



Insegnamento di Elementi di informatica (6 CFU)

Problemi, Algoritmi e Programmi

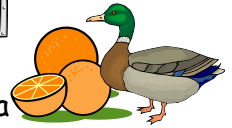
ing. Nadia Ranaldo
Facoltà di Ingegneria
Università degli Studi del Sannio

Problemi



Ogni **problema** è caratterizzato da:

1. Risultato ricercato
 - Entrare in autostrada
 - cucinare l'anatra all'arancia
2. Procedimento per risolvere il problema
 - Procedura di passaggio al casello autostradale
 - una ricetta per l'anatra all'arancia
3. Esecutore, ovvero il soggetto che ha l'effettiva capacità di risolverlo
 - l'automobilista
 - il cuoco



Problem solving



- Uno degli scopi fondamentali dell'informatica è:

la risoluzione di problemi

problema = compito che si vuole far risolvere automaticamente al computer

esecutore

Problem solving



I problemi che siamo interessati a risolvere sono di natura molto varia:

- Trovare il maggiore tra due numeri
- Dato un elenco di nomi e numeri di telefono, trovare il numero di una data persona
- Dati a e b , risolvere l'equazione $ax+b=0$
- Stabilire se una parola precede alfabeticamente un'altra
- Prenotare aerei, treni, hotel, ...
- Ordinare un elenco di nomi
- Ecc.



dati di partenza
(o di ingresso)

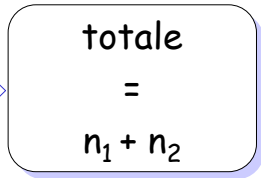
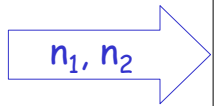


Elaborazione

individuazione dei
risultati

output

Esempio: somma



5



1. Analisi del problema
2. Definizione di un procedimento risolutivo (*algoritmo*)
3. Implementazione dell'algoritmo in un linguaggio di programmazione

6



- Capire bene gli obiettivi del problema, ovvero qual è il risultato che si vuole raggiungere
- Quindi occorre evidenziare
 - le regole da seguire
 - i dati espliciti (indicati in maniera evidente nella traccia del problema) ed impliciti (non indicati nella traccia ma necessari a risolvere il problema)
- Eliminare i dettagli inutili ed ambigui

7



La descrizione rigorosa (formale) delle azioni da compiere per risolvere un problema è detta **ALGORITMO**

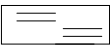

- Un algoritmo è la descrizione della soluzione di un problema espressa come un insieme di regole che, operando sui dati iniziali, consente di ottenere i risultati del problema
- Tali regole vengono determinate tramite la scomposizione del problema di partenza in problemi sempre più semplici (detti *sottoproblemi*), fino ad arrivare a sottoproblemi elementari, ognuno dei quali è detto passo (step) dell'algoritmo

8

Esempi di algoritmi



Procedura per l'ingresso in autostrada

1. **Se** non hai il Telepass vai al passo 2. Altrimenti vai al passo 6.
 2. fermati davanti al casello autostradale
 3. premi il pulsante
 4. prendi e conserva il biglietto 
 5. vai al passo 7.
 6. rallenta nei pressi della stazione
 7. attendere che la sbarra si alzi
 8. ripartire
 9. fine
- La ricetta per l'anatra all'arancia 



9

Caratteristiche di un algoritmo

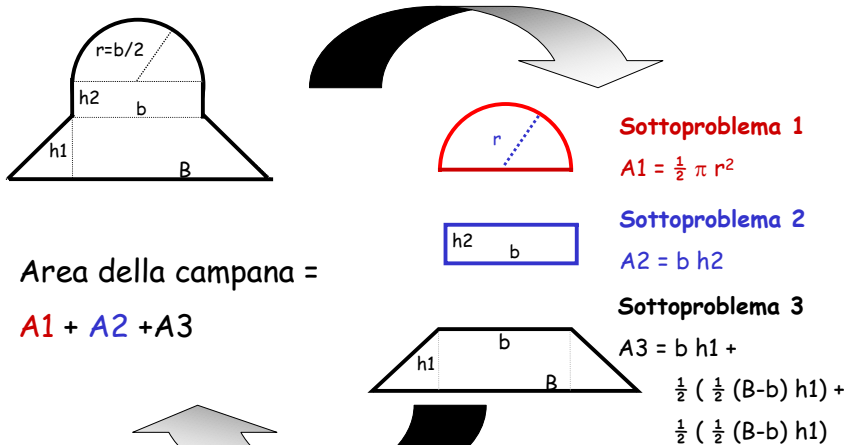


- Lunghezza finita dei passi da svolgere, qualunque siano i dati di ingresso (purchè si tratti ovviamente di un insieme finito)
- Struttura in passi elementari (ovvero che sono direttamente eseguibili dall'esecutore)
- Non casualità (risultato certo e ripetibile, ovvero, con gli stessi dati iniziali, si ottiene sempre lo stesso risultato)
- Non ambiguità
- **Comprendibilità**

Esempio di algoritmo ambiguo:
bagnare i capelli, applicare lo shampoo, sciacquare e ripetere una seconda volta.
Ripetere cosa??

10

Esempio: l'area di una campana



11

Esempio: gestione biblioteca



- Libri disposti sugli scaffali
- La posizione di ogni libro è fissa ed è individuata da due coordinate:
 - numero dello scaffale
 - posizione nello scaffale
- La biblioteca è dotata di uno schedario (ordinato per autore/i e titolo). Ogni scheda contiene, nell'ordine:
 - Cognome e nome dell'autore
 - Titolo del libro
 - Numero dello scaffale
 - Posizione attribuita al libro nello scaffale

12

Esempio di scheda



Autore: *Manzoni*

Titolo: *I promessi Sposi*

Scaffale: *33*

Posizione: *13*

13

Problema



Trovare un libro??



14

Definizione dell'algoritmo



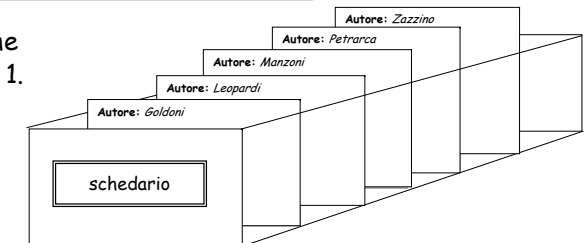
1. Cerca la scheda del libro nello schedario
2. Segnati numero scaffale e posizione indicati sulla scheda
3. Cerca lo scaffale indicato
4. Vai alla posizione indicata e prendi il libro

15

1. Cerca la scheda



Occorre trovare i passi che risolvono il sottoproblema 1.



1. Prendi la prima scheda dello schedario
2. **Se** titolo e autore/i sono quelli cercati, **allora** la ricerca termina con successo, **altrimenti** passa alla scheda successiva
3. **Continua** di scheda in scheda **finché** non trovi quella cercata. **Se** vengono esaurite le schede, **allora** significa che il libro cercato non esiste. Devi cercare il libro in un'altra biblioteca.

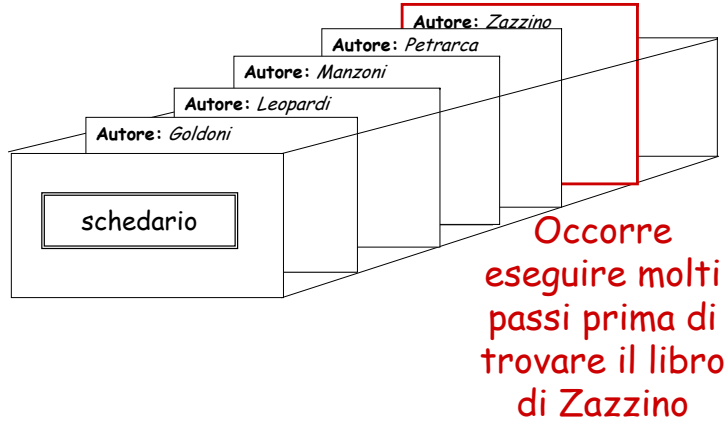
16

1. Cerca la scheda: Esempio



Si fa l'ipotesi che le schede siano in ordine alfabetico rispetto all'autore

Cosa succede se l'autore cercato è Zazzino?



1. Cerca la scheda



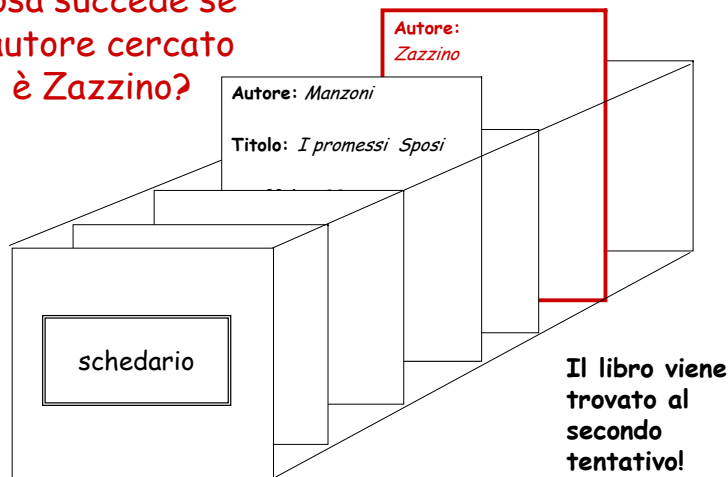
Definizione di un algoritmo che risolve il sottoproblema 1. in maniera più efficiente (veloce)

1. Esamina la scheda centrale dello schedario
2. Se la scheda centrale corrisponde al libro cercato allora la ricerca termina
3. In caso contrario, prosegui allo stesso modo nella metà superiore o inferiore dello schedario a seconda che il libro cercato segua o preceda quello indicato sulla scheda

1. Cerca la scheda: Esempio



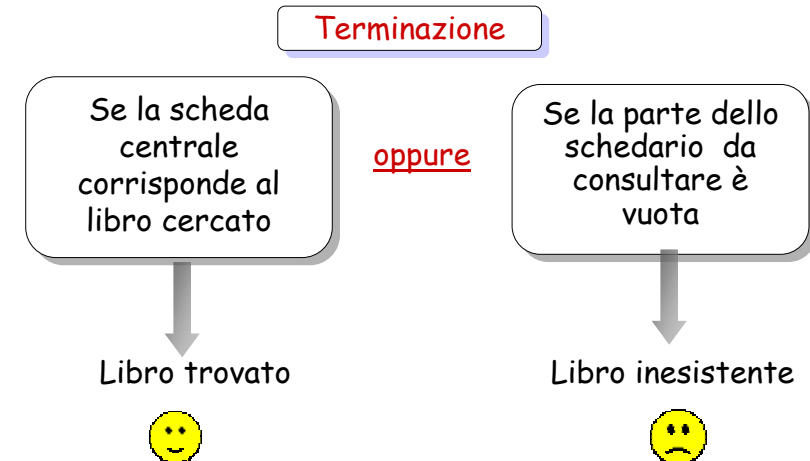
Cosa succede se l'autore cercato è Zazzino?



Revisione del passo 2



L'algoritmo è incompleto: manca la condizione di terminazione quando il libro non esiste



Qualità degli algoritmi



Due qualità fondamentali di un algoritmo sono:

- **Correttezza**
 - l'algoritmo permette effettivamente di risolvere il problema in maniera corretta
- **Efficienza**
 - l'esecuzione dell'algoritmo richiede un uso limitato di risorse
 - un algoritmo è tanto più efficiente quanto meno risorse richiede per la sua esecuzione. Una risorsa importante è il **tempo di esecuzione**
- **Esempio: gestione biblioteca**
 - Entrambi gli algoritmi di ricerca sono corretti
 - Il secondo algoritmo è più efficiente del primo

21

Esempio: algoritmo del risveglio



1. Spegnere la sveglia
2. Alzarsi dal letto
3. Togliersi il pigiama
4. Fare la doccia
5. Vestirsi
6. Fare colazione
7. Prendere il bus per andare a scuola



I passi sono eseguiti in sequenza (ovvero uno dopo l'altro così come sono scritti) e l'ordine dei passi è essenziale per la correttezza dell'algoritmo

22

Controllare il flusso delle istruzioni



Non sempre i passi da eseguire sono in sequenza, ad esempio:

1. Alzarsi dal letto
2. Togliersi il pigiama
3. Fare la doccia
4. Vestirsi
5. Fare colazione
6. **Se piove**
Prendere l'ombrello
7. Prendere il bus

23

Percorsi alternativi



1. Alzarsi dal letto
2. Togliersi il pigiama
3. Fare la doccia
4. Vestirsi
5. Fare colazione
6. **Se piove**
Prendere la macchina
Altrimenti
Prendere il bus

24

Il flusso del ciclo "mentre"



1. Alzarsi dal letto
2. Togliersi il pigiama
3. Fare la doccia
4. Vestirsi
5. Fare colazione
6. **Mentre** piove
Restare a casa
7. Prendere il bus per andare a scuola

25

Le strutture di controllo: CONDIZIONE



- Le istruzioni da eseguire sono determinate dalla valutazione di una certa **condizione**
 - **SE** la scheda centrale corrisponde a quella del libro cercato
 - **ALLORA**
 - Segna la posizione del libro
 - Termina la ricerca
 - **ALTRIMENTI**
 - Scarta la parte superiore o inferiore a seconda che l'autore del libro cercato segue o precede quella centrale

26

Le strutture di controllo: ITERAZIONE



- Le istruzioni vengono eseguite ripetutamente fino a che non si verifica una certa condizione
- Es.
 - **MENTRE** ci sono ancora schede da controllare
 - Individua la scheda centrale
 - **SE** la scheda centrale corrisponde al testo cercato
 - ...

27

... riassumendo



Algoritmo:

- Sequenza di azioni che trasforma i dati iniziali in un numero finito di passi, elementari e non ambigui, per ottenere al risultato finale
- Questa sequenza di azioni può essere eseguita direttamente da un esecutore

Caratteristiche dell'esecutore:

- Tutte le operazioni specificate dall'algoritmo devono poter essere eseguite dall'esecutore (**operazioni elementari**)
- Altrimenti è necessario scomporre il problema in **sottoproblemi** più semplici

28

I diagrammi di flusso



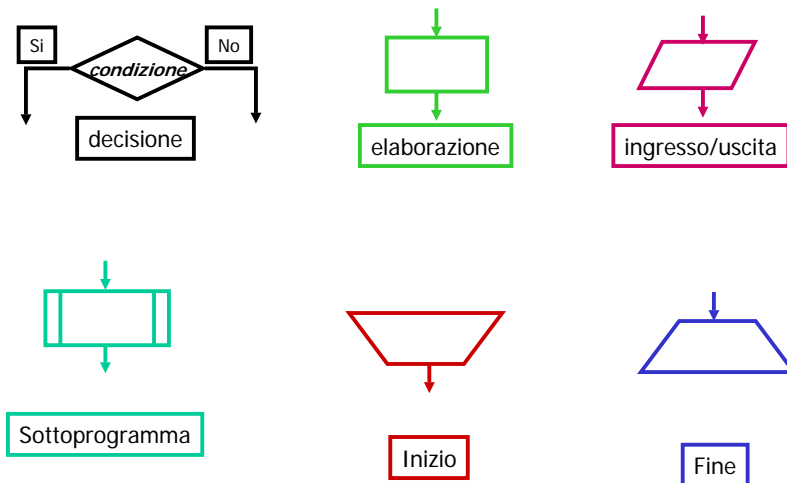
- I diagrammi di flusso sono rappresentazioni grafiche dei passi (elementari o sottoprogrammi) di un algoritmo
- Sono uno strumento efficace per descrivere un algoritmo (più della descrizione a parole, troppo generica o appesantita da troppi dettagli)
- Le operazioni base per un computer sono:
 - Trasferimento di informazioni
lettura dati di ingresso, scrittura risultati, visualizzazione dati intermedi
 - Esecuzione di calcoli (detta elaborazione)
 - Decisione, mediante la valutazione di una condizione che può dare come risposta SI o No (oppure VERO o FALSO)
 - Esecuzione di sottoprogrammi
- Ogni operazione è rappresentata da una forma geometrica

Variabili e costanti



- Una variabile è una quantità che può essere modificata durante l'esecuzione di un algoritmo
 - La possiamo immaginare come il nome di una casella che contiene un dato di un certo tipo (ad esempio numero intero, reale, etc.)
- Una costante è una quantità che non cambia durante l'esecuzione di un algoritmo
 - Esempi:
 - π (il rapporto tra circonferenza e diametro di un cerchio)
 - e (la base dei logaritmi naturali).
- Sia le variabili che le costanti sono caratterizzate da un nome e da un valore

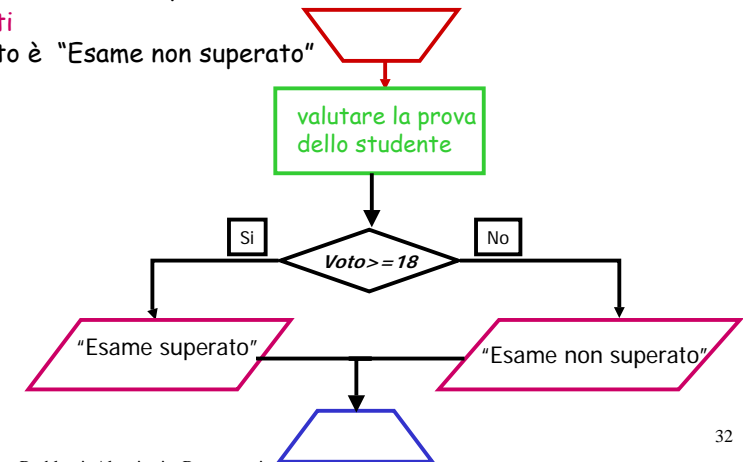
Forme geometriche



Esempio: svolgere un esame



1. Valutare la prova dello studente attribuendo un voto
2. *Se* il voto dello studente è maggiore o uguale a 18 allora il risultato è "Esame superato"
3. *altrimenti* il risultato è "Esame non superato"

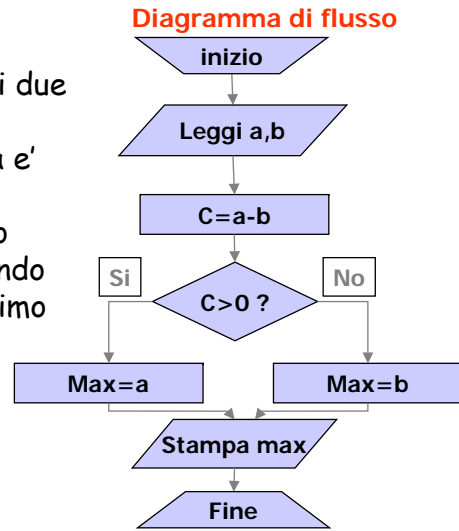


Esempio: massimo di due numeri



Linguaggio naturale

- Leggi due numeri
- Calcola la differenza fra i due numeri letti
- Controlla se la differenza e' maggiore di zero
- Se si il massimo e' il primo
- Se no il massimo è il secondo
- Stampa il valore del massimo



3. Implementazione dell'algoritmo in un linguaggio di programmazione



- Quando l'esecutore è un computer, gli algoritmi vengono detti **programmi**
 - le azioni da eseguire vengono dette **istruzioni**
 - Il risultato viene anche detto **output del programma**
- Il linguaggio formale utilizzato per scrivere un programma per computer viene chiamato **linguaggio di programmazione**
- Dopo aver analizzato il problema e aver definito un algoritmo risolutivo si procede con la **programmazione**, cioè la scrittura del programma in un linguaggio di programmazione

Sintassi e semantica



Linguaggio

Sintassi

insieme di regole che consentono di costruire correttamente le frasi del linguaggio

Semantica

disciplina che studia il significato delle parole e delle frasi

LIVELLO	FORMA CORRETTA	FORMA NON CORRETTA
sintattico	sono andato a scuola	ho andato a scuola
semantico	il gatto è un animale	l'albero è un animale

- Definizione di programma: testo (i.e. sequenza di **istruzioni**) scritto in accordo alla **sintassi e semantica** di un linguaggio di programmazione

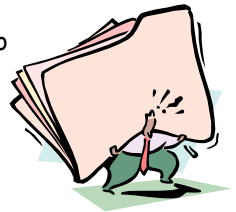
Linguaggio di programmazione ad alto livello



- Un calcolatore è in grado di eseguire direttamente solo programmi scritti nel proprio linguaggio macchina, in cui le istruzioni sono scritte in binario
- I programmatori dei primi computer, dovevano descrivere il programma utilizzando direttamente il linguaggio macchina
 - è molto difficile da comprendere per un uomo!
 - è specifico di un computer: computer di tipo diverso hanno linguaggi macchina differenti
- Per rendere più semplice e più rapido il lavoro dei programmatori sono stati introdotti i **linguaggi di programmazione ad alto livello**, ovvero che sono molto più vicini al linguaggio degli uomini (linguaggio naturale) che a quello del computer

```

01100010001
00101
11010010010
10101
00101001010
00100
11101001001
10101
.....
  
```



Alcuni linguaggi di programmazione "famosi"



- FORTRAN: metà degli anni '50
(**FOR**mula **TRAN**slator)
- COBOL: metà degli anni '50
(**C**ommon **B**usiness-**O**riented Language)
- Pascal: inizio degli anni '70
 - (nome dal matematico francese Blaise Pascal che fu il primo ad ideare una macchina calcolatrice: la Pascalina)
- **C**: inizio degli anni '70
 - suo antenato: linguaggio B
 - è stato sviluppato intorno al 1972, nei Laborator della Bell AT&T americani, da Dennis Ritchie
 - è nato come linguaggio di sviluppo del Sistema Operativo UNIX
- Prolog: inizio degli anni '70
 - (**PRO**gramming in **LOG**ic)

37

Traduzione



- Per rendere un programma (scritto in un linguaggio ad alto livello) eseguibile da un computer è necessario tradurre il programma in un programma **equivalente** scritto nel linguaggio macchina del computer
- La traduzione può avvenire in due modi:
 - compilazione
 - interpretazione

38

Compilazione



- un programma (detto programma **sorgente**) scritto in un linguaggio di programmazione ad alto livello viene trasformato in un programma in linguaggio macchina e **poi può essere eseguito più volte senza dover tradurre nuovamente il programma**



Programma scritto in C

Compilazione



```
0110001000100101
1101001001010101
0010100101000100
1110100100110101
.....
```

controllo della
correttezza sintattica

Nota:
correttezza sintattica

correttezza semantica

39

Interpretazione



- Traduzione riga per riga:
- ciascuna istruzione del programma scritto in un linguaggio di programmazione ad alto livello viene trasformata in istruzioni del linguaggio macchina ed eseguita

40

Compilazione & interpretazione



- Una analogia con la traduzione tra linguaggi diversi
- la compilazione è analoga alla traduzione di un libro
- l'interpretazione è analoga alla traduzione simultanea

- Linguaggi compilati
 - Pascal, C
- Linguaggi interpretati
 - Prolog, Lisp

41

Vantaggi & svantaggi: compilazione



- Vantaggi
 - creazione di programmi eseguibili
 - Elevata velocità d'esecuzione del programma

- Svantaggi
 - correzioni solitamente al termine della compilazione
 - impossibilità di controllare il funzionamento solo di una parte del programma

42

Vantaggi & svantaggi: interpretazione

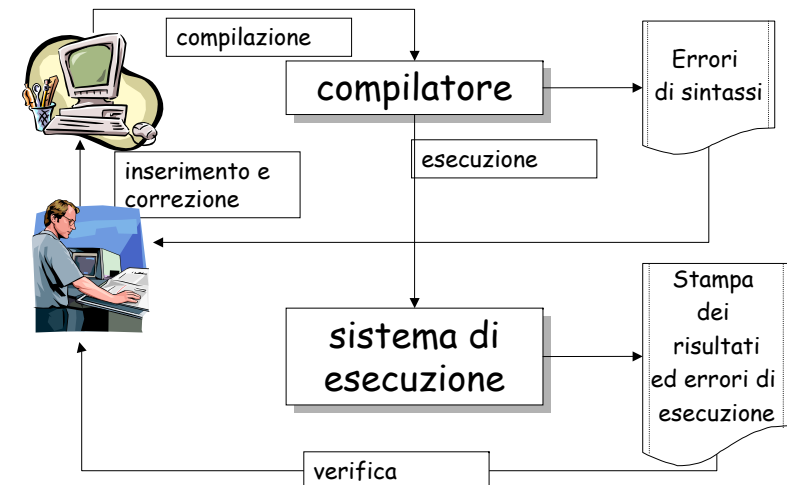


- Vantaggi
 - immediata visione dell'errore (l'interprete controlla immediatamente la sintassi)
 - possono essere eseguite parti di programmi anche non completi

- Svantaggi
 - lenta esecuzione
 - il programma eseguibile dipende dall'interprete perché il codice tradotto non si può memorizzare

43

Attività di programmazione



44

Riepilogo



Problemi  Algoritmi  Programmi

- **Problema:** compito che si vuole far risolvere automaticamente al computer

- **Algoritmo:** descrizione rigorosa delle azioni da compiere per risolvere un problema

- **Programma:** sequenza di istruzioni in un linguaggio di programmazione.

- **Scopo del programma:** fornire al calcolatore le capacità per risolvere un dato problema

