

Informatica

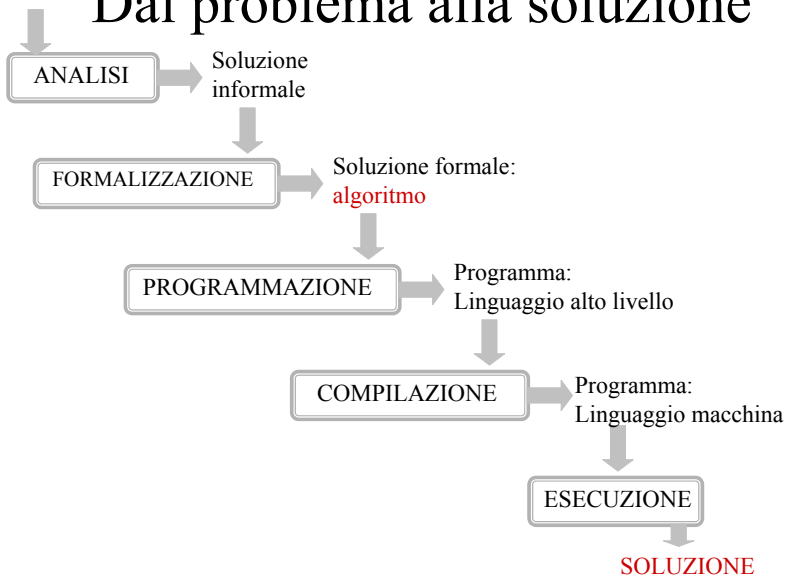


Introduzione alla Programmazione

Mario Fregonara - Informatica

PROBLEMA

Dal problema alla soluzione





Calcolatori e programmazione

- } **Un calcolatore** (o sistema di elaborazione dell'informazione) è un sistema elettronico *programmabile* al fine di svolgere diverse funzioni
- } La caratteristica fondamentale che lo contraddistingue da altri sistemi elettronici è la sua **programmabilità**
- } Equivalentemente, un calcolatore è un sistema in grado di eseguire un processo computazionale (algoritmo) su dei dati in base a delle regole specificate attraverso un programma
- } Il programma è eseguito a partire da un insieme di istruzioni elementari (Instruction Set) che il computer è in grado di comprendere ed eseguire



Calcolatori e programmazione

- } L'insieme delle istruzioni elementari è in generale piuttosto ristretto
- } Combinando istruzioni elementari in lunghe sequenze di istruzioni è possibile far eseguire ad un calcolatore compiti molto complessi in maniera estremamente veloce



Calcolatori e programmazione

- } La **programmazione** è l'attività che consiste nell'organizzare istruzioni elementari, direttamente eseguibili dall'esecutore (il calcolatore), in strutture complesse (programmi) al fine di svolgere determinati compiti elaborativi (algoritmi)
- } Quando introdurremo il concetto di *compilatore* vedremo che non è necessario scrivere programmi direttamente nel linguaggio (estremamente povero) comprensibile dal calcolatore, ma possiamo utilizzare linguaggi ad alto livello, comprensibili da esecutori astratti



Da linguaggio ad alto livello a linguaggio macchina

Linguaggio ad alto livello

```
c = a + b
```

Compilatore

Linguaggio assembly

```
load R1, a  
load R2, b  
add R1, R1, R2  
store R1, c
```

Assembler

Linguaggio macchina

```
0111000101010100  
0001101010001110  
0000100000100000  
0010001000100000
```



Linguaggio assembly

- } E' costituito dalle **istruzioni elementari** viste finora composte da codici mnemonici (es.: load, store, add), registri e locazioni di memoria
- } E' una rappresentazione simbolica (**codici mnemonici**) del linguaggio macchina
- } I codici mnemonici sono associati alle diverse istruzioni macchina che l'hardware è in grado di comprendere
- } E' più comprensibile del linguaggio macchina in quanto utilizza simboli invece che le sequenze di bit



Linguaggi ad alto livello

- } Per superare gli svantaggi della programmazione in linguaggio assembly e in linguaggio macchina, sono stati introdotti i **linguaggi ad alto livello**, in modo da facilitare la programmazione
- } Sono pensati non per essere compresi direttamente da macchine reali, ma da **macchine astratte**, in grado di effettuare operazioni più ad alto livello, rispetto alle operazioni elementari dei processori reali



Linguaggi ad alto livello

- } L'attività di programmazione viene svincolata dalla conoscenza dei dettagli architetturali della macchina utilizzata (ad esempio il numero di registri interni alla CPU)
- } Per poter eseguire su una macchina reale un programma scritto in un linguaggio ad alto livello è necessario tradurlo nel linguaggio della macchina utilizzata
- } Questa operazione di traduzione viene eseguita in modo automatico in due fasi successive



Da linguaggio ad alto livello a linguaggio macchina

Linguaggio ad alto livello

```
c = a + b
```

Compilatore

Linguaggio assembly

```
load R1, a  
load R2, b  
add R1, R1, R2  
store R1, c
```

Assembler

Linguaggio macchina

```
0111000101010100  
0001101010001110  
0000100000100000  
0010001000100000
```



Compilazione

- } Nella prima fase, il programma scritto in un linguaggio ad alto livello viene tradotto nel linguaggio assembly utilizzando appositi programmi detti **compilatori o traduttori**
- } Dopo la fase di compilazione, il programma scritto in linguaggio assembly viene tradotto in linguaggio macchina utilizzando gli **assemblatori (assembler)**



Interpreti

- } Invece di effettuare la traduzione di P, un interprete I legge ogni istruzione contenuta nel programma P, ed effettua immediatamente le operazioni corrispondenti all'istruzione elaborata, utilizzando l'hardware X
- } In pratica, la traduzione dell'intero programma prima dell'esecuzione, viene sostituita dalla **traduzione simultanea, con esecuzione immediata di ciascuna istruzione**



Errori

- } L'assenza di errori in esecuzione non implica che il programma sia corretto: il programma potrebbe produrre risultati diversi da quelli aspettati, cioè svolgere una funzione diversa da quella per cui è stato creato
- } Il **debugger** è uno strumento utile in questa fase per verificare che il programma sia corretto rispetto alle specifiche di progetto. Il debugger permette di osservare passo passo l'andamento dell'esecuzione di un programma