Pseudoalgoritmi
un'attività per riflettere sul concetto di algoritmo*

Violetta Lonati

Abstract

In questo documento si presenta l'attività chiamata "Pseudoalgoritmi". L'attività è adatta a studenti della scuola secondaria di secondo grado, studenti universitari, oppure adulti. Lo scopo è quello di dare agli studenti l'occasione di riflettere sul concetto di "algoritmo" in maniera operativa, partendo dall'analisi di una serie di procedure descritte in linguaggio naturale. L'attività può essere proposta a studenti che abbiano già lavorato su questo tema, per rivedere il concetto e verificare se sia stato compreso in profondità, oppure può essere usata proprio per introdurre il concetto per la prima volta. Il documento descrive brevemente l'attività e si sofferma sull'analisi del concetto di algoritmo e delle sue proprietà, facendo continuo riferimento agli pseudo-algoritmi da cui la riflessione prende avvio.

1 Descrizione dell'attività

Nella prima fase gli studenti, divisi a coppie, ricevono un foglio con la lista degli pseudo-algoritmi (disponibile come allegato in fondo a questo documento) e, discutendo tra loro, devono stabilire per ciascun pseudo-algoritmo se possa essere definito effettivamente algoritmo oppure no, giustificando le loro decisioni.

Nella seconda fase verranno formati gruppi unendo le coppie a due a due, e gli studenti dovranno confrontare le loro decisioni con quelle dei compagni, eventualmente rivedendole per arrivare ad una nuova decisione condivisa.

Infine ogni gruppo deve formulare una propria definizione di algoritmo e specificare quali proprietà un algoritmo deve avere per potersi definire tale. Ogni gruppo presenterà la sua definizione al resto della classe; il conduttore metterà man mano in evidenza le somiglianze e le differenze tra le varie definizioni.

L'attività si conclude con la presentazione, da parte del conduttore, di una definizione "consolidata" del concetto di algoritmo, e la sua messa a confronto con le definizioni dei gruppi.

Varianti Si può accorciare l'attività iniziando il lavoro già in gruppi di 4. In caso di classi molto numerose la seconda fase si può ripetere formando gruppi di 8 persone.

Materiali di lavoro Ciascuno studente ha a disposizione un foglio con le procedure da discutere (vedi allegato). Ciascun gruppo ha a disposizione un cartelloni e dei pennarelli o, alternativamente, un computer per produrre un "cartellone virtuale".

2 Analisi delle procedure

Gli "pseudo-algoritmi" sono stati ideati e formulati per favorire la discussione e suscitare dubbi, poiché tutti presentano delle criticità che sono rilevanti nella definizione di algoritmo. Esaminiamoli uno alla volta.

Auto-insegnamento Non è chiaro come si debba eseguire il passo 2 ("cercare di risolverlo"); non è specificato cosa fare se nel testo non c'è la soluzione del problema. Inoltre l'esecuzione potrebbe non terminare mai, se non si riesce a risolvere il problema (si avrebbe un *loop*, o ciclo infinito).

Le lasagne al forno Alcuni passi (per esempio: preparare il ragù o la besciamella) non sono elementari: servirebbero ulteriori istruzioni su come portare a termine quei compiti, a meno di dare per scontato che l'esecutore abbia già delle competenze specifiche (ad esempio, sia un cuoco). Inoltre l'ultimo passo è vago, poiché non si dice quanto deve durare la cottura e quale deve essere la temperatura del forno. Infine, non è necessario fare il ragù prima della besciamella, si potrebbe anche fare al contrario senza modificare il risultato.

*Ultimo agiornamento April 26, 2020

Per accedere a un PC del laboratorio Non sono considerati tutti i casi possibili, in particolare la procedura funziona solo se il PC è già acceso prima di cominciare. Inoltre non è specificato cosa fare se non ci sono tutor in laboratorio o se il sistema manifesta comportamenti diversi da quello indicato (cioè rispondere con la frase "utente non abilitato").

Quanto è lunga la lista della spesa? Questa procedura non si presenta come una *sequenza di istruzioni*, a cui siamo abituati a pensare quando si parla di algoritmi; gli studenti tendono a classificare questo pseudo-algoritmo piuttosto come una definizione, o un insieme di regole.

Ricetta del cocktail Cosmopolitan Possono valere considerazioni analoghe a quelle fatte per le lasagne: per quanto bisogna shakerare? Quanto "lunga" deve essere la coppetta? Inoltre gli ingredienti potrebbero essere messi nello shaker in qualsiasi altro ordine.

Somma in colonna di due numeri Il risultato è corretto solo per alcune coppie di numeri, e più precisamente quelle formate da numeri entrambi di 2 cifre e la cui somma sia inferiore a 100, ovvero ancora di due cifre. Non c'è nessuna indicazione relativa al fatto di *ripetere* la somma anche sulle colonne delle centinaia ecc. La prima istruzione non è precisa.

Piegatura di un foglio di carta in formato A4 Il passo 1 non è preciso, poiché non si specifica se la piegatura deve avvenire in orizzontale o in verticale. Inoltre è fisicamente impossibile ripiegare 16 volte su se stesso un foglio di carta, per quanto sottile.

Operazioni matematiche La procedura non sembra utile, poiché non fa altro che stampare sempre 100. Le istruzioni sono precise ma non hanno una coerenza logica interna: il risultato delle operazioni intermedie non viene usato nella parte finale della procedura quindi di fatto le operazioni intermedie risultano inutili. Inoltre la procedura produce sempre lo stesso effetto (ovvero la stampa di 100) indipendentemente dai numeri letti all'inizio, pertanto la procedura non svolge alcuna operazione matematica a dispetto di quanto affermato nel titolo della procedura.

A fronte di questa analisi, quali di questi pseudo-algoritmi si possono definire davvero algoritmi e quali no? In effetti quasi mai si può dare una risposta univoca, in particolare perché sono espressi usando il linguaggio naturale, che nella sua essenza è poco preciso e presenta delle ambiguità; inoltre l'interpretazione delle istruzioni può variare a seconda di chi le esegue, che qui non viene indicato (né si indica quali siano le sue capacità, si pensi al caso delle lasagne).

A ben riflettere, bisogna dire che entrambe queste criticità hanno a che fare più con la difficoltà di esprimere o formulare un algoritmo a parole, che non con l'algoritmo in sé (parliamo qui, in un certo senso, della stessa relazione che c'è tra un'idea e il modo in cui si riesce a comunicarla ad un'altra persona, ad esempio in forma scritta). Per superare tali criticità in informatica si usano a volte dei formalismi (come i *diagrammi di flusso* o lo *pseudo-codice*) che riducono l'ambiguità intrinseca nel linguaggio naturale. Tuttavia, il superamento vero dei questi problemi espressivi si ha soltanto quando l'interpretazione e l'esecuzione dell'algoritmo avviene ad opera di un *automa* e quando l'algoritmo viene espresso in un *linguaggi di programmazione*, diventando così un *programma*: in tale contesto, infatti, ogni "parola" ha un significato preciso ed esplicito al quale l'esecutore automatico (chiamato anche *interprete*) si attiene rigorosamente.

In un certo senso, è proprio questa esplicitazione dell'interprete, e delle sue capacità, ciò che rende possibile descrivere esattamente un algoritmo; altrimenti proprietà quali chiarezza o precisione restano sempre, necessariamente, fumose. D'altro canto, i linguaggi di programmazione sono fatti proprio per essere compresi dalle macchine e spesso sono di difficile "leggibilità" per noi; per questo motivo, tra individui (anche se programmatori), spesso risulta comunque conveniente, o necessario, descrivere un algoritmo *a parole*, nonostante l'inevitabile perdita di precisione che questo comporta.

3 Verso una definizione di algoritmo

Ecco alcuni esempi di definizioni prodotte dagli studenti:

- Serie di istruzioni, precise e non ambigue, per portare a termine un certo compito a partire da condizioni iniziali date.

- Sequenza chiara, ordinata, logica e completa di istruzioni che consentano di risolvere un problema oppure che permettano di svolgere un'operazione in modo univoco.
- Sequenza di istruzioni finite, elementari, ben definite e non ambigue utilizzate per svolgere un compito che diano un risultato determinato.
- Sequenza di istruzioni ben definite volta alla risoluzione di un problema in tempo finito.
- Sequenza ordinata di operazioni elementari e non ambigue tramite le quali è possibile risolvere una classe di problemi.

Nella definizione o nelle proprietà fondamentali enunciate dagli studenti ritroviamo di solito queste componenti:

- vi sono istruzioni, comandi, operazioni, passi, passaggi; a volte si specifica che questi passaggi sono elementari, semplici, di base, ripetibili, fattibili;
- tali istruzioni fanno parte di una sequenza, serie, lista o, più raramente, insieme;
- l'algoritmo risolve un problema o svolge un compito; in ogni caso deve avere uno scopo, servire a qualcosa;
- ricorrono aggettivi quali: preciso, chiaro, non ambiguo, inequivocabile, univoco; prevedibile, ripetibile; logico, coerente, diretto, semplice; ordinato, sequenziale; completo, universale; utile.

Nella sua accezione più tipica, è infatti appropriato definire un **algoritmo** come una "sequenza finita di passi che serve a risolvere uno specifico tipo di problemi o a svolgere uno specifico tipo di compito". Si noti che la parola "sequenza" implica di per sé che i passi siano ordinati. È interessante notare, inoltre, che nella definizione proposta non si parla di un problema ma di uno "specifico tipo di problemi": questo si riferisce al fatto che l'algoritmo, in genere, non serve a risolvere un solo problema particolare, ma tanti problemi particolari dello stesso tipo; l'algoritmo cioè fornisce un metodo che si può applicare a tanti casi diversi ma simili. Ad esempio, l'algoritmo per la somma in colonna non si applica ad una particolare coppia di numeri (ad esempio 12 e 42), ma si può applicare a tante coppie diverse di numeri interi.

Anche con queste precisazioni, la definizione resta tuttavia un po' vaga; in particolare, cosa si intende per "passo"? Chiariremo questo aspetto nella prossimo paragrafo; prima però vogliamo sottolineare come la definizione faccia implicitamente riferimento al paradigma della programmazione imperativa (al quale siamo esposti più facilmente), mentre non contempla altri paradigmi come quello della programmazione logica o funzionale, e modelli di calcolo più teorici come quello della macchina di Turing, in cui gli algoritmi sono definiti mediante insiemi finiti di *regole di transizione*. [1]

4 Definizione di algoritmo e proprietà fondamentali

Una definizione più completa e precisa è la seguente, fornita da Donald Knuth (vedi allegato, in inglese): un algoritmo è un insieme *finito* di regole che determinano una *sequenza* di passi per risolvere uno specifico *tipo di problemi* o per svolgere uno specifico tipo di compito; per essere tale, secondo Knuth, un algoritmo deve inoltre soddisfare alcune proprietà fondamentali, che traduciamo e commentiamo qui di seguito.

Finiteness - finitezza Un algoritmo deve terminare dopo un numero finito di passi. Secondo Knuth, una procedura che non termina non andrebbe chiamato algoritmo, ma *metodo computazionale*. [2]

Definiteness - precisione. Ogni passo di un algoritmo deve essere definito precisamente; le azioni da eseguire devono essere specificate rigorosamente e in maniera non ambigua per ogni caso possibile. Come già discusso prima, la possibilità di specificare in maniera precisa le azioni da compiere si realizza grazie all'uso dei linguaggi di programmazione, nei quali ogni "frase" ha un significato univoco e inequivocabile.

[1] Non è questo il contesto per approfondire queste differenze. Basti sapere che, nel paradigma della programmazione *imperativa*, un programma viene inteso come una serie di istruzioni (dette anche comandi, o direttive), ciascuna delle quali può essere pensata come un "ordine" che viene impartito alla macchina virtuale del linguaggio di programmazione utilizzato. Esempi noti di linguaggi di programmazione imperativi sono i classici C, Basic, Fortran, o i moderni linguaggi visuali Sratch o Blockly. Anche molti linguaggi *ad oggetti*, come ad esempio Java, PHP, Javascript, condividono comunque lo stile imperativo.

[2] Questa distinzione suona forse un po' datata, poiché molti degli "algoritmi" reali che sono alla base delle applicazioni che usiamo costantemente sono progettati per non terminare, si pensi ai server web o ai servizi online sempre operativi. Tuttavia, la distinzione resta importante da un punto di vista teorico, quando si studia ad esempio la *complessità* degli algoritmi.

Input e output - dati in ingresso e in uscita. Un algoritmo ha zero o più input, ovvero dati che sono messi a disposizione prima che l'algoritmo inizi, o durante la sua esecuzione. Questi input sono presi da un insieme specificato di oggetti. Un algoritmo ha uno o più output, ovvero risultati che stanno in una relazione specificata con l'input. Questo significa che, dati certi input, l'algoritmo produce sempre gli stessi output, come specificato dalla relazione suddetta. Quando l'algoritmo non ha nessun dato in ingresso, l'output prodotto è sempre lo stesso. La relazione tra input e output, in questo caso, ha questa forma: "l'output è ... per qualunque input".

La possibilità di fornire input diversi ad un dato algoritmo è ciò che consente all'algoritmo, come discusso in precedenza, di poter risolvere non un solo problema particolare, ma tanti problemi particolari dello stesso tipo (si pensi di nuovo al caso della somma in colonna, che vogliamo poter applicare ad ogni coppia di numeri e non solo, ad esempio, al caso 42 + 17).

Effectiveness - fattibilità. Ci si aspetta che un algoritmo sia effettivamente eseguibile, nel senso che tutte le sue operazioni devono essere sufficientemente elementari da poter essere svolte in principio in maniera esatta, in un periodo di tempo finito, da qualcuno che usi carta e penna. Anche qui, l'uso dei linguaggi di programmazione e dunque la definizione formale dell'interprete (o della macchina che esegue l'algoritmo) aiuta a precisare cosa si intende con *operazione elementare*.

Aggiungiamo alcune ulteriori considerazioni finali. La proprietà di finitezza riguarda la possibilità di eseguire un algoritmo, e non la sua descrizione; la finitezza della descrizione è pure richiesta, ma viene già espressa nella definizione di algoritmo ove si specifica che l'insieme di regole è, appunto, *finito*. Anche nella definizione semplificata di algoritmo come sequenza di passi, è fondamentale specificare che tale sequenza deve essere finita, in modo che l'algoritmo si possa scrivere nella sua interezza. Ad esempio, non si può elencare un insieme infinito di casi, o usare espressioni come "e così via" o "eccetera". Questo tuttavia non esclude che l'algoritmo si possa potenzialmente applicare ad un insieme infinito di situazioni.

Infine, sottolineiamo di nuovo che l'uso della parola *sequenza* nella definizione comporta che i passi vadano svolti in un ordine ben preciso, stabilito dall'algoritmo stesso; in particolare si noti che non si potrebbe sostituire la parola "sequenza" con la parola "insieme"!

5 Altre proprietà per gli algoritmi

Nelle loro discussioni, gli studenti si trovano spesso a criticare caratteristiche degli pseudo-algoritmi che però non hanno a che fare con le proprietà elencate nella definizione data sopra.

Ad esempio, l'ultimo pseudo-algoritmo, quello delle operazioni matematiche, spesso non viene legittimato come algoritmo perché "non fa nessuna operazione matematica", "non ha senso", "non è logico", "è stupido". Tuttavia, considerando la definizione sopra, non vi è motivo per dire che non si tratti di un algoritmo: tutte le proprietà sono soddisfatte! D'altro canto è senz'altro vero che questo algoritmo ha dei difetti rilevanti, in particolare:

1. non svolge il compito dichiarato dal titolo (fare operazioni matematiche);
2. produce il suo risultato (stampare 100) dopo aver svolto delle azioni totalmente scorrelate dal risultato stesso.

Il primo difetto ha a che fare con la (mancanza di) *correttezza* dell'algoritmo. Gli algoritmi si usano quando svolgono il compito che ci interessa che svolgano; in questo caso, se sono interessato a fare le operazioni matematiche, questo algoritmo non mi serve, perché produce altro. Tecnicamente, si dice che l'algoritmo non è corretto per questo compito. Si noti che l'algoritmo per definizione svolge un compito, specificato proprio dall'algoritmo stesso; il fatto che questo compito non ci sia utile, o non corrisponda esattamente ad una nostra esigenza, non toglie che esso sia comunque un algoritmo.

La seconda obiezione riguarda invece il tema dell'*efficienza* dell'algoritmo. Infatti, anche cambiando il titolo dello pseudo-algoritmo in "Stampa 100" per descrivere in maniera corretta la sua *semantica*, è chiaro che questo non svolga il suo compito in maniera efficiente, dal momento che svolge delle operazioni totalmente inutili al raggiungimento dell'obiettivo. Anche questa dell'efficienza è senz'altro una proprietà desiderabile per un algoritmo (un intero ambito dell'informatica, quello della *complessità computazionale* si occupa di questo) ma, di nuovo, il fatto che un algoritmo non sia efficiente non toglie che esso si possa comunque considerare un algoritmo.

Un'altra proprietà rilevante per un algoritmo, ma –di nuovo– non necessaria per definirlo tale, è la sua *generalità*, ovvero la sua capacità di risolvere un'ampia classe di problemi. Abbiamo visto che in alcuni casi, qualora non ci siano input, un algoritmo diventa molto specifico, nel senso che si può applicare

soltanto a una sitauzione particolare. Al contrario, l'utilità di un algoritmo è spesso legata al fatto che si possa applicare ad una buona varietà di casi.

Altre proprietà che potrebbero interessarci in un algoritmo, ma che non sono necessarie per considerarlo tale, possono riguardare ad esempio il modo in cui è formulato (si parla di *leggibilità* dei programmi), o come è organizzato (la *modularità* e la *manutenibilità* dei programmi sono aspetti molto studiati nell'ambito della *ingegneria del software*).

Dovrebbe essere a questo punto chiaro che, dato uno specifico tipo di problema, non esiste un solo algoritmo corretto per risolverlo. I diversi algoritmi però possono essere confrontati in base a diversi criteri, come l'efficienza o la manutenibilità. In particolare il fatto che certi passi possano essere fatti in maniera diversa o in altro ordine (come nel caso del cocktail), non è critico per stabilire se si tratti di un algoritmo o meno.

6 La definizione applicata agli pseudo-algoritmi

In questa parte del documento, riprendiamo in considerazione uno a uno gli pseudo-algoritmi e vediamo se soddisfano la definizione e le proprietà presentate in precedenza, oppure perché non le soddisfano.

Innanzitutto osserviamo che in quasi tutti i casi non sono previsti input, quindi i problemi risolti dai vari pseudo-algoritmi sono in genere assai specifici, ovvero poco generali. Questo tuttavia è ininfluente ai fini di stabilire se si possano o meno definire algoritmi, come abbiamo visto sopra.

Algoritmo di auto-insegnamento Non è un algoritmo perché non soddisfa né l'ultima proprietà (fattibilità), a causa del passo 2 ("cercare di risolvere il problema"), né la prima proprietà di finitezza, in quanto potrebbe non terminare mai, se non si riesce a risolvere il problema e non si ha la soluzione nel testo. In questo caso il compito che si vorrebbe svolgere è abbastanza generale, poiché l'input sembra essere un problema qualsiasi del libro di testo.

Le lasagne al forno In questo caso le proprietà in discussione sono quelle che riguardano la precisione e la fattibilità, in quanto alcune istruzioni sono imprecise (per quanto cuocere?) e alcune istruzioni non sono sufficientemente elementari (per esempio: preparare il ragù o la besciamella). Per definire questo come algoritmo bisognerebbe specificare quale è il suo interprete; in particolare, un cuoco potrebbe considerare i passi sufficientemente precisi e elementari.

Il fatto che l'ordine delle istruzioni 1 e 2 si possa invertire senza cambiare il risultato non ha nessuna rilevanza ai fini di stabilire se si tratta di un algoritmo oppure no.

Algoritmo per accedere a un PC del laboratorio La proprietà in discussione in questo caso è quella della precisione, poiché la procedura non considera tutti i casi possibili. La procedura su può considerare un algoritmo se si specifica la situazione iniziale (prima di cominciare il PC è già acceso, funzionante, e pronto sulla schermata di *login*) e si assume che i tutor siano presenti in aula durante l'esecuzione dell'algoritmo.

Quanto è lunga la lista della spesa? Come già detto, questo pseudo-algoritmo non si presenta come una *sequenza di istruzioni*, ma come un insieme di regole (si noti che questo è l'unico pseudo-algoritmo in cui non si usa un elenco numerato) che determinano una sequenza di operazioni. Tutte le proprietà sono soddisfatte, quindi questo si deve considerare un algoritmo. In effetti, questa procedura è immediatamente traducibile in un vero programma scritto in PROLOG (linguaggio di programmazione logica) o LISP (linguaggio di programmazione funzionale).

Ricetta del cocktail Cosmopolitan Anche in questo caso la proprietà in discussione è quella della precisione, in quanto alcune istruzioni sono vaghe (per quanto shakerare?). Come per le lasagne, per definire questo come algoritmo bisognerebbe specificare quale è il suo interprete; in particolare, un barista potrebbe considerare i passi sufficientemente precisi e elementari.

Il fatto che l'ordine degli ingredienti si possa invertire senza cambiare il risultato è assolutamente irrilevante ai fini di stabilire se si tratta di un algoritmo oppure no.

Somma in colonna di due numeri Tutte le proprietà sono soddisfatte, salvo forse considerare impreciso il primo passaggio (seconda proprietà: precisione).

Il compito che si vorrebbe svolgere in questo caso è generale, poiché l'input è dato da una coppia di numeri qualsiasi. Non si vuole cioè risolvere un problema specifico (ad esempio: la somma tra 12 e 36), ma un tipo specifico di problemi, ovvero la somma di due numeri qualunque.

L'output è a sua volta un numero; la relazione tra input e output dell'algoritmo non è però quella suggerita dal titolo, infatti se i due numeri in input sono maggiori di 100, l'output non sarà la loro somma. Perciò l'algoritmo non può dirsi corretto per il problema specificato, a meno di limitarci a considerare come input solo coppie di numeri di due cifre la cui somma è anch'essa di due cifre.

Volendolo considerare invece per quello che è, e cioè un algoritmo corretto per il "calcola della somma di due numeri la cui somma è inferiore a 100", potremmo tuttavia rilevare il fatto che si tratta di un algoritmo non sufficientemente generale.

Piegatura di un foglio di carta in formato A4 Questo non può essere considerato un algoritmo perché l'ultima proprietà (fattibilità) non è verificata, in quanto non è possibile ripiegare un foglio su se stesso così tante volte. Inoltre manca la seconda proprietà (precisione) poiché non è chiaro come debba essere piegato il foglio.

Operazioni matematiche In questo caso tutte le proprietà sono soddisfatte quindi si tratta di un algoritmo. In particolare si noti che l'algoritmo riceve in input due numeri x e y e stampa in output sempre il numero 100. Non sembra esserci relazione tra input e output, ma in realtà la relazione si può formulare così: "l'output è 100 non importa quali siano gli input" (in termini matematici, l'algoritmo calcola una *funzione costante*).

Tuttavia l'algoritmo non è corretto rispetto al problema evocato dal titolo (fare operazioni matematiche) e non è efficiente perché svolge operazioni totalmente inutili rispetto all'output prodotto (la stampa di 100). Non è certo un *buon* algoritmo, pur tuttavia è un algoritmo!

7 Conclusioni

Come si è visto in questa lunga discussione, l'attività sugli pseudo-algoritmi può fornire lo spunto per approfondire il concetto di algoritmo in diverse direzioni. A seconda degli aspetti che risultano maggiormente dibattuti durante i lavori di gruppo, si può scegliere quali elementi riprendere nella fase conclusiva del lavoro, in cui si presenta la definizione di Knuth, quale fonte autorevole cui poter fare riferimento.

Non è assolutamente necessario che vengano toccati, né tanto meno approfonditi, tutti gli aspetti descritti in questo documento, tuttavia è importante che il conduttore dell'attività ne abbia contezza, per poter rispondere in maniera appropriata a eventuali dubbi, domande, osservazioni. Concludiamo dunque il documento con un elenco schematico degli elementi fondamentali del discorso, col solo scopo di fornirne una sintesi:

- idea di base di algoritmo come sequenza (dunque ordinata) di istruzioni per svolgere un compito;
- caratteristiche fondamentali di ogni algoritmo per poter essere definito tale: finitezza, precisione, fattibilità, dati in ingresso e in uscita e loro relazione;
- altre proprietà desiderabili che un algoritmo può presentare (qui abbiamo citato generalità, efficienza, leggibilità, manutenibilità, modularità);
- difficoltà di descrivere un algoritmo a parole e ruolo dell'interprete automatico e dei linguaggi di programmazione;
- opportunità di estendere la definizione per includere altri paradigmi di programmazione oltre a quello imperativo.

Allegato: Pseudo-algoritmi da discutere

Algoritmo di auto-insegnamento

1. Leggere il problema
2. Cercare di risolverlo
3. Guardare la soluzione nel testo (se c'è...)
4. Se OK passa al prossimo problema, se no vai a 1

Le lasagne al forno

1. Preparare il ragù
2. Preparare la besciamella
3. Bollire la pasta
4. Mettere a strati ragù, besciamella, pasta nella teglia
5. Cuocere in forno

Algoritmo per accedere a un PC del laboratorio

1. Accendere lo schermo se è spento
2. Scrivere il proprio ¡username¿ nella riga in cui compare la scritta login
3. Scrivere la propria ¡password¿ nella riga in cui compare la scritta password
4. Se il sistema risponde con la frase: "utente non abilitato", chiamare il tutor

Quanto è lunga la lista della spesa?

- Una lista vuota è una lista senza articoli
- $[X|L]$ è la lista che ha X come primo articolo, e poi tutti gli articoli della lista L
- La lunghezza della lista vuota (senza articoli) è 0
- Se la lunghezza della lista L è m e vale la relazione $n = m + 1$, allora la lunghezza della lista $[X|L]$ è n

Ricetta del cocktail Cosmopolitan

1. Mettere nello shaker, nell'ordine,
   (a) 3 cubetti di ghiaccio
   (b) 5 parti di Vodka
   (c) 3 parti di Cointreau
   (d) 2 parti di succo di lime
2. Shakerare
3. Versare in coppetta da cocktails lunga
4. Decorare con mezza fetta di lime

Somma in colonna di due numeri

1. Incolonna i numeri
2. Somma le cifre della colonna più a destra
3. Se il risultato è composto da una cifra, scrivi il risultato
4. altrimenti scrivi le unità del risultato e riporta le decine nella colonna immediatamente a sinistra
5. Se ci sono cifre nella colonna immediatamente a sinistra, somma le cifre di tale colonna
6. altrimenti hai finito

Piegatura di un foglio di carta in formato A4

1. Piega il foglio a metà
2. Ripeti il passo precedente 16 volte

Operazioni matematiche

1. Leggi due numeri x e y
2. Somma x e il triplo di y
3. Moltiplica il risultato per 3
4. Aggiungi 73
5. Stampa 100