

Corso di: Algoritmi e Strutture Dati 1

Titolare: Dott. Francesco Villante

Orario: Lunedì - 15.00 - 16.30 - Aula 4

Mercoledì - 15.00 - 16.30 - Aula 4

Giovedì - 11.00 - 12.30 - Aula 4

Testi e Riferimenti:

T.H. Cormen, C.E. Leiserson, R.L. Rivest

Introduzione agli Algoritmi - Vol. 1 e 2

Jackson Libri

G. Proietti, Università dell'Aquila

Dispense del corso di Algoritmi e Strutture Dati,

<http://www.di.univaq.it/~proietti/didattica.html>

C. Salati, Università di Ferrara

Dispense del corso di Algoritmi e Strutture Dati,

<http://www.dm.unife.it/~salati>

Introduzione

Algoritmo = Procedimento di calcolo - Descrizione di una sequenza di azioni da eseguire per giungere alla risoluzione di un problema computazionale.

Algoritmo \leftrightarrow **Dato**

Algoritmo \rightarrow Procedimento che, a partire da dati di ingresso (input), produce dati di uscita (output)
 \rightarrow Problema della organizzazione e della **struttura dei dati**.

Gli algoritmi vengono comunemente descritti tramite programmi. Le proprietà generali di un algoritmo sono però fondamentalmente indipendenti dalle caratteristiche di uno specifico linguaggio di programmazione.

Programma \rightarrow **Pseudo-codice**

Descriviamo la sequenza di azioni che l'algoritmo deve compiere in maniera indipendente dal linguaggio di programmazione in cui l'algoritmo è implementato.

Alcune definizioni preliminari

Algoritmo = Qualsiasi procedura computazionale ben definita che trasforma un insieme di dati di **Input** in un insieme di dati di **Output**.

Istanza di un algoritmo = Insieme dei valori che specificano l'input del problema.

Correttezza di un algoritmo = Un algoritmo si dice **corretto** se, **per ogni** istanza di input, esso **termina** con l'output corretto.

Esempio guida: **Problema dell'ordinamento**

Input: Sequenza di n numeri $\langle a_1, a_2, \dots, a_n \rangle$

Output: Permutazione

$$\pi \langle a_1, a_2, \dots, a_n \rangle = \langle a_1', a_2', \dots, a_n' \rangle$$

tale che:

$$a_1' \leq a_2' \leq \dots \leq a_n'$$

Insertion sort

Insertion sort = Algoritmo di ordinamento -
Descrive in maniera formale la procedura con cui
normalmente ordiniamo una mano di carte da gioco.

```
Insertion-Sort(A)
  For j ← 2 to length[A]
    do { k ← A[j]
        i ← j - 1
        while i > 0 e A[i] > k
          do { A[i+1] ← A[i]
              i ← i - 1
            }
        A[i+1] ← k
    }
```

Alcune convenzioni sullo pseudo-codice

- La spaziatura indica la struttura di blocco (istruzioni all' interno dello stesso costrutto hanno la stessa indentatura).
- La istruzione $i \leftarrow e$ indica una operazione di assegnamento.
- Gli elementi di un array sono riferibili specificando il nome dell'array seguito dell'indice tra parentesi quadre.

Insertion sort

Insertion sort = Algoritmo di ordinamento

Insertion-Sort(A)

*A = array che contiene
la sequenza da
ordinare*

For $j \leftarrow 2$ *to* $\text{length}[A]$
do

$\left\{ \begin{array}{l} k \leftarrow A[j] \\ i \leftarrow j - 1 \\ \text{while } i > 0 \text{ e } A[i] > k \\ \quad \text{do } \left\{ \begin{array}{l} A[i+1] \leftarrow A[i] \\ i \leftarrow i - 1 \end{array} \right. \\ A[i+1] \leftarrow k \end{array} \right.$

L' insertion sort descrive in maniera formale la procedura con cui ordiniamo una mano di carte da gioco:

- L' indice j indica la carta corrente. L' elemento $A[j]$ e' copiato nella variabile k ;
- Gli elementi $A[1 \dots j-1]$ rappresentano le carte gia' ordinate;
- Confrontiamo $k = A[j]$ con tutti gli elementi $A[1 \dots j-1]$ partendo da $j-1$;
- Gli elementi piu' grandi di k vengono uno ad uno spostati "a destra" fino a che non troviamo la giusta posizione della carta corrente.

Insertion sort

Insertion sort = Algoritmo di ordinamento

Insertion-Sort(A)

For $j \leftarrow 2$ to $\text{length}[A]$

do

$k \leftarrow A[j]$

$i \leftarrow j - 1$

while $i > 0$ e $A[i] > k$

do

$A[i+1] \leftarrow A[i]$

$i \leftarrow i - 1$

$A[i+1] \leftarrow k$

A = array che contiene
la sequenza da
ordinare

Esempio

input: $A = \langle 5, 2, 4, 6, 1, 3 \rangle$

$j = 2$ 5 2 4 6 1 3 \rightarrow 2 5 4 6 1 3

$j = 3$ 2 5 4 6 1 3 \rightarrow 2 4 5 6 1 3

$j = 4$ 2 4 5 6 1 3 \rightarrow 2 4 5 6 1 3

$j = 5$ 2 4 5 6 1 3 \rightarrow 1 2 4 5 6 3

$j = 6$ 1 2 4 5 6 3 \rightarrow 1 2 3 4 5 6

Notare: L'algoritmo di insertion sort e' corretto!
Per ogni istanza di input fornisce l'output corretto.

Esercizi

- Illustrare la operazione di Insertion-Sort applicata all'array $A = \langle 31, 41, 59, 26, 41, 58 \rangle$
- Riscrivere la procedura di Insertion-Sort per ordinare in modo non crescente