Dispense del corso di Fondamenti di Informatica - Scuola Superiore per Mediatori Linguistici “Adriano Macagno” di Cuneo – Sede di Lamezia – A.A. 2022-2023

Programma del corso

Programma di teoria

Introduzione

Cos’è l’informatica
Problemi, algoritmi, programmi e calcolatore
Applicazioni dell’informatica

Rappresentazione dell’informazione

Rappresentazione di numeri naturali
Cenni di aritmetica binaria
Rappresentazione di informazione non numerica (caratteri, immagini, ecc.)
Calcolo proposizionale

Architettura del calcolatore

Processore, memoria centrale, memoria di massa, memoria cache, periferiche
Una tassonomia dei sistemi informatici

Reti di calcolatori

La comunicazione dei dati
le reti locali
le reti geografiche
Ipertesti, Multimedia, Ipermedia
Internet ed i suoi servizi

Elementi di Programmazione

Algoritmi, Linguaggi e Programmi
Variabili e Tipi
Diagrammi di Flusso
Costrutti Base di un Linguaggio di Programmazione: il Visual Basic
Assegnazione
Istruzioni Condizionali
Istruzioni Iterative
Esempi e Semplici Applicazioni

Programma di laboratorio

Elaborazione di testi: Microsoft Word
Fogli elettronici: Microsoft Excel
Elementi di programmazione in Visual Basic

Testi consigliati

- Sciuto, Bonanno, Fornaciari, Mari. **Introduzione ai Sistemi Informatici**. McGraw-Hill, 1997.
- Curtin, Foley, Sen, Morris. **Informatica di Base**. McGraw-Hill, 1999.
- Suardi. **ECDL Advanced Office Modulo AM4, Foglio elettronico**, collana ECDL Apogeo, 2003
- Materiale didattico reperibile sul sito del docente.

Prova d'esame

L'esame consiste in due prove: una teorica e una pratica.
La **prova teorica** consiste (generalmente) nello svolgimento di 10 quesiti di cui alcuni a risposta multipla sui temi trattati durante il corso (e.g., conversioni, nozioni hardware), alcuni di programmazione di semplici algoritmi, altri di risoluzione di semplici problemi.
La **prova pratica** consiste, invece, nello svolgimento al calcolatore di alcuni esercizi relativi alla formattazione di un testo in Word, alla realizzazione di un foglio avanzato di calcolo Excel con visualizzazione grafica di dati e alla stesura di semplici codici in Visual Basic.

Introduzione

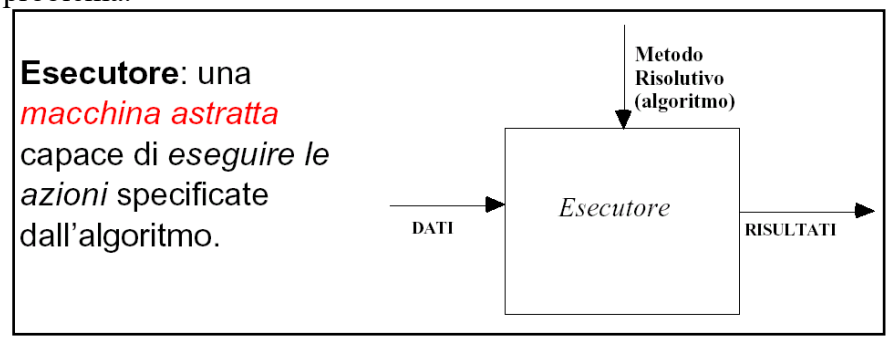
L'Informatica è definita come la Scienza della Rappresentazione e dell'Elaborazione dell'informazione o, in altri termini, lo studio delle strutture dati che rappresentano le informazioni e degli algoritmi che le elaborano.

Dati e informazioni

Una prima precisazione va fatta riguardo i termini *dato* e *informazione* legata al dato. Benché spesso utilizzati come sinonimi, una sequenza di dati non è necessariamente fonte d'informazione. Il seguente elenco di numeri non può essere, infatti, immediatamente associato ad alcuna informazione:
18, 20, 21, 24, 23, 25, 28, 18, 20, 21, 24, 25, 27, 28, 18, 21, 21, 24, 24, 25, 29, 18, 20, 20, 24, 23, 25, 30.
L'elenco potrebbe rappresentare le temperature registrate a Rende nel mese di luglio così come i voti d'esame di altrettanti studenti. I dati danno luogo a informazione nel momento in cui sono collocati in un contesto ben preciso. Ad esempio, se è noto che i numeri specificano delle temperature, allora la sequenza rappresenta l'informazione del loro andamento nell'arco di circa quattro settimane. In altri termini *se al dato è possibile associare un significato ben preciso esso diventa informazione, altrimenti rimane un mero dato*.

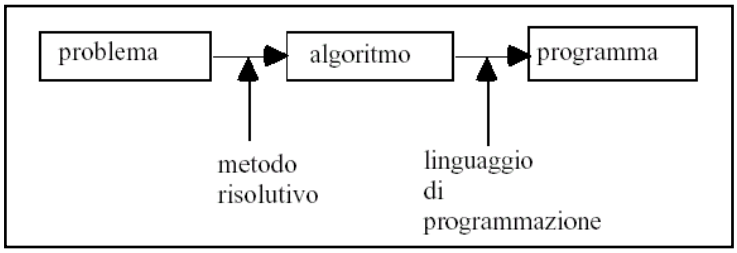
Rappresentazione ed elaborazione dei dati

L'Informatica, secondo la definizione appena data, si occupa della rappresentazione e dell'elaborazione dei dati (o delle informazioni, se si vogliono usare i due termini come sinonimi). I dati devono ovviamente essere rappresentati in maniera conveniente. Una rappresentazione conveniente della precedente sequenza di numeri potrebbe ad esempio essere quella basata su numeri naturali. Tale rappresentazione non sarebbe conveniente invece se i numeri rappresentassero anche valori negativi (per esempio se le temperature fossero relative al mese di gennaio). In tal caso potrebbe essere adottata una rappresentazione basata su numeri interi o, meglio ancora su numeri reali (qualora le temperature fossero del tipo 23.1 o 18.7).
Una volta convenientemente rappresentati, i dati possono essere elaborati. Un esempio di elaborazione potrebbe essere quello di determinare la temperatura media. A tal fine i dati vengono *processati* attraverso un algoritmo al fine di ottenere un risultato: si sommano i valori relativi alle temperature e si divide per il loro numero.
Un *algoritmo* può essere informalmente definito come una *sequenza finita di "mosse" che risolve in un tempo finito un determinato problema*. L'esecuzione delle azioni nell'ordine specificato dall'algoritmo consente di ottenere, a partire dai dati di ingresso, i risultati che rappresentano la risoluzione del problema.



Benché la definizione di algoritmo sia indipendente dall'esecutore, esso è spesso associato al calcolatore elettronico. Un algoritmo deve essere pertanto definito in modo che possa essere interpretato ed eseguito correttamente dalla macchina. A tal proposito esistono differenti *linguaggi*

di programmazione che consentono di definire le istruzioni dell'algoritmo per la loro esecuzione su macchine calcolatrici.



I linguaggi di programmazione possono essere suddivisi in linguaggi a basso e ad alto livello. Un esempio di linguaggio a basso livello è il *linguaggio macchina*, che fornisce appunto le istruzioni a basso livello (cioè direttamente eseguibili dall'elaboratore) per la risoluzione di un problema. In linguaggio macchine è per esempio possibile specificare il caricamento di dati (variabili) in locazioni di memoria ben precise (ad esempio i registri della CPU). I linguaggi ad alto livello, come il *C/C++, Java*, *Pascal*, *Basic* e altri, sono molto più simili al linguaggio naturale. Le istruzioni sono quindi molto più intuitive per il programmatore. In ogni caso, l'esecuzione di un programma scritto in un linguaggio ad alto livello è subordinata a una fase in cui le istruzioni del linguaggio sono tradotte in istruzioni a basso livello (compilazione), direttamente eseguibili dal calcolatore.
Di seguito è riportato un semplice programma in Basic per la somma di due numeri interi:

```
Sub SOMMA( )
Dim A, B as Integer
A = InputBox("Immetti un numero")
B = InputBox("Immetti un secondo numero")
Print "Somma:"; A+B
End Sub
```

Le parole in grassetto sono istruzioni del linguaggio. In questa fase è sufficiente sapere che l'istruzione **Sub** specifica il "corpo del programma", che termina con l'istruzione **End Sub**. L'istruzione **Dim** consente di definire due variabili come intere (istruzione **as Integer**). Il loro valore è letto come input tramite il comando InputBox (che non è un'istruzione base del linguaggio, ma è ottenuta come combinazione di istruzioni base) e la loro somma, calcolata tramite l'operatore "+", è stampata su video attraverso l'istruzione **Print**.

Un esempio di algoritmo leggermente più complesso del precedente è il calcolo della potenza *n*-esima di un numero intero *a*. Una possibile soluzione, espressa in linguaggio naturale, è la seguente:

```
Utilizziamo le variabili  N, Ris
Inizialmente Ris=1  e   N=n
Algoritmo:
Fino a che N>0
Calcola Ris*a e memorizzalo in Ris
Decrementa N
```

Il precedente algoritmo può essere espresso in un linguaggio di programmazione, ad esempio il Pascal:

```
Program potenza;
Integer Ris, N, A;
Read(N); Read(A);
Ris=1;
While (N>0) do
Ris=Ris*A;
```

N=N-1;
Print(Ris);

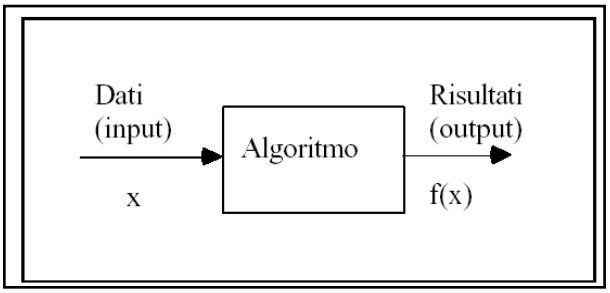
Il precedente programma va tradotto in linguaggio macchina comprensibile all'elaboratore (fase di compilazione del programma). Quando le istruzioni vengono eseguite, il programma prende i dati in ingresso (valori iniziali di N e A) attraverso la tastiera (input) e poi stampa il risultato sul video (valore finale di Ris) (output).

Le *proprietà fondamentali* di un algoritmo sono le seguenti:

1. **Eseguibilità**: ogni azione deve essere *eseguibile* da parte dell'esecutore dell'algoritmo in un tempo finito.
2. **Non-ambiguità**: ogni azione deve essere u*nivocamente interpretabile* dall'esecutore
3. **Finitezza**: il numero totale di azioni da eseguire, per ogni insieme di dati di ingresso, deve essere finito.

Due algoritmi si dicono **equivalenti** quando:

1. hanno lo stesso dominio di ingresso;
2. hanno lo stesso dominio di uscita;
3. in corrispondenza degli stessi valori nel dominio di ingresso *producono gli stessi valori* nel dominio di uscita.



Due algoritmi equivalenti forniscono lo stesso risultato, ma possono avere diversa efficienza e possono essere profondamente diversi. Un esempio di due algoritmi equivalenti ma con diversa efficienza è illustrato di seguito nel caso della moltiplicazione di due numeri interi.

Algoritmo 1	Algoritmo 2 (somma e shift)
Somme successive: 12x12 = 12+12+…+12=144	12x 12= 24 12= 144

Rappresentazione di algoritmi tramite diagrammi di flusso (flow chart)

La rappresentazione di un algoritmo può essere espressa graficamente tramite diagrammi di flusso (flow chart). È uno metodi più comuni usati per la rappresentazione di algoritmi, specialmente nel caso di algoritmi brevi. Un diagramma di flusso, detto anche diagramma a blocchi, si presenta come un insieme di figure geometriche collegate da frecce.
Tutti i diagrammi a blocchi cominciano con un'ellisse che contiene la parola inizio:

I dati in ingresso sono i dati noti del problema, quelli che devono essere elaborati per arrivare alla soluzione:

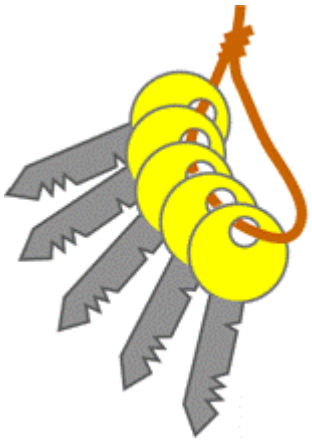
Le operazioni da svolgere sui dati sono racchiuse in rettangoli:

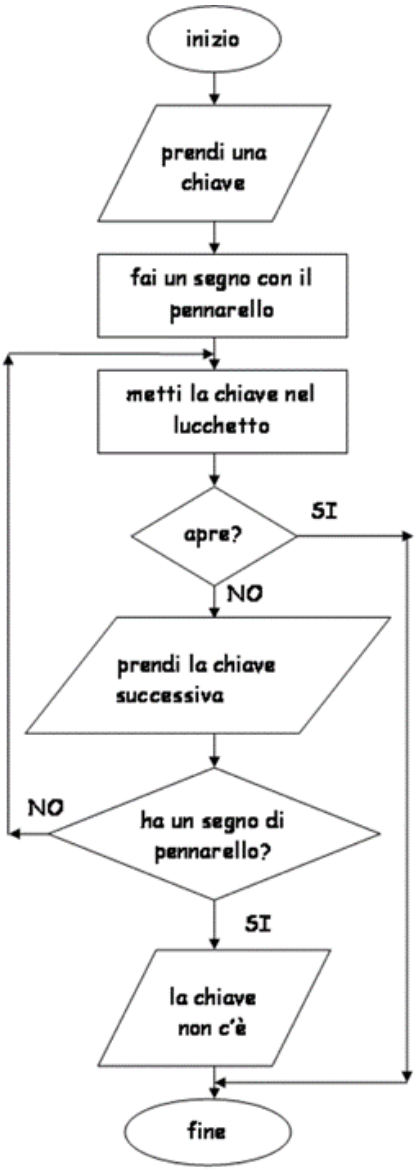
Quando si deve fare una scelta tra due possibilità si usa il rombo:

I dati in uscita sono quelli che si vuole conoscere e costituiscono il risultato dell'elaborazione:

Ogni diagramma di flusso si conclude con un'ellisse che contiene la parola fine:

Un esempio di flow chart relativo al problema di trovare in un mazzo di chiavi quella che apre un lucchetto è il seguente:





Cosa possono e non possono fare i calcolatori elettronici

La realizzazione di programmi, spesso complessi, consente una varietà di possibili applicazioni, tra cui: Word Processing (Memorizzare, elaborare testi), Basi di Dati (Memorizzare grossi archivi di dati, recupero veloce, produrre informazioni globali), Accesso Remoto (Trasmissione e recupero di informazioni), Calcolo (Risolvere problemi matematici), Simulazioni (Rappresentare e elaborare informazioni che simulano l'ambiente reale), Progettazione ingegneristica, Rappresentazione scientifica dei dati (Figura 1).
Tuttavia, esistono problemi che non possono essere risolti tramite un calcolatore elettronico per diversi motivi, tra cui: la risoluzione del problema non esiste, la risoluzione del problema impiegherebbe un tempo di calcolo eccessivo (anche infinito), la soluzione del problema è "soggettiva". Un esempio di problema non risolubile è dire se dato un programma e un input esso terminerà il suo calcolo (problema meglio noto come il problema della fermata della macchina di Turing). Un esempio di problema la cui risoluzione richiederebbe un tempo infinito è dire se data una finzione $f : N \rightarrow N$, $f(x)$ è costante per ogni valore di x. Infine, un esempio di problema la cui soluzione sia soggettiva può essere il seguente: dato un insieme di immagini di paesaggi, determinare quello più rilassante.

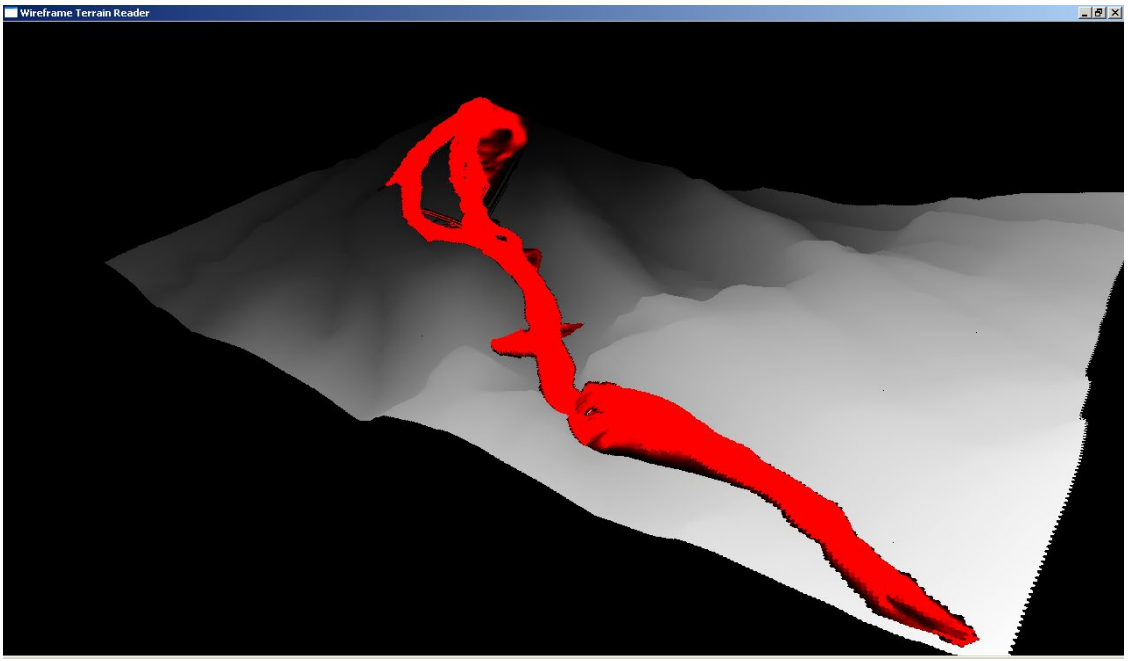


Figura 1 Esempio di visualizzazione scientifica 3D di una colata di detrito nell'area di Sarno (Campania).