



UNIVERSITÀ
DI TORINO

Laboratorio di Programmazione I LAB-C-T1-T2

Lezione n. 1: Introduzione al corso

Alessandro Mazzei

Slides Credit: Elvio Amparone

- Come è organizzato il laboratorio
 - Introduzione
 - Come accedere ai PC del laboratorio Turing
- L'interfaccia a riga di comando
 - Cos'è e come funziona
 - Comandi
 - Percorsi dei file
- Programmare con il linguaggio C
 - Installazione del compilatore C
 - Editare un semplice programma
 - Come si compila un programma C
 - Leggere e scrivere con il terminale
 - Esercizi

- Come è organizzato il laboratorio
 - Introduzione
 - Come accedere ai PC del laboratorio Turing
- L'interfaccia a riga di comando
 - Cos'è e come funziona
 - Comandi
 - Percorsi dei file
- Programmare con il linguaggio C
 - Installazione del compilatore C
 - Editare un semplice programma
 - Come si compila un programma C
 - Leggere e scrivere con il terminale
 - Esercizi

- 30 ore / 10 lezioni
- **Docente:** Alessandro Mazzei
alessandro.mazzei@unito.it
- **Informazioni su lezioni e orari:** via pagina Moodle
- **Ricevimento studenti:** scrivere via mail, oppure usare forum Moodle. Usate solo la posta elettronica di ateneo: nome.cognome@edu.unito.it . Trovate gli indirizzi email dei docenti sulle pagine Moodle del corso.

- **Moodle** è il CMS (*content management system*) che usiamo per gli insegnamenti del corso di studio
- **Link:** <https://informatica.i-learn.unito.it/>
Anno accademico 23/24 ▶ Primo anno Laurea DM270 ▶
PROGRAMMAZIONE I LAB (Progl-LabC1) [Cognomi P-Z, matricola dispari]
PROGRAMMAZIONE I LAB (Progl-LabC2) [Cognomi P-Z, matricola pari]

⚠ Iscrivetevi alle pagine
del corso di teoria e di
laboratorio!

Per il laboratorio Turing:

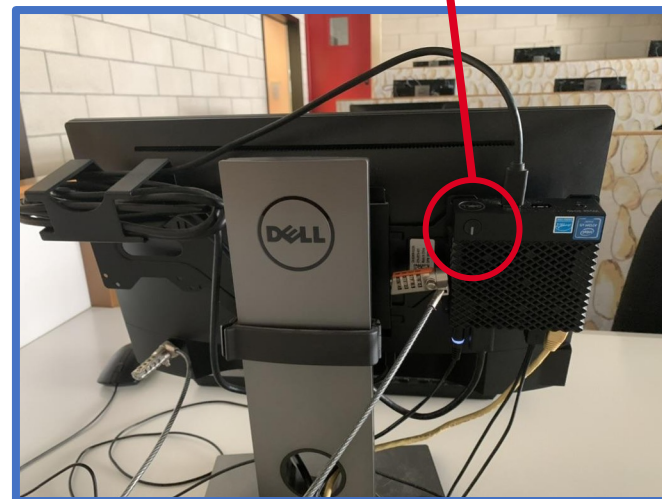
- Accesso con il **login di Ateneo**:

```
username: cognome.nome  
password: *****
```

(lo stesso login che usate per accedere a www.unito.it) e per la posta elettronica cognome.nome@edu.unito.it

- Potete avviare l'ambiente Windows o Linux.
- **⚠ Attenzione al layout della tastiera e ai caratteri speciali nella password!**
- Trovate già l'ambiente C installato, con un editor di testo.
- Potete anche usare un vostro portatile (ma dovrete installare l'ambiente C).

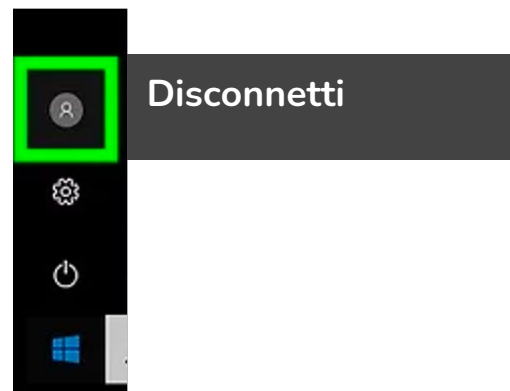
Tasto di
Accensione



Come funzionano i PC del laboratorio



- Ogni volta che accedete alle macchine del laboratorio Turing con il vostro login, il PC riparte da una configurazione di partenza, senza modifiche e senza dati.
- **⚠ I file che scrivete vengono persi al logout!**
- Ricordatevi di salvare i file che vi servono prima del logout - potete usare ad esempio il vostro Drive Google.
- Non potete spegnere i PC, per uscire dovete Disconnettervi.



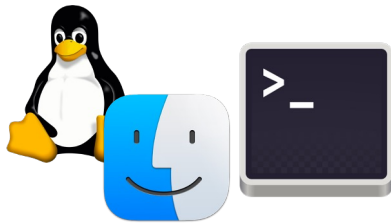
- Durante le esercitazioni in laboratorio è vietato scattare fotografie, girare video, registrare suoni ed in generale avere comportamenti non adeguati al contesto (*si può dormire ma non russare*).
- Si ricorda che tutto il materiale prodotto è protetto da diritto d'autore; può essere utilizzato per finalità di studio e di ricerca a uso individuale e non può essere utilizzato per finalità commerciali, per finalità di lucro anche indiretto (per es. non può essere condiviso su piattaforme online a pagamento o comunque su servizi erogati a scopo di lucro o su siti che guadagnano con introiti pubblicitari).
- È inoltre vietata la condivisione su qualsiasi social media di materiale coperto da diritto d'autore, salvo materiale coperto da licenze Creative Commons.
- Si richiama l'attenzione degli/delle studenti/studentesse a un uso consapevole e corretto dei materiali resi disponibili dalla comunità universitaria, nel rispetto delle disposizioni del codice etico di Ateneo.

- Come è organizzato il laboratorio
 - Introduzione
 - Come accedere ai PC del laboratorio Turing
- L'interfaccia a riga di comando
 - Cos'è e come funziona
 - Comandi
 - Percorsi dei file
- Programmare con il linguaggio C
 - Installazione del compilatore C
 - Editare un semplice programma
 - Come si compila un programma C
 - Leggere e scrivere con il terminale
 - Esercizi

L'interfaccia a riga di comando

Eseguire operazioni tramite
comandi testuali

- Useremo la **CLI** (*Command Line Interface*) per eseguire manualmente tutti i passi necessari per costruire un programma partendo da un file di testo in linguaggio C.
- Vediamo quindi un breve riassunto di come funziona la CLI del sistema operativo.
- La CLI ha vari nomi:



Shell nei sistemi UNIX
(bash, sh, csh, zsh, ...)



Command prompt in Windows
(cmd.exe, powershell)

ma anche **terminale**, **console**, **riga di comando** (*command line*).

Interagire tramite comandi testuali



- *Interazione linguistica*
- La CLI si presenta come una finestra dove l'utente vede un prompt lampeggiante, e può editare dei comandi sotto forma di linee di testo.
- Ci sono regole specifiche per scrivere i comandi.
- Premendo **INVIO**, il sistema interpreta ed esegue il comando, stampando l'output (o gli eventuali errori).

```
amazon:~ depierro$ ls -l
total 8
drwxr-xr-x  8 depierro staff 272 Jul 10  2014 Books
drwxr-xr-x 10 depierro staff 340 Oct  9 09:13 Desktop
drwxr-xr-x 14 depierro staff 476 Apr 23 16:12 Documents
drwxr-xr-x 86 depierro staff 2024 Jul 23 09:32 Downloads
drwxr-xr-x 54 depierro staff 1836 Aug  1  2014 Library
drwxr-xr-x  8 depierro staff 272 Mar  4  2015 Movies
drwxr-xr-x  4 depierro staff 136 Mar  4  2015 Music
drwxr-xr-x  1 depierro staff 100 Oct  9 09:46 MyClass.java
drwxr-xr-x 23 depierro staff 782 Jun 12 11:58 Pictures
drwxr-xr-x  5 depierro staff 170 Feb 20  2014 Public
amazon:~ depierro$
publ classmintmacbook ~ $ which javac
/usr/bin/javac
publ classmintmacbook ~ $ which java
/usr/bin/java
publ classmintmacbook ~ $ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
MyCial/games
classmintmacbook ~ $ ls -l
total 52
drwxr-xr-x  4 class class 4096 feb 18  2014 Applica
-rw-r--r--  1 class class 144 dic  6  2013 C:\nppd
3 erdrwxr-xr-x 5 class class 4096 nov  3  2014 Desktop
amadrwxr-xr-x 2 class class 4096 dic  3  2013 Downloa
drwxr-xr-x 2 class class 4096 mar  1  2014 eps
drwxr-xr-x 2 class class 4096 dic  3  2013 Music
drwxr-xr-x 2 class class 4096 feb 18  2014 nets
drwxr-xr-x 2 class class 4096 dic  3  2013 Picture
drwxr-xr-x 2 class class 4096 mar  1  2014 ps
drwxr-xr-x 2 class class 4096 dic  3  2013 Public
drwxr-xr-x 2 class class 4096 dic  3  2013 templat
drwxr-xr-x 2 class class 4096 dic  3  2013 Videos
classmintmacbook ~ $
```


Sintassi dei comandi



Un comando ha un **nome**, e può essere seguito da **opzioni** ed **argomenti**.

nome_comando **opzioni** **argomenti**

- le **opzioni** definiscono modi alternativi nei quali un comando può operare
- gli **argomenti** sono i dati o i file sui quali il comando opera

 Prompt dei comandi (Windows)

```
> date /t  
> systeminfo  
> echo ciao, come stai
```

 Shell dei comandi (Unix)

```
$ date  
$ uname -a  
$ echo "ciao, come stai"
```

⚠ Nella shell UNIX, le **opzioni** sono introdotte da un trattino '-', in Windows dallo slash '/'

- Alcuni **comandi**, detti **integrati** (*built-in*), sono riconosciuti e implementati direttamente dalla shell/command-prompt.
- Tutti gli altri **comandi** sono programmi **esterni**, che vengono avviati per eseguire il comando imputato tramite CLI.
- Quando viene dato un **comando** esterno, la shell carica in memoria il programma corrispondente e lo esegue con gli **argomenti** e le **opzioni** indicate.
- La shell cerca il **comando** esterno seguendo il percorso dei comandi (**PATH**), memorizzato in una *variabile d'ambiente*.

Esercizio: visualizzare il **PATH**



- Possiamo dare un comando per stampare il valore della variabile d'ambiente PATH:

> **echo %PATH%**

(Windows)

\$ **echo \$PATH**

(Unix)

Molti comandi operano specificando degli **argomenti** che sono nomi di *file* sul disco (sui quali effettuano delle operazioni).

Esempi:

```
$ rm importante.txt
```

```
$ cp file-su-storage-USB.mp3 lettore-portatile/
```

È essenziale poter referenziare i nomi dei file nel sistema delle cartelle in modo preciso.

- Il **percorso** (*pathname*) indica in modo preciso i file

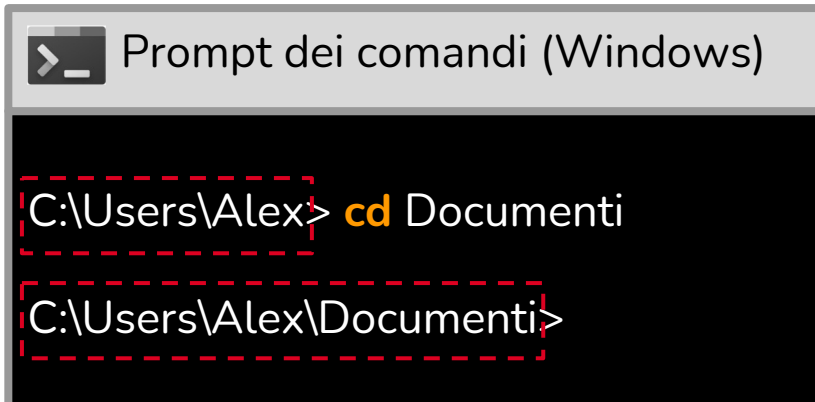
- Il **percorso assoluto** di un file è il nome completo del file nel sistema delle cartelle.
- Si forma componendo in ordine i nomi delle cartelle da attraversare a partire dalla cartella radice del sistema per arrivare al file. Gli elementi vengono separati dal carattere slash / in Unix, o dal carattere backslash \ in Windows.

Esempi:

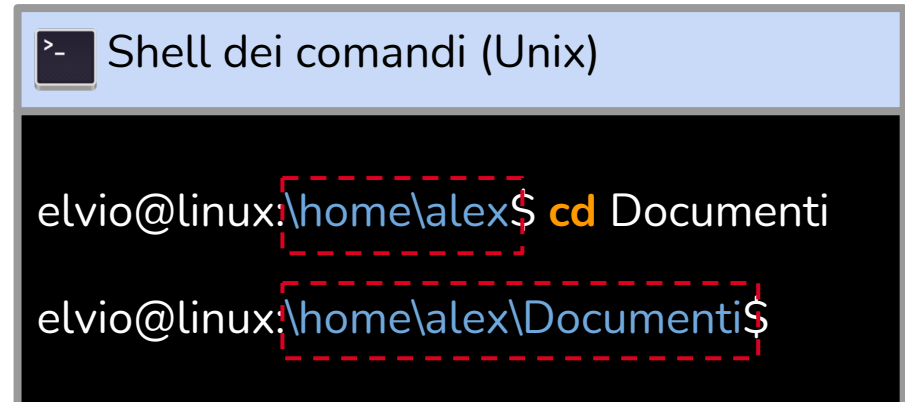
/home/alex\$ rm /home/alex/Documenti/cosedafare.txt	(UNIX)
C:\Users\Alex> del C:\Users\Alex\Documenti\cosedafare.txt	(Windows)

rimuove il file dal nome **cosedafare.txt** che si trova nella cartella **Documenti** dell'utente alex;

- Il pathname assoluto è uno schema preciso ma lungo e verboso da scrivere. In molti contesti si può abbreviare facendo leva sul concetto di **percorso relativo** alla **cartella di lavoro** (*working directory*).



```
Prompt dei comandi (Windows)  
C:\Users\Alex> cd Documenti  
C:\Users\Alex\Documenti>
```



```
Shell dei comandi (Unix)  
elvio@linux:\home\alex$ cd Documenti  
elvio@linux:\home\alex\Documenti$
```

- In Windows la working directory viene indicata prima del simbolo '>' nel prompt dei comandi (prima del '\$' nella shell Unix).

NOTA: **cd** = **c**hange **d**irectory

- Il prompt dei comandi ha sempre una **cartella di lavoro**.
- I nomi dei file passati come argomenti ai comandi possono essere **relativi alla cartella di lavoro**, oppure **assoluti**.
- **Percorso relativo**: è simile al nome assoluto del file ma la cartella iniziale è la cartella di lavoro.

Esempio:

se C:\Users\Alex è la cartella di lavoro, allora il seguente comando:

```
C:\Users\Alex> del Documenti\cosedafare.txt
```

è equivalente a

```
C:\Users\Alex> del C:\Users\Alex\Documenti\cosedafare.txt
```

In Windows le lettere (seguite dal due punti ':') indicano i dischi

- **C:** disco locale dove trovare i programmi (compilatore e interprete java, notepad++)
- **D:** o altri nomi: altri volumi che avete sul vostro PC (in laboratorio Turing abbiamo solo C:)

In **laboratorio Turing** il prompt dei comandi parte nella cartella di lavoro **C:\TDM-GCC-64**

➡ *ma non vogliamo scrivere in una cartella di sistema!!*

Per andare nella nostra cartella Documenti, scriviamo:

cd C:\Users**nome_utente**\Documents

Comandi per il command prompt di Windows

cd <i>dir</i>	Cambia la working directory in <i>dir</i>
cd ..	Cambia la working directory con la directory superiore
dir	Elenca il contenuto della working directory
dir <i>path</i>	Elenca il contenuto della directory <i>path</i>
mkdir <i>dirA</i>	Crea una nuova directory <i>dirA</i> vuota
rmdir <i>dirA</i>	Rimuove la directory <i>dirA</i> (deve essere vuota)
copy <i>fileA</i> <i>fileB</i>	Copia il contenuto del <i>fileA</i> nel nuovo <i>fileB</i>
copy NUL <i>fileA</i>	Crea un file vuoto di nome <i>fileA</i>
ren <i>fileA</i> <i>fileB</i>	Cambia il nome del <i>fileA</i> in <i>fileB</i>
move <i>pathToA</i> <i>pathToB</i>	Sposta il file <i>pathToA</i> nella nuova posizione <i>pathToB</i>
del <i>fileA</i>	Elimina il <i>fileA</i>
type <i>fileA</i>	Stampa il contenuto del file <i>fileA</i> nel terminale
start notepad++ <i>fileA</i>	Apri il <i>fileA</i> con il programma Notepad++
C: D: X:	Cambia il disco attivo (radice del filesystem)

Comandi per la shell Unix

cd <i>dir</i>	Cambia la working directory in <i>dir</i>
cd ..	Cambia la working directory con la directory superiore
ls	Elenca il contenuto della working directory
ls <i>path</i>	Elenca il contenuto della directory <i>path</i>
mkdir <i>dirA</i>	Crea una nuova directory <i>dirA</i> vuota
rmdir <i>dirA</i>	Rimuove la directory <i>dirA</i> (deve essere vuota)
cp <i>fileA</i> <i>fileB</i>	Copia il contenuto del <i>fileA</i> nel nuovo <i>fileB</i>
touch <i>fileA</i>	Crea un file vuoto di nome <i>fileA</i>
mv <i>fileA</i> <i>fileB</i>	Cambia il nome del <i>fileA</i> in <i>fileB</i>
mv <i>pathToA</i> <i>pathToB</i>	Sposta il file <i>pathToA</i> nella nuova posizione <i>pathToB</i>
rm <i>fileA</i>	Elimina il <i>fileA</i>
cat <i>fileA</i>	Stampa il contenuto del file <i>fileA</i> nel terminale
open <i>fileA</i> xdg-open <i>fileA</i> (Linux)	Apri il <i>fileA</i> con un editor di testo.

Alcuni dettagli sui comandi



Ci sono due percorsi speciali che si possono usare:

- Il percorso “.” è la cartella di lavoro

Esempio: supponiamo che la cartella di lavoro sia Documenti

I due comandi seguenti sono del tutto equivalenti:

> **del** .\verbale.txt (Windows)

> **del** verbale.txt (Windows)

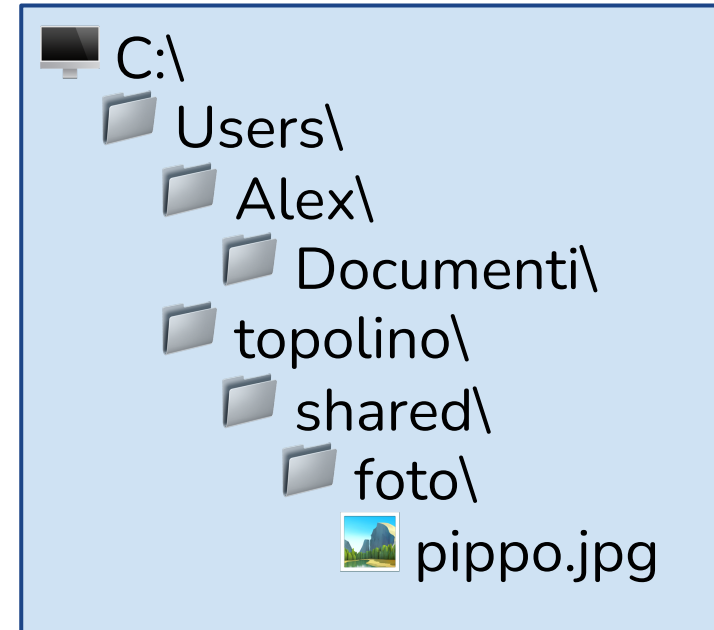
Alcuni dettagli sui comandi



- Il percorso “..” è la cartella che contiene la cartella di lavoro, e si usa per risalire la gerarchia delle cartelle

Esempio:

- Supponiamo che la cartella di lavoro sia **Documenti**, e che questa sia contenuta nella cartella **Alex**
- Si vuole copiare in **Documenti** il file **pippo.jpg** memorizzato nella cartella **shared\foto** dell'utente **topolino**



Allora possiamo usare il seguente comando:

```
C:\Users\Alex\Documenti> copy ..\..\topolino\shared\foto\pippo.jpg .
```


Tramite CLI eseguire i seguenti comandi:

1. Cambiare la cartella di lavoro in
`C:\Users\nome_utente\Documents`
2. Creare una nuova cartella **PROVA**, e al suo interno creare un nuovo file `importante.txt` vuoto
3. Rinominare il file in `rilevante.txt`
4. Fare una copia di `rilevante.txt` (usando pathname assoluto e poi relativo)
5. Cancellare la copia
6. Impostare la cartella di lavoro sulla cartella che contiene **PROVA**
7. Editare il file `rilevante.txt` con notepad++
8. Passare ad un altro disco (C: o Z:, se disponibile) (solo Windows)
9. Rientrare nella cartella **PROVA**

- Come è organizzato il laboratorio
 - Introduzione
 - Come accedere ai PC del laboratorio Turing
- L'interfaccia a riga di comando
 - Cos'è e come funziona
 - Comandi
 - Percorsi dei file
- Programmare con il linguaggio C
 - Installazione del compilatore C
 - Editare un semplice programma
 - Come si compila un programma C
 - Leggere e scrivere con il terminale
 - Esercizi

Programmare con il linguaggio C

Introduzione alla compilazione di
un programma

Il linguaggio C (in inglese si pronuncia /si:/) è un linguaggio di programmazione

- procedurale imperativo
- strutturato
- con supporto alla ricorsione
- tipato (*typed*) o tipizzato
- vicino all'hardware



Vi sono molte risorse online che documentano gli aspetti del linguaggio C e della sua libreria.

Tra i più rilevanti:

- **C Standard Library header files**

<https://en.cppreference.com/w/c/header>

- **C Programming/Standard library reference**

https://en.wikibooks.org/wiki/C_Programming/Standard_library_reference

https://en.wikipedia.org/wiki/C_standard_library

- **C Language Library**

<https://cplusplus.com/reference/library/>

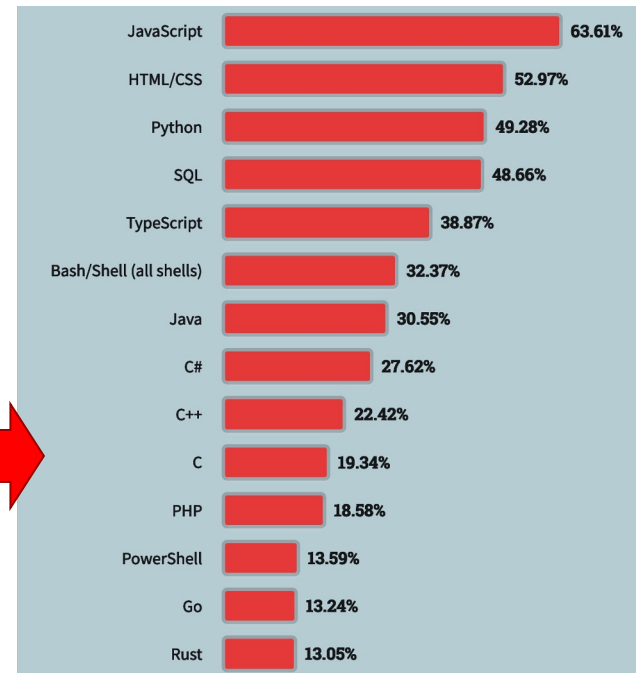
Popolarità del linguaggio C

TIOBE 2023
(indice di popolarità dei linguaggi
basato su motori di ricerca)

Stack Overflow Developer
Survey 2023



Sep 2023	Sep 2022	Change	Programming Language	Ratings	Change
1	1		 Python	14.16%	-1.58%
2	2		 C	11.27%	-2.70%
3	4	▲	 C++	10.65%	+0.90%
4	3	▼	 Java	9.49%	-2.23%
5	5		 C#	7.31%	+2.42%
6	7	▲	 JavaScript	3.30%	+0.48%
7	6	▼	 Visual Basic	2.22%	-2.18%
8	10	▲	 PHP	1.55%	-0.13%
9	8	▼	 Assembly language	1.53%	-0.96%
10	9	▼	 SQL	1.44%	-0.57%



- Esistono molti compilatori per il linguaggio C.
- Alcuni esistono da decenni. Si tratta di software molto importante e molto sviluppato e testato.
- Il software di sistema mondiale si poggia sulla solidità e sull'affidabilità dei compilatori per il linguaggio C!
- Noi useremo un compilatore che si chiama **gcc**

(ma le indicazioni che vedremo sono valide sostanzialmente per tutti i compilatori C disponibili)

Installare il compilatore C



Windows: usiamo TDM-GCC-64

- distribuzione per Windows di **gcc** basata su MinGW
- download da: <https://jmeubank.github.io/tdm-gcc/>



Linux: installare il pacchetto gcc

- **sudo apt install build-essential**
- **sudo dnf install gcc make**



macOS: installare *XCode command line tools*
(da AppStore o sito apple)

Installare TDM-GCC-64 (Windows)



Download da <https://jmeubank.github.io/tdm-gcc/>

tdm-gcc
GCC compiler, Windows-friendly
[home](#) / [about](#) / [download](#) / [donate](#) / [archive](#)

The latest release is based on GCC 10.3.0

MinGW-w64 based
[tdm64-gcc-10.3.0-2.exe](#), 76.6 MB

MinGW.org based
[tdm-gcc-10.3.0.exe](#), 60.2 MB

Home

TDM-GCC 10.3.0 release

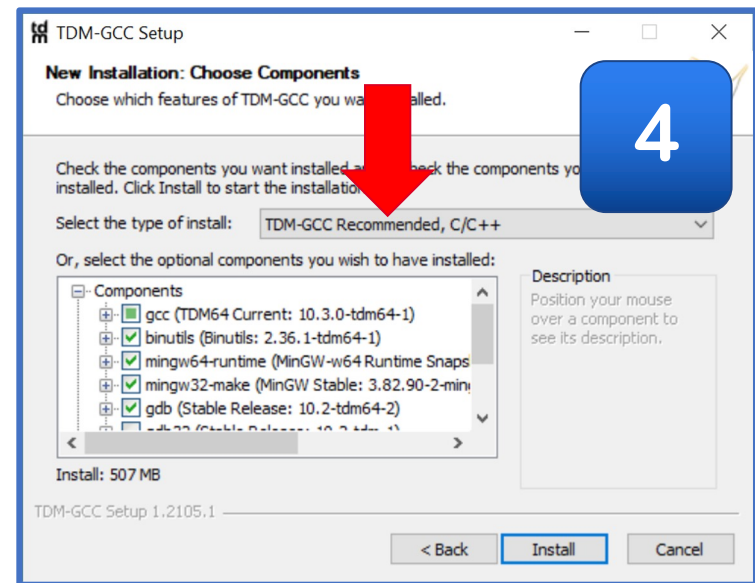
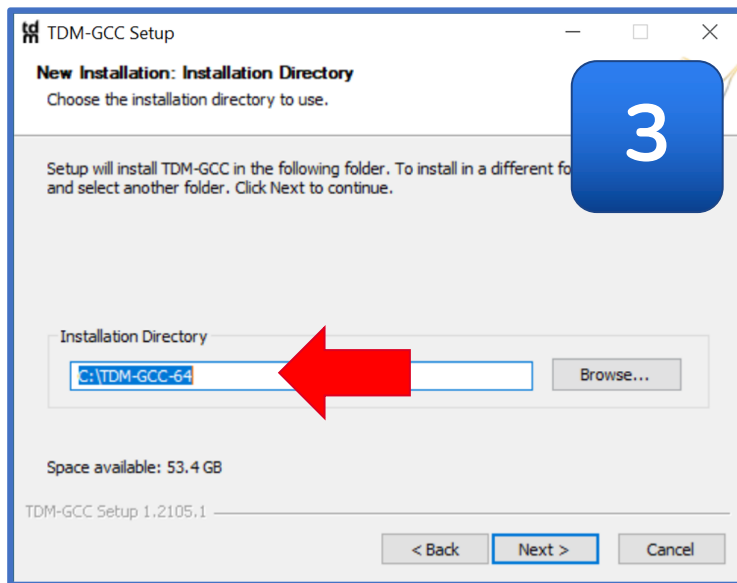
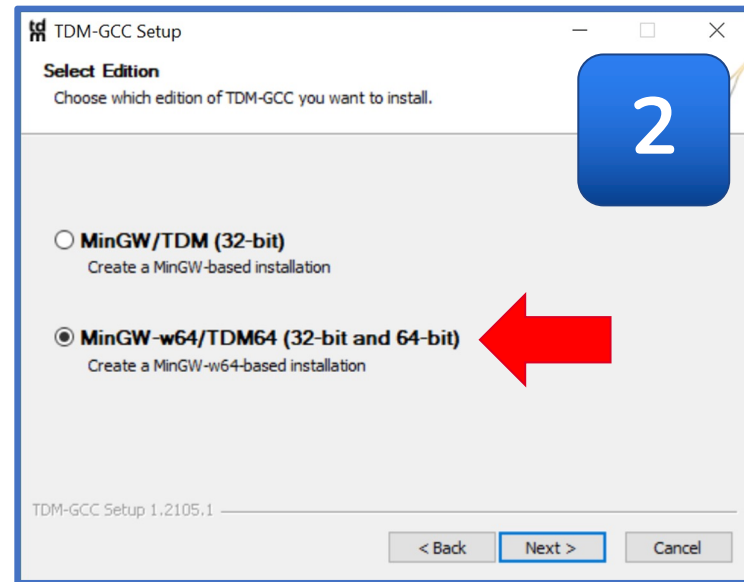
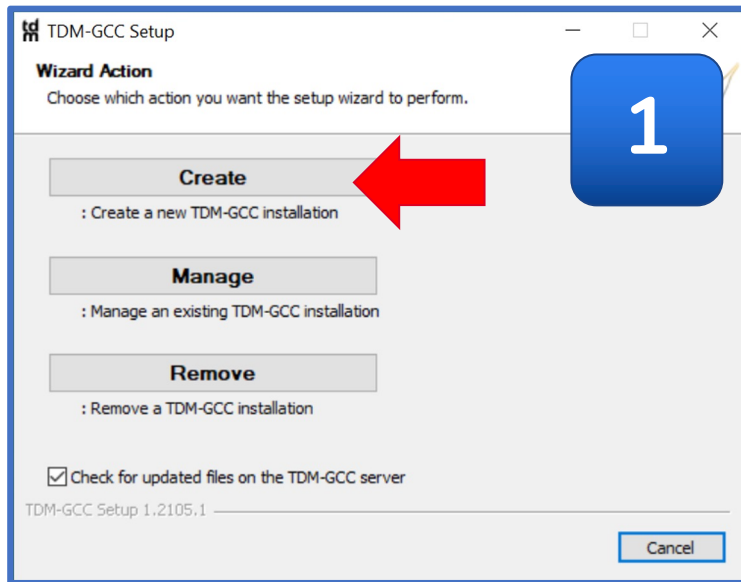
24 MAY 2021 • 1 min read

TDM-GCC 10.3.0 is now available, along with GDB 10.2, binutils 2.36.1, and new versions of the MinGW.org and MinGW-w64 runtime distributions. Thanks for your patience!

Download a TDM-GCC installer:

tdm-gcc-webdl.exe	Minimal online installer. Select the components you want, and it downloads and unpacks them. Either edition, latest release only. (GCC 10.3.0)
tdm64-gcc-10.3.0-2.exe	64+32-bit MinGW-w64 edition. Includes GCC C/C++, GNU binutils, mingw32-make, GDB (64-bit), the MinGW-w64 runtime libraries and tools, and the windows-default-manifest package.
tdm-gcc-10.3.0.exe	32-bit-only MinGW.org edition. Includes GCC C/C++, GNU binutils, mingw32-make, GDB (32-bit), the MinGW.org mingwrt and w32api packages, and the windows-default-manifest package.

Installare TDM-GCC-64 (Windows)

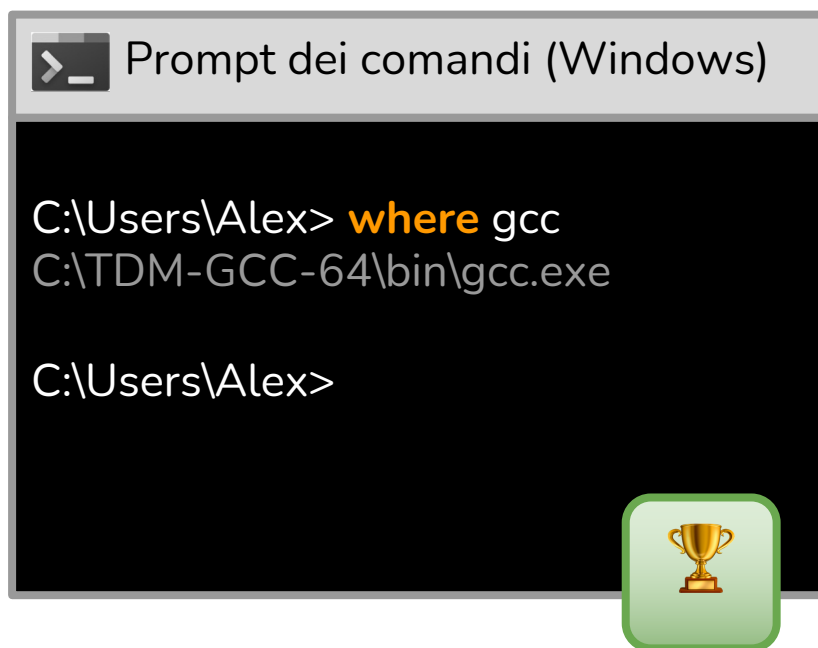


Verifica di installazione



Per verificare che l'installazione sia andata a buon fine, apriamo il prompt dei comandi e scriviamo:

> **where gcc**

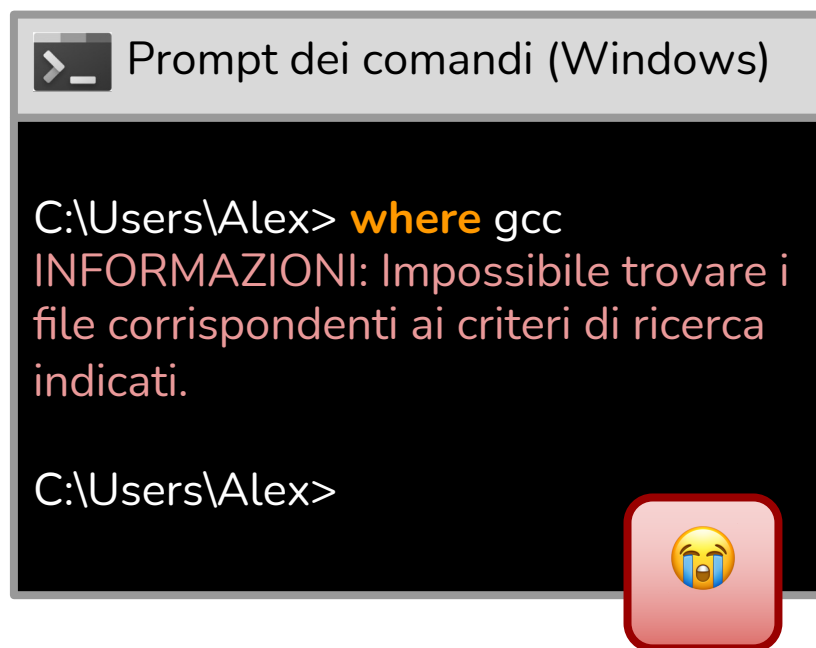


```
Prompt dei comandi (Windows)

C:\Users\Alex> where gcc
C:\TDM-GCC-64\bin\gcc.exe

C:\Users\Alex>
```

A green trophy icon is displayed at the bottom right of the window, indicating a successful installation.



```
Prompt dei comandi (Windows)

C:\Users\Alex> where gcc
INFORMAZIONI: Impossibile trovare i
file corrispondenti ai criteri di ricerca
indicati.

C:\Users\Alex>
```

A crying face emoji is displayed at the bottom right of the window, indicating an unsuccessful installation.

Scrivere il codice C

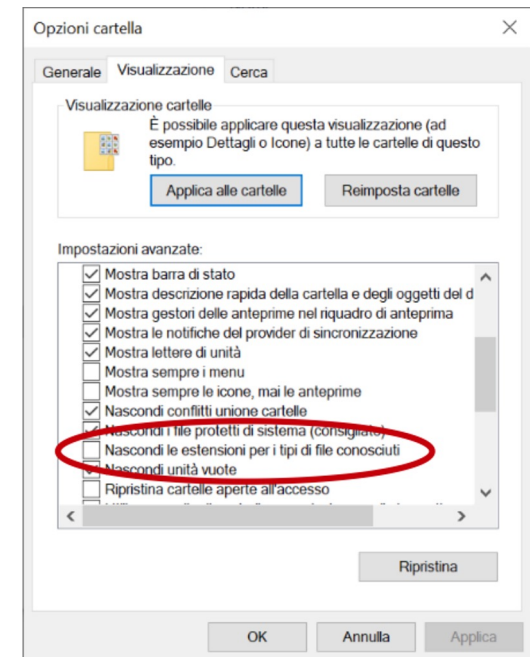
Serve un editor di testo semplice, ma che sia adatto alla scrittura di codice. Notepad non va bene per questo scopo.

Editor suggerito:



- **Notepad++** (Windows)
- ... ma ve ne sono molti altri (gedit, Sublime, Atom, etc.)

Suggerimento: in Windows disabilitare l'opzione che nasconde le estensioni dei nomi dei file per i file di formato "conosciuto". In questo modo verifichiamo che i file di codice che andremo a scrivere abbiano effettivamente estensione .c ("Opzioni" in File Explorer)



Creiamo un file .c da compilare

Creiamo un semplice file sorgente di prova in linguaggio C con nome:

buongiorno.c



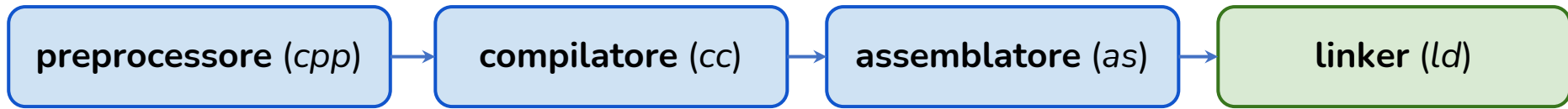
buongiorno.c

```
#include <stdio.h>

int main() {
    printf("Buongiorno!\n");
    return 0;
}
```

Scriviamo il testo in linguaggio C con un editor appropriato.

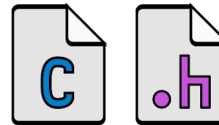
Cosa ci serve per compilare



Gli strumenti che useremo:

- Il compilatore C
- Il collegatore (linker)
- Il disassemblatore

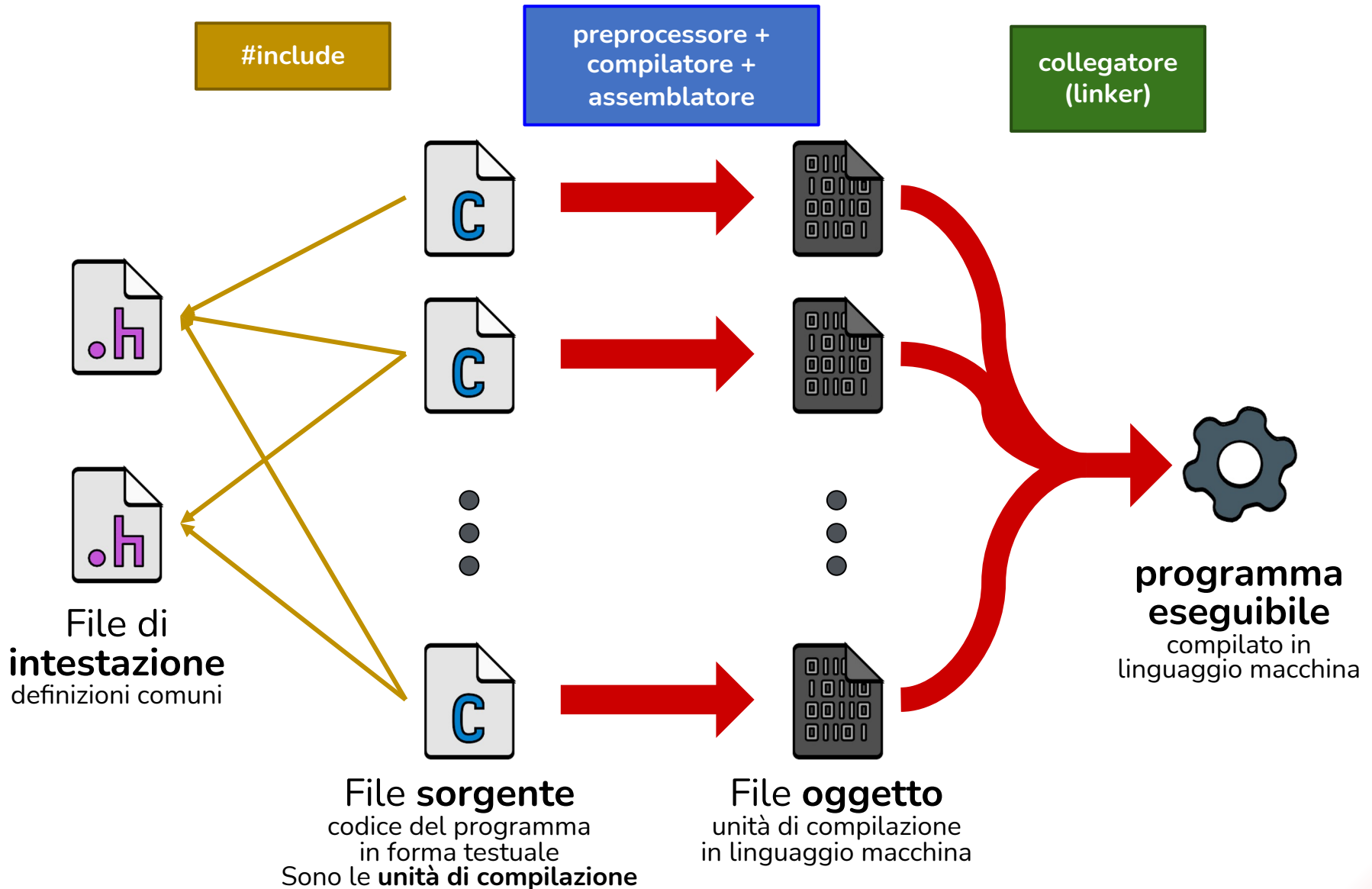
Gestiremo vari tipi di file:



- I file **sorgente** in C (estensioni **.c** e **.h**)
- I file **oggetto** (estensione **.o** oppure **.obj**)
- I **programmi** compilati (estensione **.exe** o nessuna)



Come funziona la compilazione con il C



gcc è il compilatore (**clang** sotto macOS) che traduce il programma C in forma testuale in un file oggetto.

NOTA: stiamo facendo **preprocessing**, **compilazione** vera e propria, ed **assemblamento**.

Unix: `gcc -c sorgente.c -o oggetto.o`

Windows: `gcc -c sorgente.c -o oggetto.obj`



Cosa ci serve per collegare (linkare)



Il linker prende i file oggetto e li unisce in un unico eseguibile.
Il linker è lo stesso comando **gcc**, ma senza l'opzione **-c**

Unix: **gcc** oggetto1.o oggetto2.o **-o** eseguibile
Windows: **gcc** oggetto1.obj oggetto2.obj **-o** eseguibile.exe



Compilare e collegare in un solo passo



Per programmi semplici (es. con un solo file sorgente), è possibile anche compilare e collegare in un solo passo:

Unix: `gcc sorgente.c -o eseguibile`

Windows: `gcc sorgente.c -o eseguibile.exe`



Come disassemblare un programma C



Il disassembler ci mostra una versione testuale del programma compilato.

⚠ **Attenzione:** viene visualizzato il formato assembly dell'eseguibile, non il testo in linguaggio C da cui siamo partiti!

Linux: `objdump -d eseguibile`
macOS: `otool -tv eseguibile`
Windows: `objdump -d eseguibile.exe`



scrivi_intero.c

```
#include <stdio.h>

int main() {
    int number;
    printf("Scrivi un numero intero: ");

    // leggi il numero inserito
    scanf("%d", &number);

    // mostra il numero a video
    printf("Hai scritto: %d\n", number);
    return 0;
}
```

Tabella delle moltiplicazioni



tabella_moltiplicazioni.c

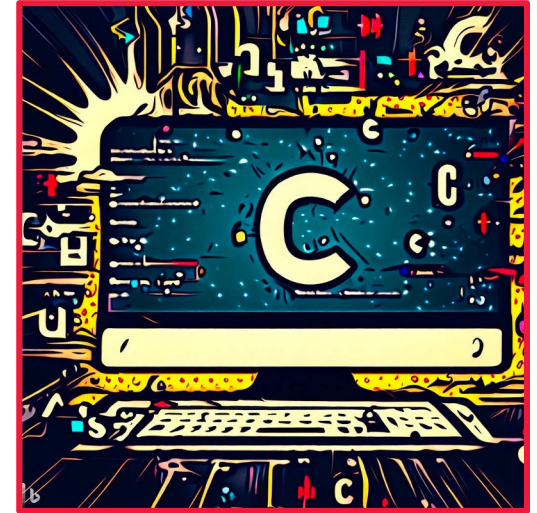
```
#include <stdio.h>
int main() {
    int n;
    printf("Scrivi un numero intero: ");
    scanf("%d", &n);

    // stampa i primi 10 multipli di n
    for (int i = 1; i <= 10; ++i) {
        printf("%d * %d = %d \n", n, i, n * i);
    }
    return 0;
}
```

Primi esercizi con il linguaggio C



Scriveremo i programmi con un editor di testo semplice (ma adatto alla programmazione), ed eseguiremo manualmente i passi che servono per la compilazione.



- Stampa due righe usando due chiamate a **printf**
- Stampa due righe usando una singola chiamata a **printf**
- Chiedi due numeri, e poi stampali
- Stampa la somma di due numeri
- Dichiarare tre numeri interi **x**, **y** e **z**, inizializzati con valori 5, 8 e 11, e stampa il loro prodotto usando **printf**
- Leggere gli errori di compilazione