



FAB LAB  
Western-Sicily

# Coding Class da Scratch a Python

> Enrico La Sala  
> FabLab Western Sicily  
> 03 ottobre 2020

# Breve storia del Coding

1842

Ada Lovelace scrive il **primo algoritmo della storia** per la macchina analitica di Babbage. È l'inizio della scienza dell'informazione.

1970

Niklaus Wirth crea il **Pascal**, il primo linguaggio di programmazione adatto alla didattica.

1984

Il C viene ampliato con la teoria della programmazione ad oggetti: nasce il **C++**.

1842

Grazie anche al lavoro di **Alan Turing** negli anni precedenti, nasce la **macchina astratta di Von Neumann**.

1946

1946

1954

Un gruppo di lavoro dell'IBM guidato da John Backus teorizza e successivamente implementa il **FORTRAN**, uno dei capostipiti dei linguaggi di programmazione moderni.

1954

1970

1972

Viene creato il **C** da **Dennis Ritchie**, ancora oggi uno dei linguaggi più potenti e più usati nella programmazione.

1972

1984

# L'Informatica e il pensiero computazionale

- > **2020**, le macchine teorizzate da Babbage e Turing sono diventate reali e hanno rivoluzionato il mondo e l'idea stessa di innovazione!
- > L'esercizio di programmazione fatto da Ada pensando ad una macchina "inesistente" è diventato la professionalità più ricercata sul mercato del lavoro per far funzionare i miliardi di oggetti programmabili esistenti che usiamo ogni giorno.
- > Per essere adeguatamente preparato a lavori del futuro (o meglio del presente!) è indispensabile avere una buona comprensione dei **concetti di base dell'informatica**. Esattamente com'è accaduto nel passato per le altre materie scientifiche.



I benefici del "*pensiero computazionale*": In tutti i lavori moderni ogni giorno si devono affrontare **problemi complessi**; ipotizzare **soluzioni** che prevedono **più fasi** e la **COLLABORAZIONE** con altre persone; formulare una descrizione chiara di cosa fare e quando farlo e quali debbano essere gli **output**.



[scratch.mit.edu](https://scratch.mit.edu)

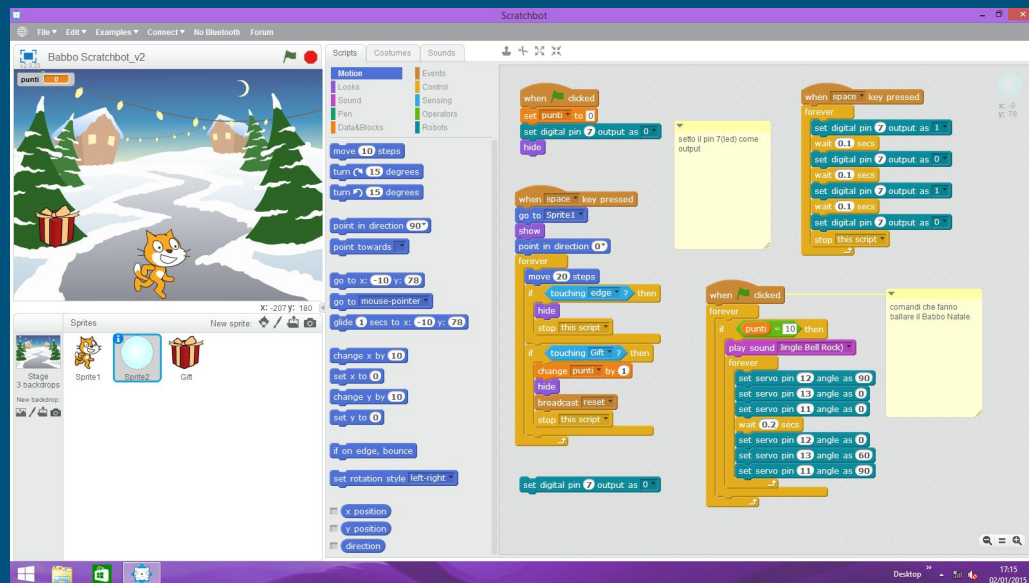
[scratch.mit.edu/projects/editor/](https://scratch.mit.edu/projects/editor/)

Sviluppato dal MIT Media Lab a partire dal 2003, consente di elaborare storie interattive, giochi, animazioni, arte e musica.

Inoltre permette di condividere i progetti con altri utenti del web!

## CHE COS'È?

Linguaggio di programmazione ispirato alla **teoria costruzionista dell'apprendimento** e progettato per l'insegnamento della programmazione tramite primitive visive.





[python.org](https://python.org)

[repl.it/languages/python3](https://repl.it/languages/python3)

Ideato da Guido van Rossum all'inizio degli anni novanta, il nome fu scelto per la passione dello stesso inventore verso i Monty Python.

## CHE COS'È?

Linguaggio di programmazione dinamico **orientato agli oggetti** utilizzabile per molti tipi di sviluppo software.

Offre un forte supporto all'integrazione con altri linguaggi e programmi.

Python supporta la maggioranza dei sistemi operativi (Windows, Mac, Linux, Mobile etc.) e può essere utilizzato nella maggior parte degli ambiti professionali.

Negli ultimi anni è stato il linguaggio di programmazione con il **più alto tasso di adozione**.

# Concetti computazionali

Concetto	Descrizione
SEQUENZA	Una serie di passaggi in un azione
LOOPS	Eeguire la stessa sequenza più volte
PARALLELISMO	Far accadere le cose contemporaneamente
EVENTI	Una causa determina un effetto
CONDIZIONI	Prendere decisioni in base alle condizioni
OPERATORI	Espressioni matematiche e logiche
DATI	Memorizzare, recuperare ed aggiornare valori

# Pratiche computazionali

Pratica	Descrizione
PRODURRE PER ITERAZIONI ED INCREMENTI	Sviluppare una parte per volta, provare e svilupparne ancora
TESTARE E RIMUOVERE GLI ERRORI	Essere sicuri che le cose funzionino, cercare e risolvere gli errori
RICICLARE E MESCOLARE	Creare qualcosa partendo da materiale creato da altri
ASTRARRE E RENDERE MODULARE	Creare qualcosa di grande mettendo insieme una collezione di piccole parti

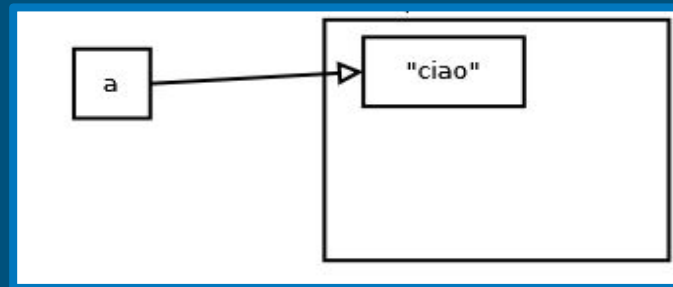
# Alcuni concetti base

“Ciao” è un testo o una sequenza di caratteri.

In informatica una sequenza di caratteri si chiama **STRINGA**

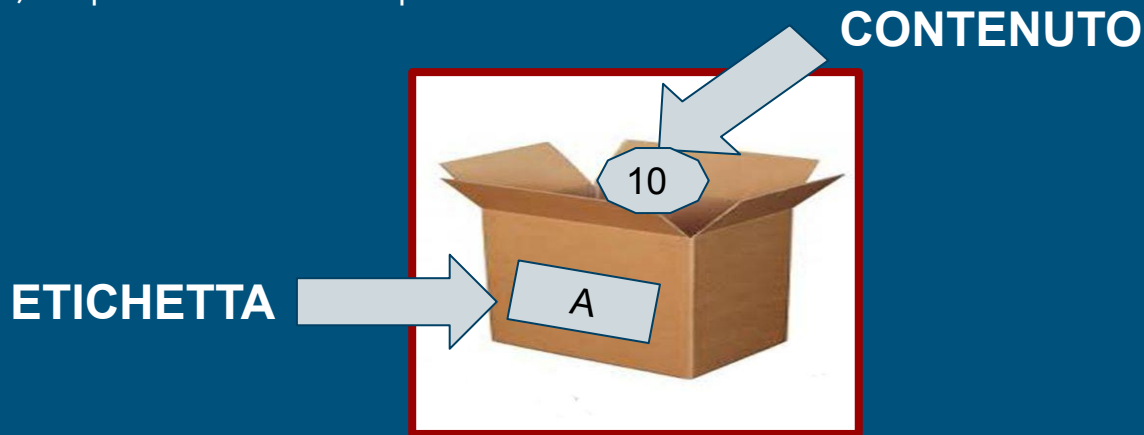
I numeri rappresentano dei **VALORI**

**576**



Una **VARIABLE** è un contenitore (scatola) che ha:

1. Un'etichetta, che è il nome della variabile
2. Un contenuto, che può cambiare nel tempo





# I concetti di ingresso e uscita

## Azioni di ingresso (**INPUT**):

Azioni con cui il programma richiede in ingresso un dato. ( Ex. Python - input() )

Possibilità dell'utente di inserire i dati per risolvere il problema.

“Leggono” da una unità di ingresso e “scrivono” su una variabile, il cui stato viene pertanto alterato.

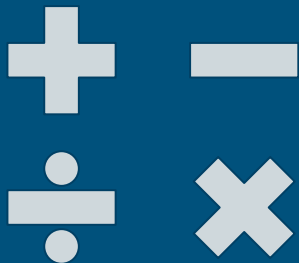
## Azioni di uscita(**output**):

Azioni con cui il programma mostra un dato in uscita. (Ex. Python - print(), return)

Mostrano il valore finale di una variabile o un messaggio, sulla base del comportamento del programma e dei dati di ingresso inseriti.



# Gli operatori

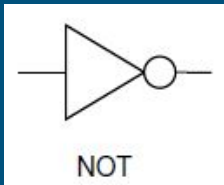
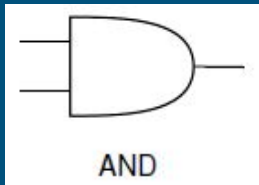
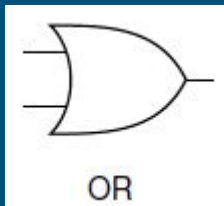


## Operatori Matematici:

“+” , “-” , “x” e “/” , vengono detti operatori matematici

Un' ESPRESSIONE aritmetica, in matematica, è un insieme di numeri legati da operatori

$$66 - 44 + 1$$



## Operatori Logici

*somigliano alle congiunzioni della lingua italiana*

|| **OR**: condizione1 && condizione2, verifica che almeno una delle condizioni sia verificata

&& **AND** : condizione1 && condizione2, verifica che entrambe le condizioni siano verificate

| **NOT**: !condizione, indica l'opposto della condizione

# Le strutture condizionali - if/else

Le strutture di controllo condizionali consentono di specificare che una data istruzione o un dato blocco di istruzioni devono essere eseguiti **"(solo) se"** vale una certa condizione.

L'alternativa **if-then** (*se-allora*) è la più semplice forma di alternativa.

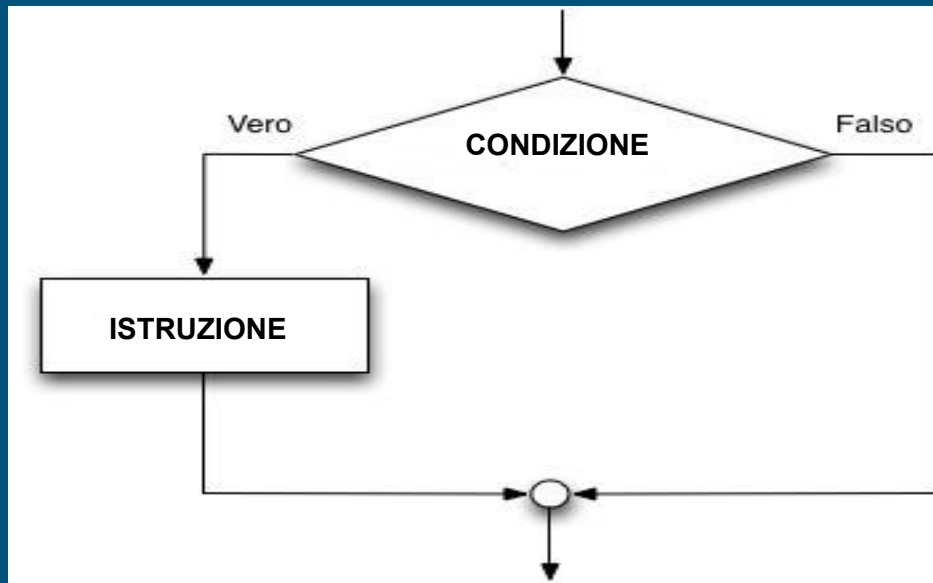
Il suo significato può essere  
parafrasato con la frase

**SE**

vale la condizione C,

**ALLORA**

esegui l'istruzione (blocco) I.



# Le strutture condizionali - if/else

La maggior parte dei linguaggi di programmazione ammette anche (come variante) la forma più articolata **if-then-else** ("**se-allora-altrimenti**"), che si può parafrasare come:

**SE**

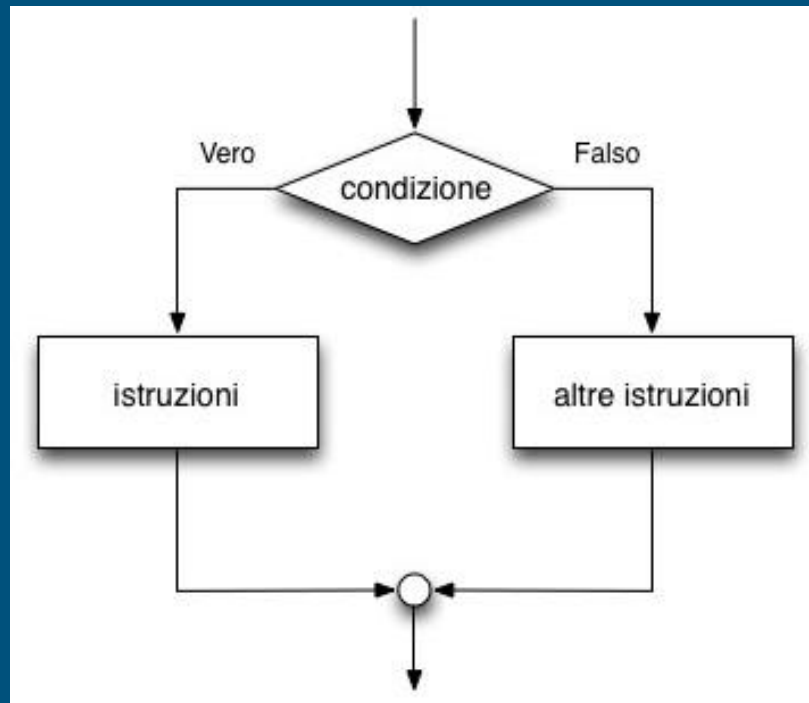
vale la condizione *C*

**ALLORA**

esegui l'istruzione (blocco) *I1*

**ALTRIMENTI**

esegui l'istruzione (blocco) *I2*



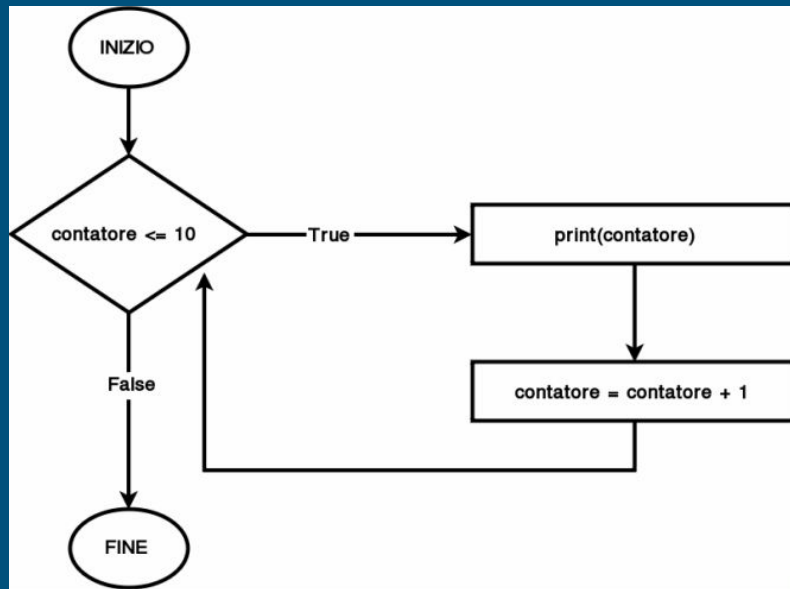
# Le strutture di controllo iterativo - while

Il ciclo while è un flusso di controllo che consente l'esecuzione ripetuta di alcune istruzioni in base a una condizione di entrata (**"finché la condizione C è vera esegui I"**).

Il ciclo si può interrompere attraverso il cambiamento della variabile considerata nella condizione di entrata nelle istruzioni o attraverso il cambiamento di quest'ultima attraverso eventi esterni al ciclo.

**FINCHÈ**  
vale la condizione C

**ALLORA**  
esegui l'istruzione (blocco) I



# Le strutture di controllo iterativo - for

Il ciclo for è una struttura di controllo iterativa che permette l'esecuzione di istruzioni ripetute per un certo numero **noto** di volte.

Il fatto che il numero di ripetizioni sia definito direttamente all'entrata del ciclo lo distingue dal ciclo while.

Una delle possibilità per definire il numero di cicli è la funzione `range(n)`, dove *n* indica il numero di volte in cui si vogliono eseguire le istruzioni del ciclo.

**FINCHÈ**

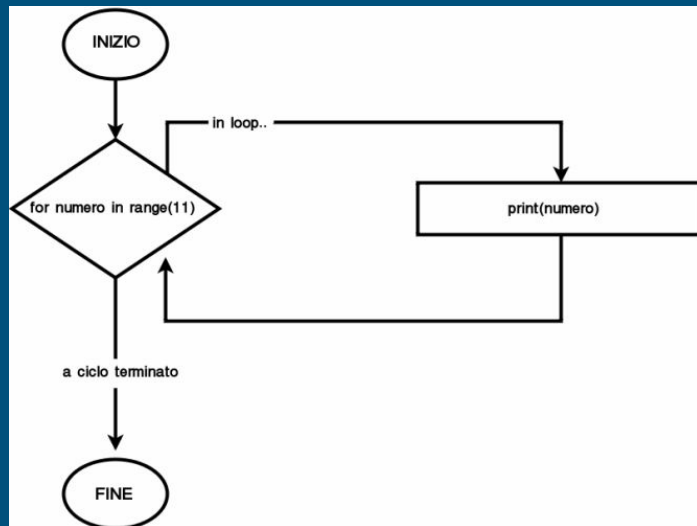
Un numero N è minore di un certo valore

**ALLORA**

esegui l'istruzione (blocco) I

**INCREMENTA N**

$N = N + 1$

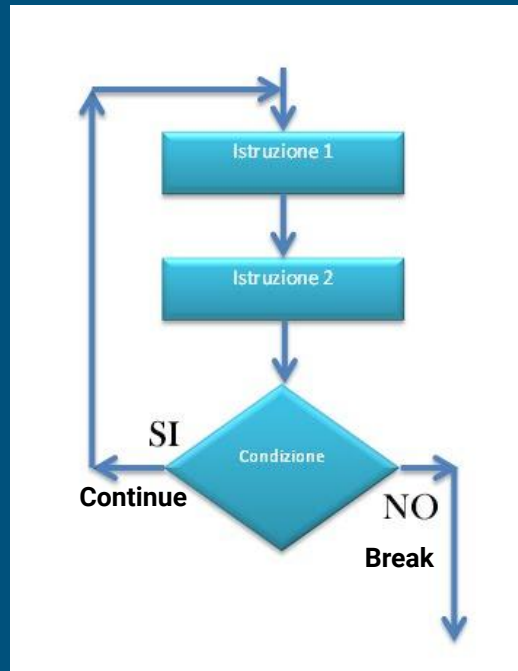


# Le strutture di controllo iterativo - break e continue

Ci sono due espressioni correlate ai cicli while e for: le istruzioni **break** e **continue**.

L'istruzione **break** **INTERROMPE** immediatamente l'esecuzione del ciclo, in questo modo il programma eseguirá l'istruzione successiva al ciclo iterativo.

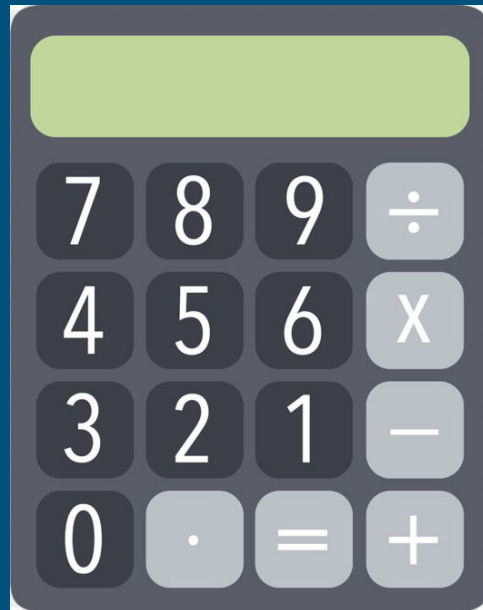
L'istruzione **continue** interrompe l'esecuzione dell'iterazione e **PASSA ALLA SUCCESSIVA** (Incrementando l'iterazione nel caso di for)



# Costruiamo una calcolatrice

Per costruire un programma che simula una calcolatrice utilizzeremo i seguenti strumenti visti oggi:

- Operatori numerici e variabili
- Funzione `input()` e output (`print()`)
- Istruzioni di controllo `if` e `else`
- Funzioni di conversione di tipo (`str()`, `float()`)
- Ciclo `while` e istruzioni `break` e `continue`






# Domande?



**FAB LAB**  
**Western-Sicily**

seguici su 

**FAB LAB**  
Western-Sicily



Via Ludovico Anselmi Correale, 12

91025

Marsala (TP)



+393293638728



info@fablabws.org



www.fablabws.org

