SCHEDA 3.01

Un **algoritmo** è un procedimento che consente di risolvere un problema attraverso una sequenza (finita) di istruzioni precise e non ambigue.

Ecco due ricette di cucina. Utilizzando la definizione qui sopra, possiamo dire che queste ricette sono algoritmi?

FRULLATO	INSALATA
• tagliare la frutta in pezzi • inserire la frutta in un frullatore • aggiungere latte • azionare il frullatore per 5 minuti • versare il tutto in un bicchiere	• lavare delle foglie d'insalata • aggiungere olio q.b. • aggiungere un aceto q.b. • aggiungere sale q.b.

La risposta è no: le due ricette **NON** sono algoritmi!
Per quale motivo?

La ricetta del frullato manca di precisione:

- tagliare la frutta in pezzi

  In quanti pezzi devo tagliare la frutta? Quanto devono essere grossi?

- aggiungere latte

  Quale dev'essere la quantità di latte? Un cucchiaio, un l bicchiere?

La ricetta dell'insalata manca di non ambiguità:

- aggiungere olio q.b.
- aggiungere aceto q.b.
- aggiungere sale q.b.

  Cosa vuol dire "quanto basta"? Come faccio a sapere quando è abbastanza?

SCHEDA 3.03

Un computer saprebbe leggere, interpretare e utilizzare le precedenti ricette?

Un computer saprebbe leggere, interpretare e utilizzare le precedenti ricette?

No, perché al computer manca quel "qualcosa in più" che permette a un essere umano di comprendere le istruzioni anche quando esse non sono chiare e precise: la deduzione.

Se in una ricetta leggiamo quantità come "quanto basta" o "un cucchiaino", deduciamo quanto di un ingrediente dobbiamo più o meno aggiungere. Un computer, invece, non è in grado di farlo. Servono misure precise!

Quindi il computer come potrebbe capire queste ricette?

Dovranno essere tradotte in un **codice** appropriato, scritto in un'apposita "lingua" che il computer può comprendere e che prende il nome di **linguaggio di programmazione**.

Logo, Python, Java sono tutti nomi di linguaggi di programmazione finalizzati a portare a termine compiti di natura diversa.

Cos'è un bug?

Durante la stesura del codice il programmatore potrebbe inavvertitamente sbagliare a scrivere un'istruzione. Quando il programma verrà eseguito e si arriverà all'istruzione errata si potranno verificare due diversi scenari:

1. Il programma non funzionerà affatto.

   (es. stiamo preparando un'insalata, ma la ricetta chiede di usare strumenti ed ingredienti che non abbiamo a disposizione nella nostra cucina , quindi è impossibile proseguire nella realizzazione dell'insalata!)

2. Il programma funzionerà, ma non nel modo previsto.

   (es.vogliamo prepare un risotto , ma dopo aver seguito tutta la ricetta ci ritroviamo davanti un piatto di insalata di riso c'è qualcosa che non va!)

Il mal funzionamento del programma che porta a risultati errati o diversi da quelli attesi, prende il nome di **bug** ed è dovuto ad un errore. In gergo informatico il programma viene detto **buggato**.

Dopo aver scritto un programma è importantissimo controllare che non ci siano errori, attraverso un processo che prende il nome di **debugging** e che serve a trovare (e correggere) tutti gli errori presenti nel codice!

Quanti tipi di bug esistono?

I bug si suddividono in tantissime categorie, vediamone alcune tra le più importanti:

- Bug di sintassi

  Quando il programmatore non rispetta le regole del linguaggio di programmazione che usa e scrive il programma inserendo errori di scrittura, cosa succederà all'esecuzione?Il programma non funzionerà! Quando il programmatore fa eseguire al computer la stessa istruzione un numero sbagliato di volte, oppure inserisce un blocco sbagliato nell'algoritmo, cosa succederà all'esecuzione?Il programma funzionerà, ma restituirà risultati errati!

- Bug logici

  Sono errori che derivano dagli errori di progettazione dell'algoritmo risolutivo, quindi sono degli errori che si commettono prima ancora della scrittura del programma nel linguaggio di programmazione.

SCHEDA 3.07

Da dove deriva il termine “bug”?

È una parola inglese che vuol dire “insetto”.

Il comune utilizzo del termine in ambito informatico risale al 9 settembre 1947, quando Grace Hopper, all’epoca tenente della marina statunitense, archiviò un caso molto particolare di malfunzionamento del famoso computer Harvard Mark II: una falena incastrata tra i circuiti! All’epoca i computer erano grossi come stanze intere, quindi che ci finisse dentro un insetto era un evento comune! Il tenente Hopper inserì la falena all’interno del registro del computer annotandola come “primo caso effettivo di insetto”.

Da qui il termine “bug” è divenuto di uso comune per indicare un qualsiasi malfunzionamento di un computer.

SCHEDA 3.08

Ecco una foto del bug del tenente Hopper, il registro è oggi conservato allo Smithsonian National Museum of American History, a Washington:

