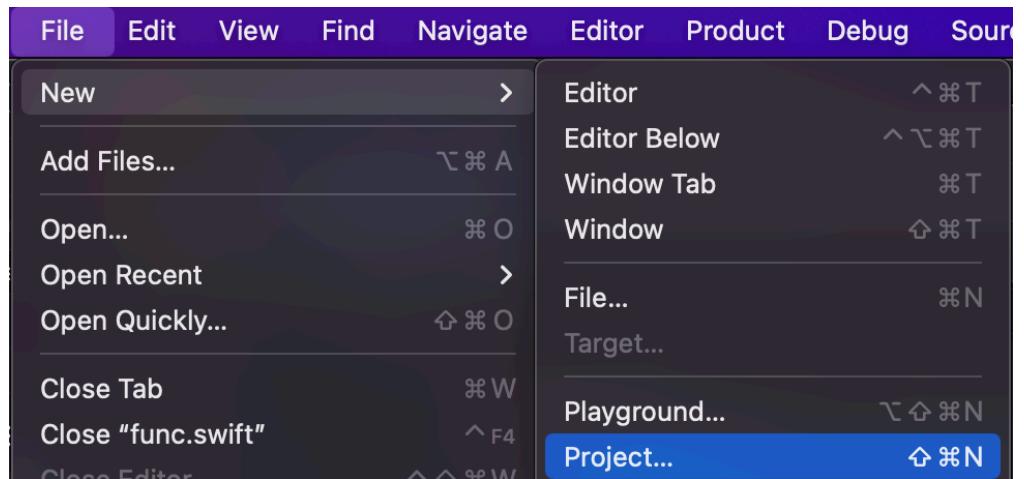
# 1' progetto

Concetti sottesi:

- useremo **SOLO** il C
- ogni attività ingegneristica ha un "progetto"
- creeremo un mini-project
- Xcode serve x moltissimi altri tipi di progetti.... Non li vedremo!

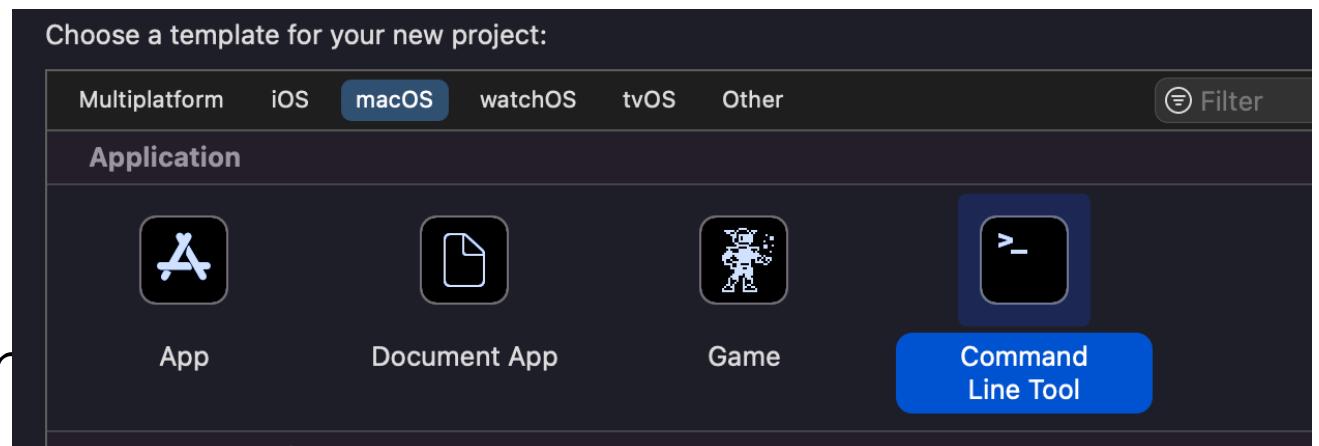
# 1' progetto: Hello World

1) da menu File,  
nuovo progetto..  
(cmd shift N)



2) appare:

Scegliere  
macOS E Cmdlier

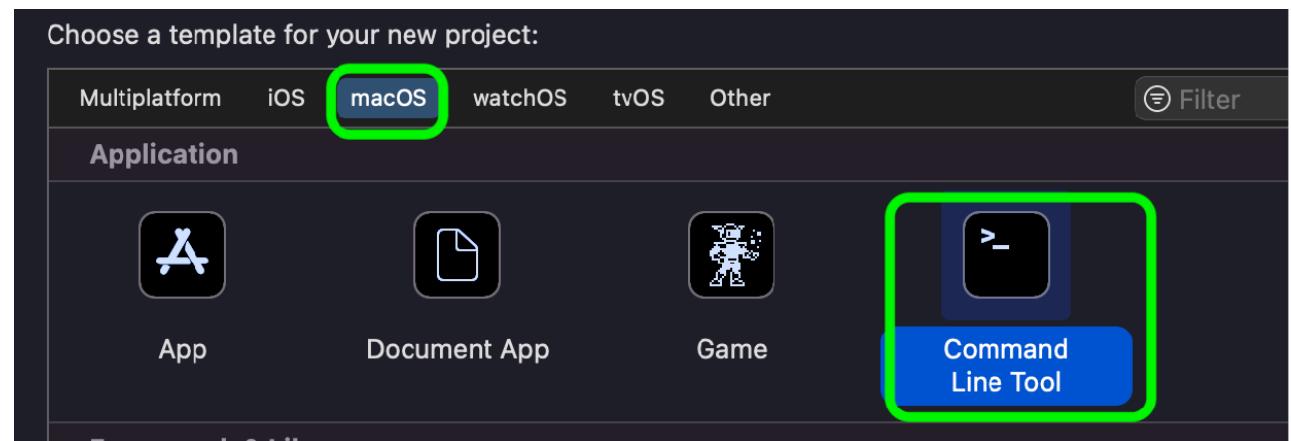


# 1' progetto: Hello World (2)

3) Scegliere

macOS E

Command line Tools

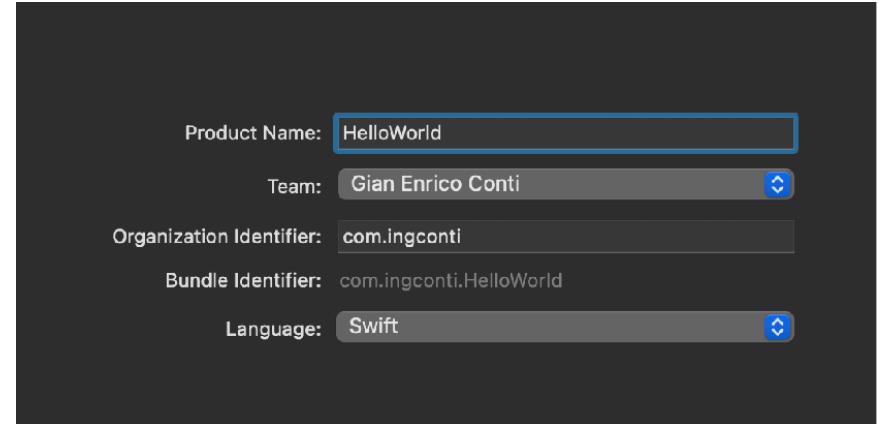


Altre "combinazioni" non servono/non vanno usate.

# 1' progetto: Hello World (3)

- A) Product name:  
HelloWorld

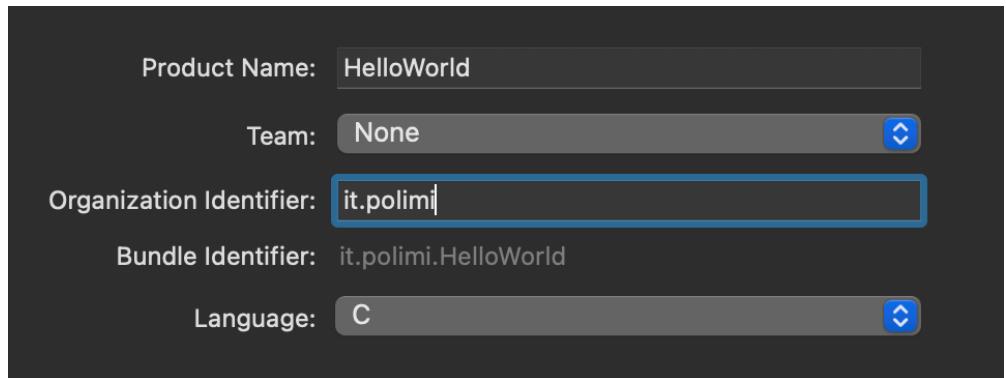
(Non usare spazi... ogni altro nome va benissimo...)



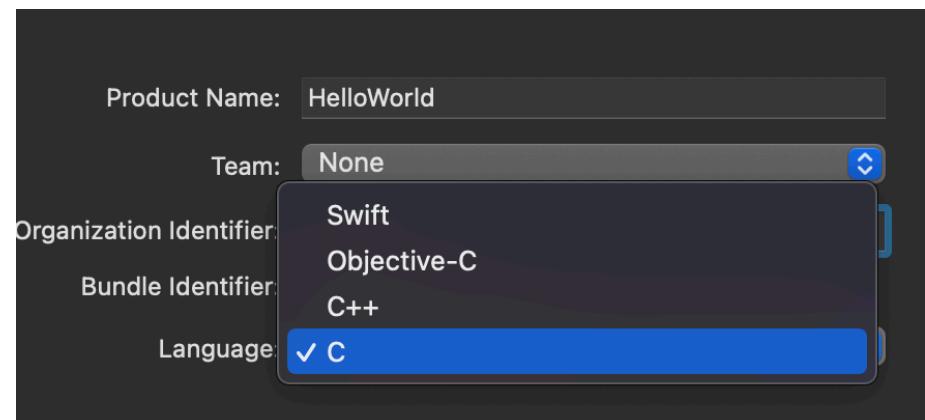
- B) Team: None

- C) Org. Identifier: it.polimi

(A meno ne abbiate già uno...)

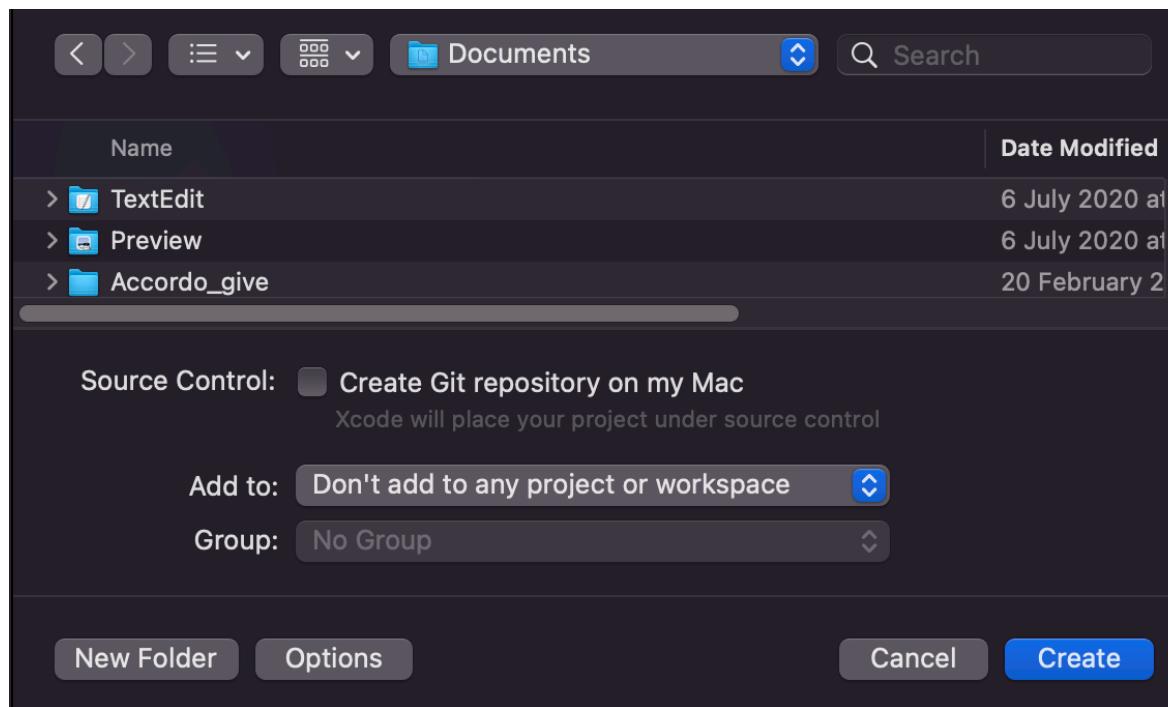


- D) linguaggio "C"



# 1' progetto: Hello World (4)

Salvare in cartella a vs scelta.. (evitare desktop...)

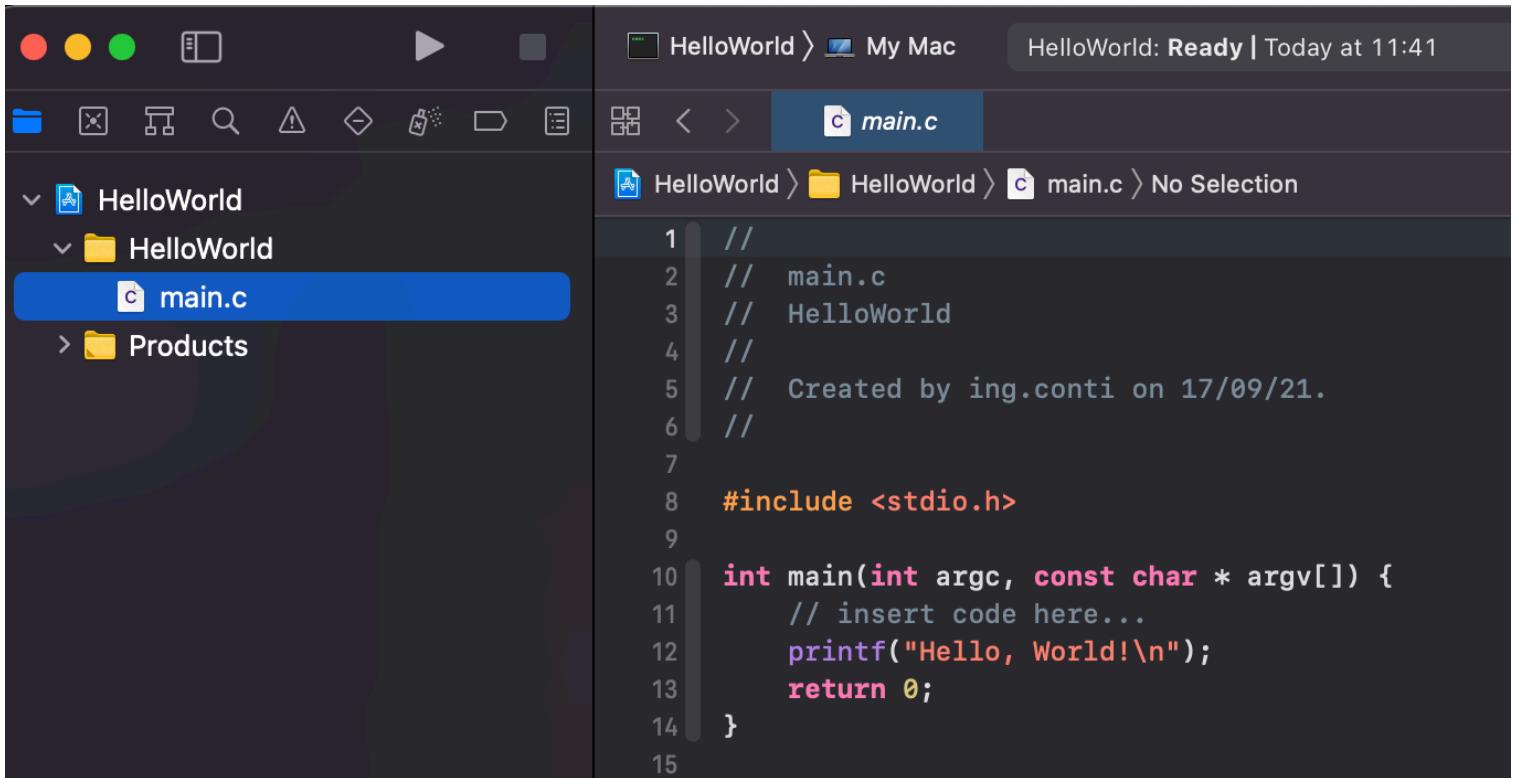


"Git" non serve, nè workspace, per noi.

**"Create"...**

# 1' progetto: Hello World (5)

Apparirà:



The screenshot shows the Xcode interface on a Mac. The top status bar indicates "HelloWorld: Ready | Today at 11:41". The left sidebar shows a project structure with a "HelloWorld" group containing "main.c" (which is selected and highlighted in blue) and "Products". The right pane displays the contents of "main.c". The code is as follows:

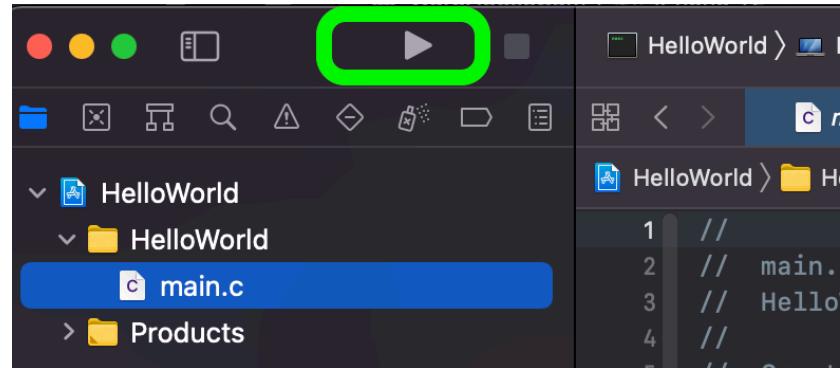
```
//  
// main.c  
// HelloWorld  
//  
// Created by ing.conti on 17/09/21.  
  
#include <stdio.h>  
  
int main(int argc, const char * argv[]) {  
    // insert code here...  
    printf("Hello, World!\n");  
    return 0;  
}
```

Fate click su main.c: AVETE un programma già completamente **funzionante!**

# 1' progetto: Hello World (6) RUN

Cliccare sul pulsante

RUN (non è "play 😊 )



Apparirà:

The screenshot shows the Xcode interface with the code editor open to the 'main.c' file. The code is as follows:

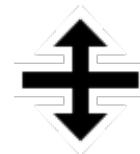
```
1 //  
2 // main.c  
3 // HelloWorld  
4 //  
5 // Created by ing.conti on 17/09/21.  
6 //  
7  
8 #include <stdio.h>  
9  
10 int main(int argc, const char * argv[]) {  
11     // insert code here...  
12     printf("Hello, World!\n");  
13     return 0;  
14 }
```

Below the code editor is the 'Output' tab, which displays the program's output. The text 'Hello, World!' and 'Program ended with exit code: 0' is shown, and it is also highlighted with a green box.

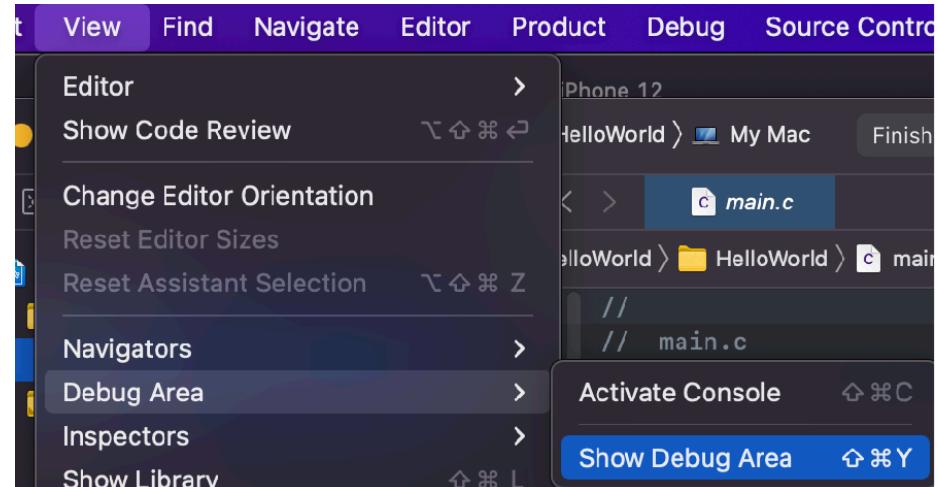
# 1' progetto: Hello World (7) Area inferiore

Se non vedete la zona inferiore..

A) spostate le zone con il cursore



B) oppure Cmd Shift Y



# 1' progetto: Hello World (9) errors

Se avete errori,

```
#include <stdio.h>

int main(int argc, const char * argv[]) {
    // insert code here...

    int n
    |
```

Expected ';' at end of declaration

Appare diagnostica.

Se fate click sul pulsante rosso, FIXA lui!