

Eco-design Digitale di Base per i servizi ICT

Programmazione in C e Python

Introduzione

La programmazione e i suoi linguaggi

```
require ( TEMPLATEPATH.DS."yjscore/yjscore" )
$renderer = $document->loadRender
$options = array( 'style' => "ra
$module = JModuleHelper::getMod
$stopmenu = false; $subnav = fals
Main Menu
if ( $default_menu_style == 1 or $def
    $module->params = "menutype=
    $stopmenu = $renderer->render
    $menuclass = 'horiznav';
    $stopmenuclass = 'top_menu';
elseif ( $default_menu_style == 3 or
    $module->params = "menutype=
    $stopmenu = $renderer->render
    $menuclass = 'horiznav_d';
    $stopmenuclass = 'top_menu_d';
```


Cos'è un linguaggio di programmazione?

*“Un **linguaggio di programmazione** è un sistema di notazione per la scrittura di **programmi** per **computer**. La maggior parte dei linguaggi di programmazione sono **linguaggi formali** basati su testo...”*

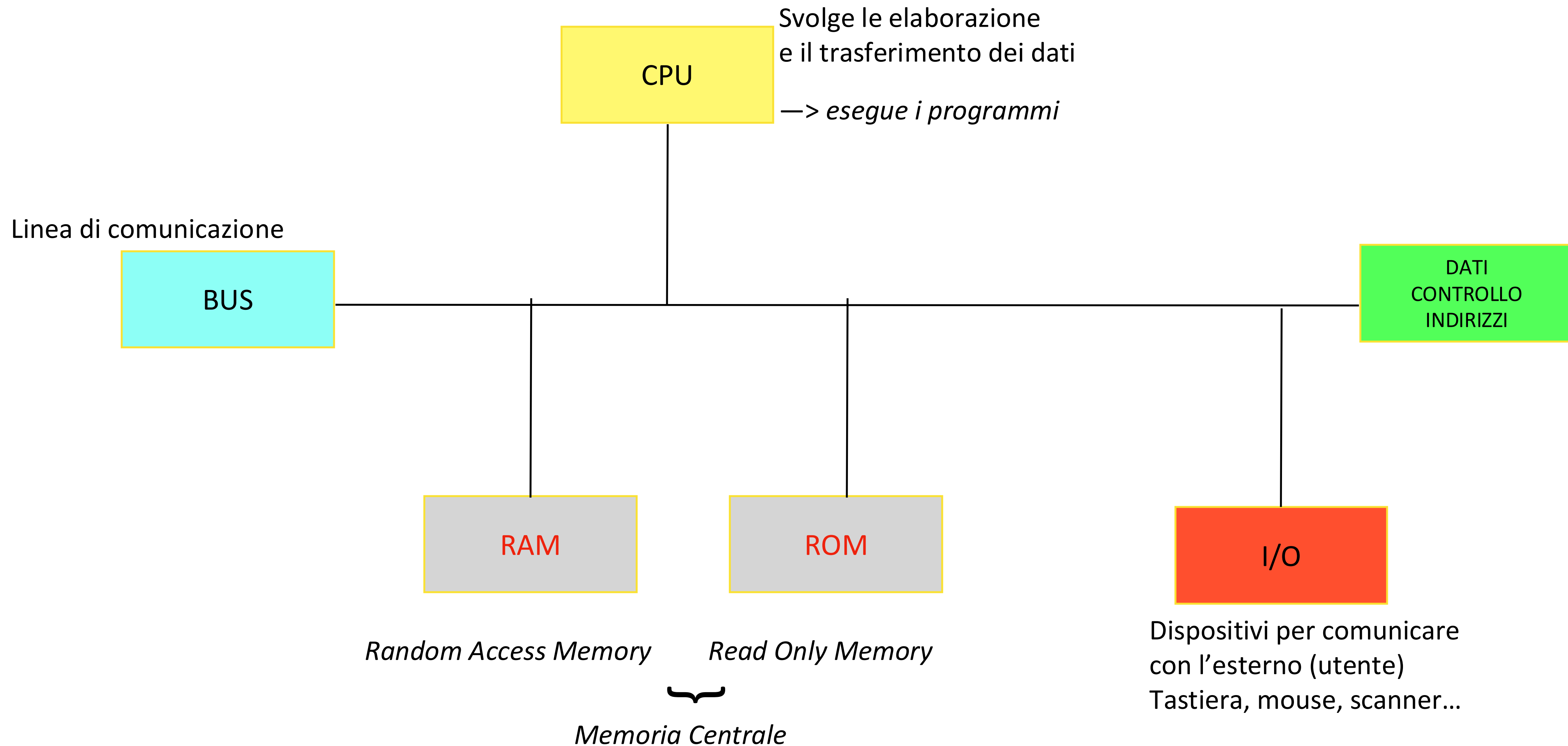
[Fonte: Wikipedia]

Programma: insieme ordinato di istruzioni scritte in un linguaggio interpretabile da un computer per esprimere un algoritmo in grado di risolvere dei problemi.

```
0h init:      LHI R30, 0x4000      ; set R30 = 0x40000000h
4h           SW R29, 0x0000(R30)   ; save R29 in 0x40000000h (RAM)
8h           SW R28, 0x0004(R30)   ; save R28 in 0x40000004h (RAM)
Ch           LHI R29, 0xC000      ; set R29 = 0xC0000000h (STARTUP address)
10h          LBU R28, 0x0000(R29)   ; read STARTUP signal into R28
14h          BEQZ R28, handler     ; if STARTUP == 0 then jump to (interrupt) handler
18h          SB R0, 0x0004(R29)    ; set STARTUP = 0
1Ch          J main               ; jump to main:
20h handler:  LHI R29, 0x3000      ; set R29 = 0x30000000h (INPUT_PORT address)
24h          LBU R28, 0x0004(R29)   ; read interrupt INPUT_PORT signal into R28
28h          BNEZ R28, input_port  ; if INT_I != 0 then jump to (interrupt) input_port
2Ch          LHI R29, 0x9000      ; set R29 = 0x90000000h (LED address)
30h          SB R0, 0x0004(R29)    ; switch LED signal
34h          LW R28, 0x0004(R30)    ; restore R28 value from memory (RAM)
38h          LW R29, 0x0000(R30)    ; restore R29 value from memory (RAM)
3Ch          RFE
40h
44h
48h main:     ADDI R1,R0,0x0000     ; Fibonacci sequence
4Ch          ADDI R2,R0,0x0001     ; set R1 = 0
50h          ADDI R3,R0,0x0001     ; set R2 = 1
54h          ADDI R4,R0,0x0014     ; set R3 = 1
58h loop:     ADD R1,R2,R0          ; set counter R4 = 0x14
5Ch          ADD R2,R3,R0          ; copy R2 into R1
60h          ADD R3,R2,R1          ; copy R3 into R2
64h          SUBI R4,R4,0x0001     ; R3 = R2 + R1
68h          RFE                  ; decrease R4 by 1
```

Esempio di codice Assembly

Computer oggi



Un po' di storia...

The Evolution Of Computer Programming Languages

 $\frac{1}{2}ex$ 

Assembler



C



Fortran



C++



Java



Ruby

Paradigmi di programmazione

- **Procedurale:** Il codice è suddiviso in funzioni/procedure. (*divide et impera*)
- **Strutturata:** ogni algoritmo può essere scritto con sequenze, selezioni e iterazioni.
- **Orientata agli oggetti (OOP):** Si basa su *classi* e *oggetti* (istanze delle classi).
- **Dichiarativa:** si concentra sul “*cosa*” fare più che sul “*come*”.
- **Programmazione funzionale:** funzioni matematiche pure.
- **Programmazione logica:** basata sulla logica del primo ordine.

Obiettivi del corso

- Strutture di controllo, funzioni e gestione della memoria (C)
- Tipi di dato complessi, puntatori e gestione dinamica (C)
- Scripting, OOP di base e utilizzo di librerie standard (Python)
- Best practice, ottimizzazione

Struttura del corso

- 04 giugno dalle ore 14:00 alle ore 18:00
- 06 giugno dalle ore 14:00 alle ore 18:00
- 11 giugno dalle ore 14:00 alle ore 18:00
- 16 giugno dalle ore 14:00 alle ore 18:00
- 20 giugno dalle ore 14:00 alle ore 18:00
- 23 giugno dalle ore 14:00 alle ore 18:00


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

```
int main()  
{
```

```
    int c, n, fact = 1;
```

```
    printf("Enter a number: ");  
    scanf("%d", &n);
```

```
    for (c = 1; c <= n; c++)  
        fact = fact * c;
```

```
    printf("Number of %d = %d", n, fact);
```

```
    return 0;
```

```
}
```

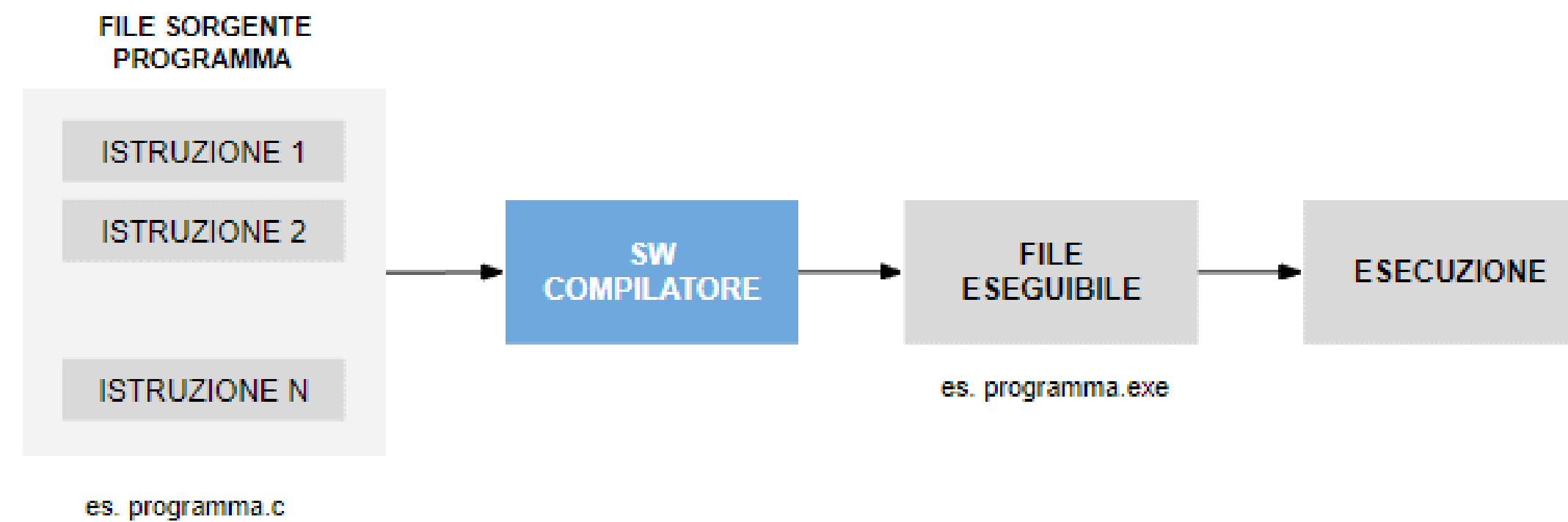
Linguaggio C

Il mattone dei sistemi operativi

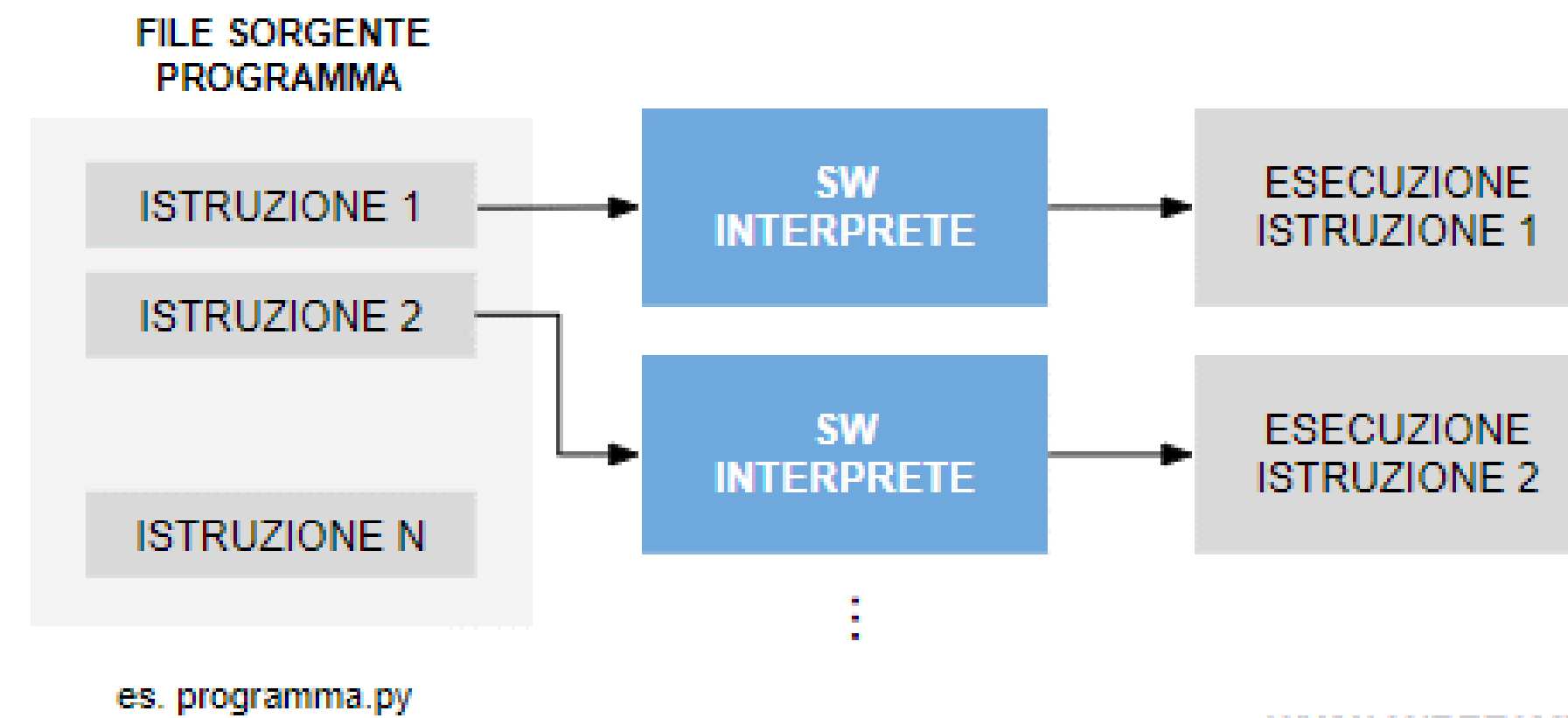
Linguaggio C

- Cos'è?
Un linguaggio di programmazione **compilato, general-purpose** e **tipizzato staticamente**.
- **Tipizzazione statica**: Il tipo delle variabili è noto a tempo di compilazione e non può cambiare.
- **Portabilità**: I programmi scritti in C possono essere compilati su diversi sistemi operativi con minime modifiche.

Compilato vs Interpretato



WWW.ANDREAMININI.COM



WWW.ANDREAMININI.COM

Ciao mondo!

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

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


Fasi di esecuzione di un programma

Dall'editing alla compilazione

1. **Editing:** il programmatore scrive il codice usando un editor (es. nano, o IDE come Visual Studio)
2. **Preelaborazione:** il preprocessore gestisce direttive come `#include` e `#define`
3. **Compilazione:** il compilatore converte il codice in linguaggio macchina generando un file oggetto (.o)

Fasi di esecuzione di un programma

Dal linking al loading

4. **Linking:** il linker collega il file oggetto con le librerie necessarie, producendo un eseguibile (di default a.out)

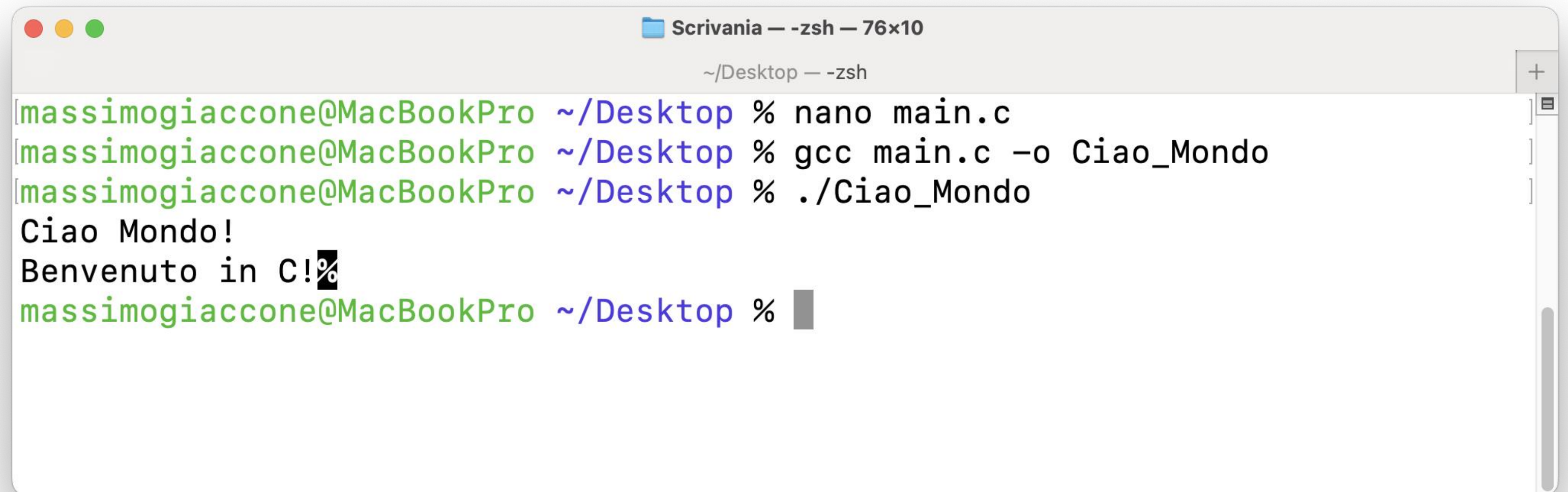
5. **Loading:** il loader carica l'eseguibile in memoria RAM

Fasi di esecuzione di un programma

Esecuzione

6. Esecuzione: la CPU esegue il programma, istruzione per istruzione

Esempio:



```
Scrivania — -zsh — 76x10
~/Desktop — -zsh
massimogiaccone@MacBookPro ~/Desktop % nano main.c
massimogiaccone@MacBookPro ~/Desktop % gcc main.c -o Ciao_Mondo
massimogiaccone@MacBookPro ~/Desktop % ./Ciao_Mondo
Ciao Mondo!
Benvenuto in C!%
massimogiaccone@MacBookPro ~/Desktop %
```

Lessico del C

Variabili

- **Variabili:** locazione di memoria a cui viene associato un nome e che contiene un valore.
- Sintassi: **tipo nome_variabile;**
- Esempio: **int età;**
- Tipi comuni: **int, float, char, double**

Lessico del C

Commenti

- I **commenti** servono a documentare il codice e non vengono eseguiti.

```
9  #include <stdio.h>
10
11 int main()
12 {
13     printf("Hello World"); //commento in una sola riga
14
15     /*
16      * commento multiriga
17      */
18
19     return 0;
20 }
```



È buona prassi commentare il codice in maniera da renderlo più leggibile

Lessico del C

Categorie

- **Parole chiavi:** termini riservati del linguaggio che non possono essere utilizzati come nomi di variabili; usate per definire il comportamento di un programma

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Lessico del C

Costanti

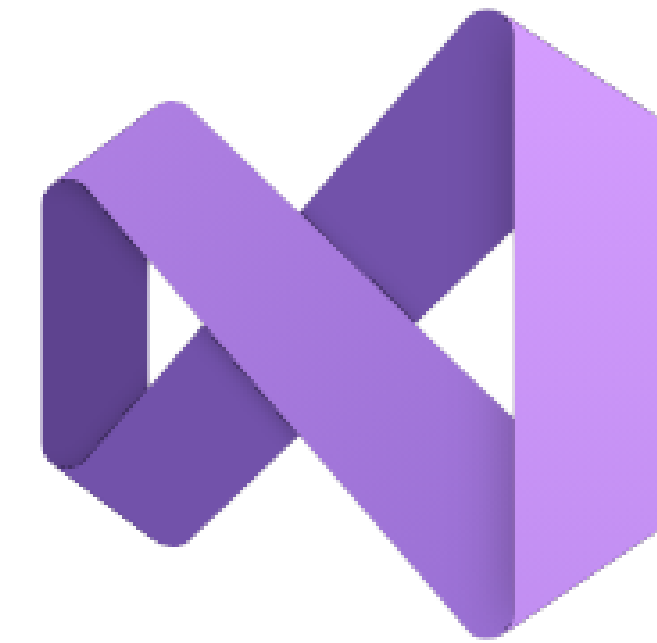
- Una **costante** è un valore che non cambia durante l'esecuzione del programma.

```
12 #define PI 3.14 // (direttiva del preprocessore)
13 const int giorniSettimana = 7; // (sintassi C)
```


Installiamo l'editor

Utenti Windows

- Visual Studio (più pesante, ma tutto incluso)



- Visual Studio Code (più leggero, ma con aggiunta del compilatore)



Utenti Mac

- Xcode (più pesante, ma tutto incluso)



- Visual Studio Code (più leggero, ma con aggiunta del compilatore)



Utenti Mac

- Visual Studio Code (più leggero, ma con aggiunta del compilatore)
 1. Installare HomeBrew da Terminale
 2. Successivamente digitare **brew install gcc**
 3. Testare il compilatore



Come testiamo il compilatore?

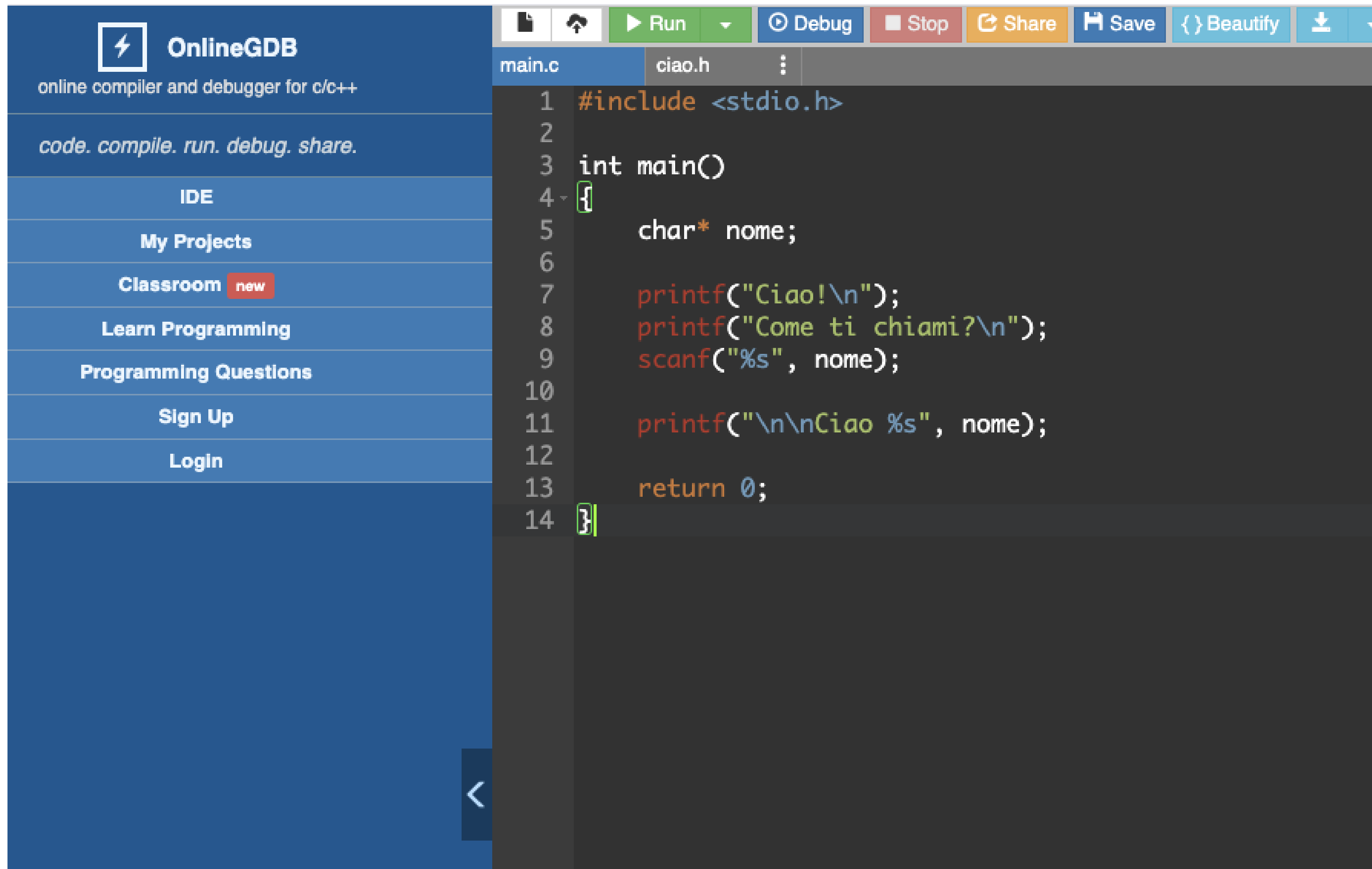
1. Aprire Terminale/Prompt dei comandi
2. Digitare `gcc --version`
3. Creare un file con estensione `.c`
4. Compilare con il comando:
`gcc nome_del_file.c -o Nome_eseguibile`
5. Lanciare da terminale il comando:
`./Nome_eseguibile`

IDE/Compilatori online

- https://www.onlinegdb.com/online_c_compiler
- <https://www.mycompiler.io/it/new/c>
- <https://www.programiz.com/c-programming/online-compiler/>

Primo esercizio

Riferiamo la nostra identità al programma...



The screenshot shows the OnlineGDB web interface. On the left is a sidebar with navigation links: OnlineGDB, code.compile.run.debug.share., IDE, My Projects, Classroom (with a 'new' badge), Learn Programming, Programming Questions, Sign Up, and Login. The top toolbar contains icons for file operations, a 'Run' button, 'Debug', 'Stop', 'Share', 'Save', 'Beautify', and a download icon. Below the toolbar, two tabs are visible: 'main.c' (active) and 'ciao.h'. The main editor area displays the following C code:

```
1 #include <stdio.h>
2
3 int main()
4 {
5     char* nome;
6
7     printf("Ciao!\n");
8     printf("Come ti chiami?\n");
9     scanf("%s", nome);
10
11     printf("\n\nCiao %s", nome);
12
13     return 0;
14 }
```


Tipi di dato

Tipi di dato	Dichiarazione	printf	scanf	Descrizione
int	int n;	%d	%d	Numero intero
float	float f;	%f	%f	Numero reale con virgola
double	double d;	%lf	%lf	Numero reale (più preciso)
char	char c;	%c	%c	Singolo carattere
char [] (stringa)	char stringa [20];	%s	%s	Stringa di caratteri

Secondo esercizio

Dato un numero intero come input, il programma deve restituire prima il suo precedente, poi il suo successivo, infine il doppio del numero inserito.

Priorità operatori in C

- 1** `() [] . ->` → Postfissi → associatività da sinistra a destra
- 2** `++ -- + - ! ~` → Unari / prefissi → associatività da destra a sinistra
- 3** `* / %` → Moltiplicazione, divisione, modulo → associatività da sinistra a destra
- 4** `+ -` → Addizione, sottrazione → associatività da sinistra a destra
- 5** `< <= > >=` → Relazionali → associatività da sinistra a destra
- 6** `== !=` → Uguaglianza → associatività da sinistra a destra
- 7** `&` → AND bit a bit → associatività da sinistra a destra

Priorità operatori in C

- 8** \wedge \rightarrow XOR bit a bit \rightarrow associatività da sinistra a destra
- 9** $|$ \rightarrow OR bit a bit \rightarrow associatività da sinistra a destra
- 10** $\&\&$ \rightarrow AND logico \rightarrow associatività da sinistra a destra
- 11** $||$ \rightarrow OR logico \rightarrow associatività da sinistra a destra
- 12** $?:$ \rightarrow Operatore ternario \rightarrow associatività da destra a sinistra
- 13** $=$ $+=$ $-=$ \dots \rightarrow Assegnamento \rightarrow associatività da destra a sinistra
- 14** $,$ \rightarrow Separatore di espressioni \rightarrow associatività da sinistra a destra

Terzo esercizio

Data la seguente formula:

$$C = \frac{5}{9} (F - 32)$$

Dove:

- C: gradi Celsius;
- F: gradi Fahrenheit

Si scrivi un programma che, preso in input la temperatura in gradi Fahrenheit, faccia la conversione in gradi Celsius

Esempio: 77 gradi Fahrenheit → 25 gradi Celsius