Università di Udine, Facoltà di Scienze della Formazione

Corso di Informatica Applicata alla Didattica
(Giorgio T. Bagni)

Appunti: algoritmi numerici

1. IL CONCETTO DI ALGORITMO

1.1. Una sequenza di istruzioni

Spesso, in matematica (e non solo), la risoluzione di un problema può essere suddivisa in *fasi* da eseguire una dopo l'altra. Anzi, quasi sempre la scomposizione di una procedura lunga e complicata in più fasi distinte rende assai più chiaro e comprensibile l'intero procedimento risolutivo del problema in questione. Inoltre, è auspicabile che le caratteristiche operative di queste fasi siano fissate con precisione, e che la stessa scrupolosa attenzione sia prestata alla comunicazione del "da farsi" all'operatore che sarà chiamato a risolvere il problema.

Quanto sopra accennato ci consente di dare una prima introduzione del concetto di algoritmo.

Definizione 1. Diremo *algoritmo* (o *procedura effettiva*) una sequenza ordinata di istruzioni, ciascuna delle quali è descritta precisamente. L'esecuzione di ogni singola istruzione è detta *passo*.

Osservazione. La descrizione precedente è, in effetti, soltanto discorsiva, ovvero non "definisce" il concetto di algoritmo con il rigore generalmente associato alle definizioni matematiche vere e proprie. Riteniamo tuttavia che l'importanza degli algoritmi in una moderna visione della matematica sia tale da giustificare l'uso forse non del tutto rigoroso della qualifica di "definizione" anche in questo caso.

Proponiamo un esempio di algoritmo nel quale compariranno alcune fasi distinte come la lettura dei *dati*, la loro *elaborazione*, la determinazione di un *risultato*.

Esempio 1. Descriviamo, attraverso un algoritmo, il calcolo della misura (approssimata alla terza cifra decimale) dell'ipotenusa di un triangolo rettangolo con i cateti che misurano $b \in \mathbf{Q}^+$ e $c \in \mathbf{Q}^+$ (da eseguire mediante una calcolatrice tascabile in grado di eseguire la radice quadrata).

Istruzione 0. Inizio. Passare all'istruzione seguente.
Istruzione 1. Leggere i razionali positivi b, c. Passare all'istruzione seguente.
Istruzione 2. Calcolare b^2. Passare all'istruzione seguente.
Istruzione 3. Calcolare c^2. Passare all'istruzione seguente.
Istruzione 4. Calcolare b^2+c^2. Passare all'istruzione seguente.
Istruzione 5. Estrarre la radice quadrata di b^2+c^2. Passare all'istruzione seguente.
Istruzione 6. Attribuire a *risultato* il numero ottenuto troncato dopo la terza cifra decimale. Passare all'istruzione seguente.
Istruzione 7. Fine.

Si noti la presenza dell'istruzione 0, "inizio" e dell'istruzione conclusiva, "fine"; il ruolo di quest'ultima è di avvertire l'esecutore dell'algoritmo che la sequenza di operazioni da eseguire è terminata.

L'istruzione 1 è molto importante: essa attua l'indispensabile inizializzazione del problema, ovvero fissa i dati sui quali l'algoritmo sarà chiamato ad operare. L'istruzione 6 individua il risultato del problema.

Osservazione. L'insieme ordinato delle istruzioni presentato nell'esempio è un algoritmo in quanto ciascuna delle istruzioni è costituita da un'informazione precisa sull'operazione che l'esecutore dovrà compiere con una calcolatrice. Qualche problema potrebbe sorgere se non avessimo indicato l'uso di tale calcolatrice e l'approssimazione richiesta: come avremmo dovuto calcolare, in tale caso, la radice quadrata di b^2+c^2? Ccon quale metodo? E con quale approssimazione?

Esempio 2. La sequenza ordinata di istruzioni che presenteremo non è un algoritmo propriamente detto.

Istruzione 0. Inizio. Passare all'istruzione seguente.
Istruzione 1. Spremere tre pompelmi. Passare all'istruzione seguente.
Istruzione 2. Aggiungere acqua minerale.

Passare all'istruzione seguente.
Istruzione 3. Rinfrescare in frigorifero quanto basta.
Passare all'istruzione seguente.
Istruzione 4. Servire dopo avere aggiunto alcuni cubetti di ghiaccio.
Passare all'istruzione seguente.
Istruzione 5. Fine.

Infatti la "ricetta" per ottenere una dissetante spremuta non ha i requisiti di chiarezza e di precisione indispensabili per poter essere considerata un algoritmo; molti sono i dubbi che restano, dopo la sua lettura.

Ad esempio:

- "aggiungere acqua minerale": *quanta?*
- "rinfrescare in firgorifero": *a quale temperatura? In quale recipiente? Per quanto tempo?*
- "alcuni cubetti di ghiaccio": *quanti?*

1.2. Eseguibilità di algoritmi

Abbiamo stabilito che un algoritmo è una sequenza ordinata di istruzioni caratterizzate da alcuni requisiti di precisione. Tali requisiti, infatti, garantiscono che l'esecuzione dell'algoritmo possa avere luogo senza ambiguità. Ma tutto ciò è sufficiente a rendere un algoritmo *eseguibile*? In altri termini: ogni sequenza ordinata di istruzioni descritta dalla definizione può essere effettivamente eseguita da un calcolatore? Oppure possono esistere sequenze particolarmente difficoltose, la cui messa in pratica incontra ostacoli anche insormontabili?

Esempio 3. L'algoritmo illustrato nell'esempio 1 è eseguibile senza difficoltà alcuna. Infatti, se applichiamo l'algoritmo al calcolo della misura $\mathbf{Q}^+$ approssimata alla terza cifra decimale dell'ipotenusa di un triangolo rettangolo con i cateti che misurano $b = 12$ e $c = 18$ (da eseguire mediante una calcolatrice tascabile), otteniamo la seguente successione dei passi:

Passo 1.	(si esegue l'*istruzione 1*)	È: $b = 12$; $c = 18$; si passa all'istruzione 2.
Passo 2.	(si esegue l'*istruzione 2*)	$b^2 = 144$; si passa all'istruzione 3.
Passo 3.	(si esegue l'*istruzione 3*)	$c^2 = 324$;

		si passa all'istruzione 4.
Passo 4.	(si esegue l'*istruzione 4*)	$b^2+c^2 = 468$; si passa all'istruzione 5.
Passo 5.	(si esegue l'*istruzione 5*)	$\sqrt{b^2+c^2} = 21{,}63330765...$; si passa all'istruzione 6.
Passo 6.	(si esegue l'*istruzione 6*)	È: *risultato* = 21,633; si passa all'istruzione 7.
Passo 7.	(si esegue l'*istruzione 7*)	Fine del procedimento.

Esempio 4. Consideriamo la sequenza ordinata di istruzioni:

Istruzione 0.	Inizio. Passare all'istruzione seguente.
Istruzione 1.	Se la congettura di Goldbach è vera, porre: $n = 4$; se la congettura di Goldbach è falsa, porre: $n = 7$. Passare all'istruzione seguente.
Istruzione 2.	Calcolare $2 \cdot n$. Passare all'istruzione seguente.
Istruzione 3.	Calcolare $2 \cdot n - 5$. Passare all'istruzione seguente.
Istruzione 4.	Attribuire a *risultato* il valore $2 \cdot n - 5$. Passare all'istruzione seguente.
Istruzione 5.	Fine.

Questa sequenza di istruzioni sembra rispettare senza alcun problema la definizione di algoritmo. Non sottovalutiamo, però, l'istruzione 1, che fa esplicito riferimento alla *congettura di Goldbach*; ricordiamo che la congettura di Goldbach è l'enunciato che afferma:

ogni naturale pari maggiore di 2 è la somma di due numeri primi

Attenzione: *nessuno* è stato finora in grado di dimostrare o di smentire tale celebre affermazione! Pertanto: l'istruzione 1 della sequenza precedente è certamente precisa e chiara, giacché la congettura di Goldbach, indubbiamente, sarà vera o falsa.

Ma allo stato attuale della conoscenza matematica (novembre 2008) *non* appare possibile stabilire la verità o la falsità di tale congettura, e pertanto *non* è possibile eseguire l'istruzione 1 della sequenza proposta.

Situazioni come quella presentata nel precedente esempio sono interessanti e, per certi versi, affascinanti. Noi, tuttavia, rivolgeremo la nostra attenzione esclusivamente ad algoritmi eseguibili.

1.3. Quanti passi sono necessari per eseguire un algoritmo?

Al lettore attento non sarà sfuggita la differenza tra *istruzione* e *passo*: diciamo *istruzione* ogni singola riga che costituisce l'algoritmo, *passo* ogni singola operazione che, indicata da un'istruzione, viene eseguita.

Talvolta il numero delle istruzioni che costituiscono l'algoritmo coincide con il numero totale dei passi necessari all'esecuzione dell'algoritmo stesso (il lettore è invitato a rivedere l'esempio 1): ciò accade quando ogni istruzione viene eseguita una ed una sola volta. Ma non sempre accade così.

Alcuni algoritmi, infatti, prevedono che una o più istruzioni siano da eseguire *più* volte; ed in alcuni casi, il numero di passi dipende dai dati con i quali il problema viene inizializzato: può anche accadere che, in corrispondenza di dati che rispettano alcune particolari condizioni (espressamente enunciate nelle istruzioni dell'algoritmo), alcune istruzioni siano da "saltare".

Esempio 5. Consideriamo l'algoritmo espresso dalla sequenza ordinata di istruzioni:

Istruzione 0.	Inizio. Passare all'istruzione seguente.
Istruzione 1.	Leggere il naturale n. Passare all'istruzione seguente.
Istruzione 2.	Se $n<2$, passare all'istruzione 6; altrimenti passare all'istruzione seguente.
Istruzione 3.	Se $n>1000$, passare all'istruzione 6; altrimenti passare all'istruzione seguente.
Istruzione 4.	Calcolare $2 \cdot n$. Passare all'istruzione seguente.
Istruzione 5.	Sostituire n con $2 \cdot n$. Passare all'istruzione 3.
Istruzione 6.	Attribuire a *risultato* il valore di n. Passare all'istruzione seguente.
Istruzione 7.	Fine.

Le istruzioni 2 e 3 sono particolarmente interessanti: esse, infatti, propongono un'alternativa e richiedono pertanto una valutazione da parte dell'esecutore: tale valutazione potrà far sì che l'esecuzione dell'algoritmo termini dopo pochi passi, oppure che essa prosegua per un maggiore numero di passi.

Esaminiamo che accade se $n = 1$.

Passo 1.	(si esegue l'*istruzione 1*)	È: $n = 1$; si passa all'istruzione 2.
Passo 2.	(si esegue l'*istruzione 2*)	Essendo $n<2$, si passa all'istruzione 6.

Passo 3.	(si esegue l'*istruzione 6*)	È: *risultato* = 1; si passa all'istruzione 7.
Passo 4.	(si esegue l'*istruzione 7*)	Fine del procedimento.

Se $n = 1$, il procedimento termina in 4 passi.
Esaminiamo che accade se $n = 1200$.

Passo 1.	(si esegue l'*istruzione 1*)	È: $n = 1200$; si passa all'istruzione 2.
Passo 2.	(si esegue l'*istruzione 2*)	Non essendo $n<2$, si passa all'istruzione 3.
Passo 3.	(si esegue l'*istruzione 3*)	Essendo $n>1000$, si passa all'istruzione 6.
Passo 4.	(si esegue l'*istruzione 6*)	È: *risultato* = 1200; si passa all'istruzione 7.
Passo 5.	(si esegue l'*istruzione 7*)	Fine del procedimento.

Se $n = 1200$, il procedimento termina in 5 passi.
Esaminiamo infine che accade se $n = 200$.

Passo 1.	(si esegue l'*istruzione 1*)	È: $n = 200$; si passa all'istruzione 2.
Passo 2.	(si esegue l'*istruzione 2*)	Non essendo $n<2$, si passa all'istruzione 3
Passo 3.	(si esegue l'*istruzione 3*)	Non essendo $n>1000$, si passa all'istruzione 4.
Passo 4.	(si esegue l'*istruzione 4*)	$2 \cdot n = 400$; si passa all'istruzione 5.
Passo 5.	(si esegue l'*istruzione 5*)	È: $n = 400$; si passa all'istruzione 3.
Passo 6.	(si esegue l'*istruzione 3*)	Non essendo $n>1000$, si passa all'istruzione 4.
Passo 7.	(si esegue l'*istruzione 4*)	$2 \cdot n = 800$; si passa all'istruzione 5.
Passo 8.	(si esegue l'*istruzione 5*)	È: $n = 800$; si passa all'istruzione 3.
Passo 9.	(si esegue l'*istruzione 3*)	Non essendo $n>1000$, si passa all'istruzione 4.
Passo 10.	(si esegue l'*istruzione 4*)	$2 \cdot n = 1600$; si passa all'istruzione 5.
Passo 11.	(si esegue l'*istruzione 5*)	È: $n = 1600$; si passa all'istruzione 3.

Passo 12.	(si esegue l'*istruzione 3*)	Essendo $n>1000$, si passa all'istruzione 6.
Passo 13.	(si esegue l'*istruzione 6*)	È: *risultato* = 1600; si passa all'istruzione 7.
Passo 14.	(si esegue l'*istruzione 7*)	Fine del procedimento.

Se $n = 200$, il procedimento termina in 14 passi.

Finora ci siamo occupati di algoritmi la cui esecuzione si conclude in un numero *finito* di passi. Ma accade *sempre* così?

Nell'esempio seguente è illustrato un algoritmo (sempre che sia lecito chiamarlo ancora così...) che *non* si conclude in un numero finito di passi.

Esempio 6. Consideriamo l'algoritmo (sempre che sia lecito chiamarlo ancora così...) espresso dalla sequenza ordinata di istruzioni:

Istruzione 0.	Inizio. Passare all'istruzione seguente.
Istruzione 1.	Leggere il reale x. Passare all'istruzione seguente.
Istruzione 2.	Se $x = 0$, passare all'istruzione 5; altrimenti passare all'istruzione seguente.
Istruzione 3.	Calcolare $x/2$. Passare all'istruzione seguente.
Istruzione 4.	Attribuire a x il valore $x/2$. Passare all'istruzione 2.
Istruzione 5.	Fine.

In quanti passi si conclude l'esecuzione della procedura sopra indicata?

Se $x = 0$, la situazione è facilmente controllabile: vengono eseguite, in ordine, le istruzioni (0), 1, 2, 5; quindi il numero di passi eseguiti è finito.

Ma se x è un reale diverso da 0, il procedimento viene ad essere ben diverso; esaminiamo che accade se $x = 1$:

Passo 1.	(si esegue l'*istruzione 1*)	$x = 1$; si passa all'istruzione 2.
Passo 2.	(si esegue l'*istruzione 2*)	Non essendo $x = 0$, si passa all'istruzione 3.
Passo 3.	(si esegue l'*istruzione 3*)	$x/2 = 0{,}5$; si passa all'istruzione 4.
Passo 4.	(si esegue l'*istruzione 4*)	È: $x = 0{,}5$; si passa all'istruzione 2.
Passo 5.	(si esegue l'*istruzione 2*)	Non essendo $x = 0$, si passa all'istruzione 3.

Passo 6.	(si esegue l'*istruzione 3*)	$x/2 = 0{,}25$; si passa all'istruzione 4.
Passo 7.	(si esegue l'*istruzione 4*)	È: $x = 0{,}25$; si passa all'istruzione 2.
Passo 8.	(si esegue l'*istruzione 2*)	Non essendo $x = 0$, si passa all'istruzione 3.
Passo 9.	(si esegue l'*istruzione 3*)	$x/2 = 0{,}125$; si passa all'istruzione 4.
Passo 10.	(si esegue l'*istruzione 4*)	È: $x = 0{,}125$; si passa all'istruzione 2.
Passo 11.	(si esegue l'*istruzione 2*)	Non essendo $x = 0$, si passa all'istruzione 3.
Passo 12.	(si esegue l'*istruzione 3*)	$x/2 = 0{,}0625$; si passa all'istruzione 4.
...	...	...

Quando si conclude il procedimento precedente? Non è possibile stabilire il numero di passi necessari a concludere l'esecuzione dell'algoritmo: continuando a dimezzare un reale non nullo, infatti, troviamo sempre dei reali non nulli. Pertanto, dobbiamo concludere che l'algoritmo ora presentato, inizializzato con un reale x non nullo, *non può essere eseguito in un numero finito di passi*.

Non ci occuperemo di situazioni di tal genere: la considerazione di procedure che *non* si arrestano dopo un numero finito di passi, infatti, richiede l'analisi di problemi specifici (questioni di convergenza e di stabilità). Lo stesso termine algoritmo, come più volte notato, potrebbe essere messo in discussione.

2. L'ALGORITMO DI EUCLIDE

2.1. Determiniamo il massimo comune divisore di due naturali

La storia della matematica è ricca di idee feconde, di spunti attuali: un classico algoritmo è fatto risalire all'opera di Euclide (III sec. a.C.): si tratta di un procedimento semplice ed efficace per determinare il massimo comune divisore di due naturali (detto *metodo della "divisione euclidea"*).

Ricordiamo che, considerati due numeri naturali a e b, con $a>b$, dividendo a per b otterremo come quoziente il naturale n_1 e come resto r_1:

$$a{:}b = n_1 + \frac{r_1}{b}$$

Dividendo poi ancora b per r_1, otterremo come quoziente n_2 e come resto r_2:

$$b{:}r_1 = n_2 + \frac{r_2}{r_1}$$

Iterando il procedimento, ovvero dividendo r_1 per r_2, otterremo come quoziente n_3 e come resto r_3:

$$r_1{:}r_2 = n_3 + \frac{r_3}{r_2}$$

e così via. Possiamo riassumere quanto sino ad ora ottenuto nella scrittura:

$$a{:}b = n_1 + \frac{r_1}{b} = n_1 + \frac{1}{\frac{b}{r_1}} = n_1 + \frac{1}{n_2 + \frac{r_2}{r_1}} = n_1 + \frac{1}{n_2 + \frac{1}{\frac{r_1}{r_2}}} = n_1 + \frac{1}{n_2 + \frac{1}{n_3 + \ldots}}$$

Se a e b sono numeri naturali, il procedimento ha termine: ciò significa che si giungerà (in un numero *finito* di passi) ad una situazione in cui la divisione da effettuare è esatta (ovvero con quoziente q e resto nullo). Ad esempio, potrebbe accadere che, nella situazione seguente, r_2 sia divisibile per r_3:

$$a{:}b = n_1 + \frac{1}{n_2 + \frac{1}{n_3 + \frac{r_3}{r_2}}} = n_1 + \frac{1}{n_2 + \frac{1}{n_3 + \frac{1}{\frac{r_2}{r_3}}}} = n_1 + \frac{1}{n_2 + \frac{1}{n_3 + \frac{1}{q}}}$$

Si dimostra allora che *il massimo comune divisore dei naturali a, b è l'ultimo resto trovato*: nel caso esemplificato, r_3.

Esempio 7. Determiniamo, mediante l'algoritmo di Euclide, il massimo comune divisore di 330 e 84:

330:84 = 3, con resto 78

84:78 = 1, con resto **6**
78:**6** = 13 con resto 0

(dove abbiamo evidenziato l'ultimo resto).
Possiamo riassumere quanto scritto in:

$$330:84 = 3+\frac{78}{84} = 3+\frac{1}{1+\frac{6}{78}} = 3+\frac{1}{1+\frac{1}{\frac{78}{6}}} = 3+\frac{1}{1+\frac{1}{13}}$$

In base a quanto sopra affermato, concludiamo:

MCD (330; 84) = 6

Esempio 8. Applichiamo l'algoritmo di Euclide ad una coppia di numeri coprimi (ovvero tali che le rispettive scomposizioni in fattori primi non abbiano alcun fattore comune): 231 e 80.

231:80 = 2 con resto 71
80:71 = 1 con resto 9
71:9 = 7 con resto 8
9:8 = 1 con resto **1**
1:**1** = 1con resto 0

(dove abbiamo evidenziato l'ultimo resto).
In base a quanto sopra affermato, concludiamo:

MCD (231; 80) = 1

e ciò (pur apparendo banale) è corretto: infatti il massimo comune divisore di due naturali coprimi qualsiasi è 1.

2.2. Formalizzazione dell'algoritmo di Euclide

L'algoritmo di Euclide, precedentemente illustrato, può essere sintetizzato nella sequenza ordinata di istruzioni:

Istruzione 0. Inizio. Passare all'istruzione seguente.

Istruzione 1. Leggere i naturali a, b, con $a>b$.
Passare all'istruzione seguente.
Istruzione 2. Dividere a per b, ottenendo un quoziente naturale ed il resto r. Passare all'istruzione seguente.
Istruzione 3. Se $r=0$, passare all'istruzione 5;
altrimenti passare all'istruzione seguente.
Istruzione 4. Attribuire ad a il valore b ed attribuire a b il valore r.
Passare all'istruzione 2.
Istruzione 5. Attribuire a *MCD* il valore b.
Passare all'istruzione seguente.
Istruzione 6. Fine.

Osservazione. Può sorgere un problema nell'interpretazione della lista di istruzioni ora compilata. Consideriamo infatti quanto richiesto nell'istruzione 2:

Dividere a per b, ottenendo un quoziente naturale ed il resto r.

Sembra molto semplice, ma *come* si esegue, operativamente, la divisione del naturale a per il naturale b in modo da ottenere il quoziente (naturale) ed il resto (naturale)? Una semplice prova con la calcolatrice tascabile sarà sufficiente a convincerci che l'operazione richiesta non è elementare come sembra: ad esempio, siano: $a = 50$ e $b = 6$. Con una calcolatrice tascabile otteniamo:

50:6 = 8,333333333

Già la lettura del quoziente (naturale), 8, a partire dal risultato, 8,333333333, non è immediata: bisogna "trascurare" le cifre dopo la virgola. Problemi ancora maggiori, poi, sorgono nella determinazione del resto (naturale), 2, *resto che è necessario per proseguire l'esecuzione dell'algoritmo*: bisognerebbe moltiplicare per 6 la parte decimale del risultato, ovvero 8,333333333−8, ma attenzione: 0,333333333×6 = 1,999999998 e quindi bisognerebbe correggere...

Queste noiose difficoltà, concettualmente per nulla insormontabili, ci fanno capire che nel nostro algoritmo qualcosa non va. Così com'è scritto, quindi, l'algoritmo non raggiunge appieno gli auspicabili requisiti di chiarezza e di immediatezza.

Per eliminare le imperfezioni evidenziate nell'osservazione precedente, l'algoritmo della divisione euclidea può essere così riscritto.

Istruzione 0. Inizio. Passare all'istruzione seguente.
Istruzione 1. Leggere i naturali a, b, con $a>b$.
Passare all'istruzione seguente.

Istruzione 2. Porre $k = 2 \cdot b$. Passare all'istruzione seguente.
Istruzione 3. Se $k \geq a$, passare all'istruzione 6;
altrimenti passare all'istruzione seguente.
Istruzione 4. Calcolare $k+b = k'$. Passare all'istruzione seguente.
Istruzione 5. Attribuire a k il valore k'. Passare all'istruzione 3.
Istruzione 6. Se $k > a$, passare all'istruzione seguente;
altrimenti passare all'istruzione 9.
Istruzione 7. Porre $r = a–k+b$. Passare all'istruzione seguente.
Istruzione 8. Attribuire ad a il valore b ed attribuire a b il valore r.
Passare all'istruzione 2.
Istruzione 9. Attribuire a *MCD* il valore b.
Passare all'istruzione seguente.
Istruzione 10. Fine.

La sequenza di istruzioni ora scritta appare certamente più lunga e più complicata della precedente: ad esempio, l'esecuzione dell'algoritmo di Euclide per calcolare il MCD dei naturali $a = 330$ e $b = 84$ (come già fatto, in pochi passaggi, nell'esempio 7) richiede, utilizzando quest'ultima sequenza di istruzioni, ben 55 passi; ma abbiamo ottenuto una formalizzazione dell'algoritmo *chiara e direttamente eseguibile in tutti i suoi passaggi*, come potremo constatare nell'esempio seguente.

Esempio 9. Utilizzando la precedente formalizzazione dell'algoritmo di Euclide, calcoliamo il massimo comune divisore dei naturali $a = 42$ e $b = 30$.

Passo 1.	(si esegue l'*istruzione 1*)	È: $a = 42$; $b = 30$; si passa all'istruzione 2.
Passo 2.	(si esegue l'*istruzione 2*)	$k = 2 \cdot 30 = 60$; si passa all'istruzione 3.
Passo 3.	(si esegue l'*istruzione 3*)	Essendo $60 \geq 42$, si passa all'istruzione 6.
Passo 4.	(si esegue l'*istruzione 6*)	Essendo $60 > 42$, si passa all'istruzione 7.
Passo 5.	(si esegue l'*istruzione 7*)	$r = 42–60+30 = 12$; si passa all'istruzione 8.
Passo 6.	(si esegue l'*istruzione 8*)	È: $a = 30$; $b = 12$; si passa all'istruzione 2.
Passo 7.	(si esegue l'*istruzione 2*)	$k = 2 \cdot 12 = 24$; si passa all'istruzione 3.
Passo 8.	(si esegue l'*istruzione 3*)	Non essendo $24 \geq 30$, si passa all'istruzione 4.

Passo 9.	(si esegue l'*istruzione 4*)	24+12 = 36 = *k'*; si passa all'istruzione 5.
Passo 10.	(si esegue l'*istruzione 5*)	*k* = 36; si passa all'istruzione 3.
Passo 11.	(si esegue l'*istruzione 3*)	Essendo 36 ≥ 30, si passa all'istruzione 6.
Passo 12.	(si esegue l'*istruzione 6*)	Essendo 36 > 30, si passa all'istruzione 7.
Passo 13.	(si esegue l'*istruzione 7*)	*r* = 30–36+12 = 6; si passa all'istruzione 8.
Passo 14.	(si esegue l'*istruzione 8*)	È: *a* = 12; *b* = 6; si passa all'istruzione 2.
Passo 15.	(si esegue l'*istruzione 2*)	*k* = 2·6 = 12; si passa all'istruzione 3.
Passo 16.	(si esegue l'*istruzione 3*)	Essendo 12 ≥ 12, si passa all'istruzione 6.
Passo 17.	(si esegue l'*istruzione 6*)	Non essendo 12 > 12, si passa all'istruzione 9.
Passo 18.	(si esegue l'*istruzione 9*)	È: *MCD* = 6; si passa all'istruzione 10.
Passo 19.	(si esegue l'*istruzione 10*)	Fine.

Possiamo dunque concludere:

MCD (42; 30) = 6

2.3. Partiamo da due grandezze qualsiasi…

Di grande interesse è il procedimento precedente nel caso in cui *a* e *b* siano due grandezze qualsiasi (ad esempio: le lunghezze di due segmenti):

- se *a* e *b* sono *commensurabili* (cioè se essi ammettono una sottomultipla comune) il procedimento ha termine in un numero finito di passi; si giunge al rapporto (razionale) *a*/*b*;
- se *a* e *b* sono *incommensurabili* (cioè se essi non ammettono alcuna sottomultipla comune) il procedimento non ha termine in un numero finito di passi; il rapporto (irrazionale) tra *a* e *b* è allora dato dalla *frazione continua* illimitata:

$$n_1 + \cfrac{1}{n_2 + \cfrac{1}{n_3 + \cfrac{1}{n_4 + \ldots}}}$$

In quest'ultimo caso, dunque, il procedimento al quale si fa riferimento avrebbe delle caratteristiche che non si inquadrano negli algoritmi veri e propri, che terminano dopo un numero finito di passi. Ma l'importanza concettuale della questione merita di essere sottolineata.

Esempio 10. Si dimostra che:

$$\frac{\sqrt{5}-1}{2} = \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \ldots}}}$$

Tale sviluppo può essere intuito considerando che

$$x = \frac{\sqrt{5}-1}{2}$$

è la radice positiva dell'equazione:

$$x^2 + x - 1 = 0$$

la quale può essere scritta nella forma:

$$x = \frac{1}{1+x}$$

Sostituendo ripetutamente $\frac{1}{1+x}$ al posto della x al denominatore, ricaviamo una scrittura che si approssima sempre più alla frazione continua poco fa introdotta.

---

3. IL GRAFO DI FLUSSO DI UN ALGORITMO

3.1. Che cos'è un grafo?

In questa sezione utilizzeremo uno strumento elegante ed efficace, messo a disposizione dalla matematica moderna. Ci baseremo sui *grafi*, introdotti dalla definizione seguente.

Definizione 2. Un *grafo lineare* è costituito dalla coppia (I; φ), dove:

- I è un insieme di punti, detti *vertici*;
- φ è una relazione binaria definita in I (ovvero: un sottoinsieme di I×I).

Se conveniamo di indicare ogni coppia ordinata (P; Q) di $\varphi \subseteq I \times I$ con una arco di curva orientata che unisce P e Q, otteniamo un insieme J di *archi*.

Nell'esempio seguente è indicato un grafo lineare.

Esempio 11. Lasciamo al lettore la verifica che il seguente grafo è lineare:

$$I = \{A; B; C; D; E\}$$

$$\varphi = \{(A; B); (B; C); (B; D); (D; E)\}$$

3.2. La rappresentazione di un algoritmo

Nella sezione 1 abbiamo affermato che un algoritmo è una sequenza ordinata di istruzioni; abbiamo anche constatato che spesso, in un algoritmo, le istruzioni da eseguire dipendono dai dati, ovvero dalle quantità su cui l'algoritmo stesso si trova a dover operare: in altri termini, possono verificarsi situazioni di biforcazione, sequenze alternative di istruzioni da eseguire.

Un utile metodo per rappresentare praticamente un algoritmo è basato sui grafi: in un grafo lineare, ogni istruzione sarà associata ad un nodo, e l'ordine di esecuzione delle singole istruzioni potrà essere desunto dagli archi che collegano i nodi.

Esempio 12. Lasciamo al lettore la costruzione del grafo di flusso dell'algoritmo presentato nell'esempio 1, costituito dalla sequenza di istruzioni:

Istruzione 0. Inizio. Passare all'istruzione seguente.
Istruzione 1. Leggere i razionali positivi b, c. Passare all'istruzione seguente.
Istruzione 2. Calcolare b^2. Passare all'istruzione seguente.
Istruzione 3. Calcolare c^2. Passare all'istruzione seguente.
Istruzione 4. Calcolare b^2+c^2. Passare all'istruzione seguente.
Istruzione 5. Estrarre la radice quadrata di b^2+c^2.
Passare all'istruzione seguente.
Istruzione 6. Attribuire a *risultato* il numero ottenuto troncato dopo la terza cifra decimale. Passare all'istruzione seguente.
Istruzione 7. Fine.

Si tratta di un grafo molto semplice, senza alcuna "biforcazione".

Esempio 13. Il lettore costruisca il grafo di flusso dell'algoritmo di Euclide, nella forma data dalla sequenza di istruzioni:

Istruzione 0. Inizio. Passare all'istruzione seguente.
Istruzione 1. Leggere i naturali a, b, con $a>b$.
Passare all'istruzione seguente.
Istruzione 2. Porre $k = 2 \cdot b$. Passare all'istruzione seguente.
Istruzione 3. Se $k \geq a$, passare all'istruzione 6;
altrimenti passare all'istruzione seguente.
Istruzione 4. Calcolare $k+b = k'$. Passare all'istruzione seguente.
Istruzione 5. Attribuire a k il valore k'. Passare all'istruzione 3.
Istruzione 6. Se $k > a$, passare all'istruzione seguente;
altrimenti passare all'istruzione 9.
Istruzione 7. Porre $r = a–k+b$. Passare all'istruzione seguente.
Istruzione 8. Attribuire ad a il valore b ed attribuire a b il valore r.
Passare all'istruzione 2.
Istruzione 9. Attribuire a *MCD* il valore b.
Passare all'istruzione seguente.
Istruzione 10. Fine.
