

In questo file avete un insieme di definizioni (che vi abbiamo dato nel corso dell'attività) che mi sono permessa di raccogliere e scrivere per voi allo scopo di farvele imparare bene.

Gli appunti presi a lezione a volte sono incompleti o con imprecisioni e, non avendo un testo di riferimento, questi appunti saranno il vostro testo. **PERSONALIZZATELI!**

DEFINIZIONE rigorosa di algoritmo:

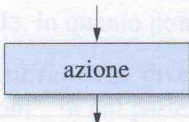

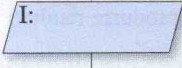
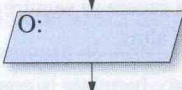



Un **algoritmo** è un insieme finito di azioni che risolvono un determinato problema, trasformando i dati di input in dati di output (o risultati) attraverso le relazioni esistenti tra gli input e gli output.

### Le proprietà degli algoritmi

Un algoritmo deve essere costituito in modo da rispettare alcune proprietà; in particolare, occorre che sia:

- **generale**, cioè deve essere in grado di risolvere non un solo problema ma una classe di problemi dello stesso tipo;
- **finito**, cioè deve avere un inizio e una fine e non deve contenere istruzioni che si ripetono all'infinito;
- **non ambiguo**, cioè deve contenere solamente istruzioni con interpretazione univoca e non contraddittoria.

Uno **schema di flusso** (o **diagramma a blocchi** o **flow chart**) è una rappresentazione grafica di un algoritmo realizzata mediante l'utilizzo di simboli, la cui forma dipende dal tipo di azione che si vuole descrivere, uniti da frecce che rappresentano il flusso dell'esecuzione delle istruzioni che compongono l'algoritmo.

Istruzione	Simbolo	Significato
Azione		<i>Blocco di Azione:</i> eseguo l'azione descritta all'interno del rettangolo
Controllo (Condizionale)		<i>Blocco di Controllo:</i> verifico la condizione e se il risultato è vero passo a eseguire le istruzioni sul ramo corrispondente a <i>vero</i> , altrimenti passo a eseguire le istruzioni sul ramo corrispondente a <i>falso</i>
Comunicazione (Trasmissione)	di Ingresso 	<i>Blocco di Input dati:</i> chiedo in ingresso all'utente un valore che verrà scritto in una variabile in memoria
	di Uscita 	<i>Blocco di Output dati:</i> fornisco in uscita all'utente un valore che verrà visualizzato a video
Salto	È rappresentato da una freccia che si innesta nel punto dell'algoritmo cui si deve saltare 	<i>Blocco di Salto condizionato o incondizionato:</i> vado a eseguire l'istruzione indirizzata dal flusso delle frecce
Inizio algoritmo		<i>Blocco di Inizio algoritmo</i>
Fine algoritmo		<i>Blocco di Fine algoritmo</i>

COSA si utilizza quando si scrive un algoritmo:

Le **variabili** e le **costanti** sono gli oggetti utilizzati dalle istruzioni del programma. Esse risiedono nella memoria dell'elaboratore e corrispondono a contenitori dei valori che vengono man mano elaborati durante l'esecuzione del programma.

Alle variabili e alle costanti è associato un **nome**, detto **identificatore**, che le individua univocamente all'interno del programma. Alla variabile è associato un **valore** che *può cambiare* durante l'esecuzione del programma, mentre alla costante è associato un **valore** che *non cambia* per tutta l'esecuzione.

Operazioni con variabili e costanti:

L'operazione tipica che viene eseguita su una variabile è detta **assegnazione** e corrisponde *all'inserimento* di un valore nel contenitore associato all'identificatore della variabile, *sostituendo* il valore precedentemente contenuto. Solitamente viene indicata da una freccia che va dall'operazione (a destra) verso il nome (identificatore) della variabile (a sinistra). Gli effetti dell'assegnazione in tabella

Assegnazione	Risultato	Commento
$A \leftarrow 9$	Prima: A <input type="text"/> Dopo: A <input type="text" value="9"/>	Nel contenitore associato all'identificatore della variabile $A$ viene inserito il valore 9. Nel caso in cui il contenitore associato ad $A$ contenga un precedente valore, questo viene perso e sostituito dal nuovo valore.
$B \leftarrow A$	Prima: A <input type="text" value="9"/> B <input type="text"/> Dopo: A <input type="text" value="9"/> B <input type="text" value="9"/>	Nel contenitore identificato da $B$ e corrispondente alla variabile $B$ viene inserito il valore contenuto nel contenitore associato alla variabile $A$ . In questo caso anche $B$ assume il valore 9. Il contenitore associato ad $A$ , in questo caso, non perde il suo valore di partenza.
$A \leftarrow A+1$	Prima: A <input type="text" value="9"/> Dopo: A <input type="text" value="10"/>	Viene calcolato il valore dell'espressione a destra della freccia, leggendo il contenuto del contenitore identificato da $A$ e incrementandolo di un'unità, per poi inserire il risultato ottenuto nuovamente nel contenitore della variabile $A$ . La variabile $A$ perde il vecchio valore, che è sostituito con quello nuovo.
$A \leftarrow A+B$	Prima: A <input type="text" value="10"/> B <input type="text" value="9"/> Dopo: A <input type="text" value="19"/> B <input type="text" value="9"/>	Il contenuto della variabile $A$ è aggiunto al contenuto della variabile $B$ e il risultato dell'espressione è inserito nel contenitore della variabile $A$ (a sinistra della freccia). La variabile $A$ perde il vecchio valore che viene sostituito con quello nuovo, mentre la variabile $B$ mantiene il suo valore inalterato.
$AUS \leftarrow A$ $A \leftarrow B$ $B \leftarrow AUS$	Prima: A <input type="text" value="19"/> B <input type="text" value="9"/> Dopo: A <input type="text" value="9"/> B <input type="text" value="19"/> AUS <input type="text" value="19"/>	Nello scambio di $A$ con $B$ occorre utilizzare una variabile ausiliaria, per poter scambiare il valore contenuto nelle variabili $A$ e $B$ . In questo modo non si perde nessun valore.

## Il concetto di tipo di dato

Come abbiamo detto, alle variabili è associato un **valore** che è memorizzato in un contenitore in memoria, univocamente individuato da un **identificatore**. Il valore di una variabile può *cambiare* durante l'esecuzione.

Si definisce **tipo della variabile** l'intervallo di valori che una variabile può assumere, l'insieme delle operazioni che su di essa possono essere svolte, la quantità di memoria che essa occupa e il tipo di rappresentazione in memoria del valore in essa contenuto.

I principali tipi di dati sono:

- **Numerico**: i valori che la variabile può assumere appartengono all'insieme dei numeri interi relativi e all'insieme dei numeri reali; gli operatori applicabili sono quelli aritmetici usuali (+, -, \*, /), quelli relazionali (=, <, >, ≤, ≥, ≠) più altri operatori che dipendono dal linguaggio di programmazione utilizzato;
- **Carattere**: i valori che la variabile può assumere appartengono all'insieme delle lettere dell'alfabeto, minuscole (dalla a sino alla z) e maiuscole (dalla A sino alla Z), delle cifre numeriche (da 0 a 9) e dei simboli speciali (' , ' , %, =, ...). Più caratteri in sequenza costituiscono un tipo di dati chiamato **stringa**. Sui caratteri si possono applicare gli operatori relazionali usuali più altri operatori, che dipendono dal linguaggio di programmazione che si sta utilizzando;
- **Logico o booleano**: i valori sono quelli appartenenti alle coppie (vero, falso), (true, false), (si, no), (1,0), come definiti nell'algebra booleana. Gli operatori applicabili sono gli operatori booleani, di cui parleremo più avanti.

*Il tipo di dato determina la quantità di memoria occupata dalla variabile e il tipo di rappresentazione in memoria del valore in essa contenuto.*

Ogni linguaggio di programmazione presenta diversi tipi di dati e i principali fra essi fanno riferimento alle tipologie appena descritte.

A ogni distinto tipo è associato:

- un nome del tipo;
- un intervallo di validità;
- una occupazione in memoria;
- una rappresentazione in memoria della variabile;
- un insieme di operatori applicabili a quel tipo.

## ESEMPIO DI TIPO

Prendiamo ad esempio Il tipo int del c++

- un nome del tipo: int (16 bit)
- un intervallo di validità: da -32768 a 32767
- una occupazione in memoria: 2 byte
- una rappresentazione in memoria della variabile: in virgola mobile (caratteri in codice ASCII)
- un insieme di operatori applicabili a quel tipo: / (=div), % (=mod), +, -, \*, ^ (=elevamento a potenza)