



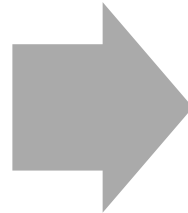
ALESSANDRO FUSER - INFORMATICA

# TIPI DI LINGUAGGIO

Quali sono le tipologie di linguaggi di programmazione che si possono usare?

# COS'È UN LINGUAGGIO DI PROGRAMMAZIONE?

E' un **linguaggio formale** che specifica un insieme di istruzioni che possono essere usate per produrre dati in output



Utilizzabile per il **controllo del comportamento** di una macchina formale o di una implementazione di essa

# CARATTERISTICHE

**Lessico**

**Sintassi**

**Semantica**

**Istruzione**

# CONCETTI PRINCIPALI

Variabile e  
costante

Espressione

Strutture dati

Strutture di  
controllo

Sottoprogramma

Input/output

Commenti

# CODICE SORGENTE

---

**Esprime l'algoritmo del programma tradotto nel linguaggio di programmazione**

---

Contiene le istruzioni da eseguire e (spesso) alcuni dati noti e costanti

---

Attraverso editor di testo

---

IDE → dato editor che è in grado di lavorare a stretto contatto con gli altri strumenti di lavoro

# CODICE ESEGUIBILE

---

**Codice compilato, cioè tradotto in istruzioni di linguaggio macchina da un programma compilatore**

---

File binario eseguibile che non ha bisogno di altri programmi per andare in esecuzione

---

Per i linguaggi di scripting è norma usare un interprete piuttosto che un compilatore

# COMPILAZIONE

## Vantaggio

- Eseguibili velocissimi nella fase di esecuzione adattando vari parametri all'hardware

## Svantaggio

- Necessità di compilare un eseguibile per ogni sistema operativo o piattaforma

# INTERPRETAZIONE

---

Nati per cercare di eliminare il problema della portabilità

---

Per linguaggi che si basano soltanto su librerie compilate ad hoc per ogni piattaforma

---

Il codice viene interpretato e linkato ai vari moduli



# CLASSI DI LINGUAGGI - RICERCA

Imperativi

Strutturati

Orientati  
ad oggetti

Funzionali

Logici

Scripting

# IMPERATIVI

L'istruzione è un comando esplicito che opera su una o più variabili oppure sullo stato interno della macchina

Vengono eseguite in un ordine prestabilito → calcolo per iterazione

Occuparsi di cosa la macchina deve fare per ottenere il risultato che si vuole ottenere → stesura algoritmi

Es. assembly, basic, cobol

# STRUTTURATI

Limitare la complessità  
della struttura del controllo  
dei programmi

Vincolo di usare le strutture  
di controllo definite dal  
Teorema di Bohm-Jacopini

- Sequenza
- Selezione
- Ciclo

Evitare istruzioni di salto  
incondizionato

Es. C, pascal

# ORIENTATI AD OGGETTI

Evoluzione del  
concetto di tipo di  
dato astratto

Caratterizzata da

- Incapsulamento
- Ereditarietà
- Polimorfismo

Es. C++, java,  
php, python

# FUNZIONALI

Basati sul  
concetto  
matematico di  
funzione

Si usa soltanto il  
passaggio dei  
parametri

Ricorsione e  
pattern matching

Struttura dati più  
diffusa è la lista

Es. lisp

# DICHIARATIVI/LOGICI

Istruzione come clausola  
che descrive una relazione  
fra i dati

Descrivere l'insieme delle  
relazioni esistenti fra i dati  
e il risultato evoluto

È compito dell'interprete  
trovare l'ordine giusto di  
esecuzione

Struttura di controllo  
principale = **cut**

- rosso = modifica il comportamento del programma
- Verde = rende più efficiente il calcolo

Tutti gli algoritmi devono  
essere ripensati in termini  
ricorsivi e di backtracking

Risultati eccezionali nel  
manipolare gruppi di enti  
in relazione tra loro

# SCRIPTING

Per automatizzare  
compiti lunghi e  
ripetitivi da eseguire

File contenenti liste di  
comandi o  
registrazioni di  
comandi visuali  
(macro)

# VALUTARE UN LINGUAGGIO

## Espressività

Facilità e  
semplicità con cui  
si può scrivere un  
linguaggio

## Didattica

Rapidità con cui  
lo si può  
imparare

## Leggibilità

Facilità con cui si  
può capire cosa  
fa e come  
funziona

## Robustezza

Capacità di  
prevenire gli  
errori di  
programmazione

## Modularità

Facilita la  
scrittura di  
programmi  
indipendenti che  
comunicano tra  
loro



# VALUTARE UN LINGUAGGIO

## Flessibilità

Possibilità di  
adattare il  
linguaggio  
con la  
definizione di  
nuovi comandi  
e nuovi  
operatori

## Generalità

Facilità di  
codificare  
algoritmi e  
soluzioni di  
problemi in  
campi diversi

## Efficienza

Velocità di  
esecuzione e  
uso delle  
risorse del  
sistema

## Coerenza

Applicazione  
dei principi  
base in modo  
uniforme

# VALUTARE UN LINGUAGGIO

## Diffusione

Numero di  
programmatori  
che usa il  
linguaggio

## Standardizzazione























Pochi dialetti

## Integrabilità

Usare parti di  
codice scritte in  
altri linguaggi

## Portabilità

Portare il  
codice su  
diverse  
piattaforme  
senza doverlo  
modificare

Language Rank	Types	Spectrum Ranking
1. Python	  	100.0
2. C++	  	98.4
3. C	  	98.2
4. Java	  	97.5
5. C#	  	89.8
6. PHP		85.4
7. R		83.3
8. JavaScript	 	82.8
9. Go	 	76.7
10. Assembly		74.5

**Worldwide**, Dec 2018 compared to a year ago:

Rank	Change	Language	Share	Trend
1	↑	Python	25.36 %	+5.2 %
2	↓	Java	21.56 %	-1.1 %
3	↑	Javascript	8.4 %	+0.0 %
4	↑	C#	7.63 %	-0.4 %
5	↓↓	PHP	7.31 %	-1.3 %
6		C/C++	6.4 %	-0.4 %
7		R	4.01 %	-0.3 %
8		Objective-C	3.21 %	-0.9 %
9		Swift	2.69 %	-0.7 %
10		Matlab	2.06 %	-0.3 %

# N-BODY SIMULATION

## PROGRAMMING LANGUAGES PERFORMANCE COMPARISON

