Appendice 3
La trigonometria e il calcolatore

1. Programma per risolvere i triangoli rettangoli
2. Un'osservazione sulla programmazione
3. Somma di armoniche
4. Risoluzione di equazioni trigonometriche
5. Risoluzione di triangoli qualunque
6. Le istruzioni BASIC

In questa Appendice svilupperemo alcuni programmi che risolvono problemi di trigonometria. Il primo programma riguarda la soluzione dei triangoli rettangoli e si basa sulle nozioni esposte nel cap. 1; segue un programma che disegna la somma di armoniche (cap. 3) e uno che risolve le equazioni trigonometriche (cap. 5). Vedremo infine un programma per risolvere i triangoli qualunque; si riferisce a quanto visto nel cap. 2 ma è presentato per ultimo perché è il più impegnativo.

Supporremo che gli elementi fondamentali della programmazione in linguaggio BASIC siano già noti e faremo dei passi avanti: impareremo a progettare un programma in modo quasi professionale.

Per chi avesse dei dubbi, le istruzioni BASIC sono richiamate e brevemente illustrate nell'ultimo paragrafo. Le istruzioni BASIC presenti sui vari calcolatori non sono tutte uguali; quelle utilizzate in questa Appendice sono dell'Apple II, ma è facile modificare i programmi per adattarli ad altre macchine, aiutandosi con la descrizione delle istruzioni presentata nell'ultimo paragrafo.

1. Programma per risolvere i triangoli rettangoli

Il nostro primo programma risolve i triangoli rettangoli, cioè chiede gli elementi noti di un triangolo rettangolo e calcola quelli incogniti. Per prima cosa, elenchiamo le funzioni che il programma deve svolgere:

I) accettare i valori degli elementi noti;
II) calcolare gli elementi incogniti;
III) mostrare il risultato.

Analizziamo separatamente ciascuna di queste tre funzioni.

I) **Accettare i valori degli elementi noti**. Gli elementi noti possono essere lati o angoli; inoltre, dobbiamo distinguere se fra i lati c'è anche l'ipotenusa o no. Quanto agli angoli, uno è certamente retto, perciò se viene dato un altro angolo è automaticamente determinato anche il terzo. Osserviamo poi che lati ed angoli non possono essere qualsiasi; sappiamo infatti che ogni cateto deve essere minore dell'ipotenusa e ogni angolo (tranne quello retto) deve essere minore di 90°.

Riassumiamo queste osservazioni in una lista in cui elenchiamo le operazioni che il programma dovrà svolgere:

— accettare la lunghezza dell'ipotenusa;
— accettare la lunghezza di un cateto;
— verificare che l'ipotenusa sia maggiore del cateto;
— accettare la lunghezza dell'altro cateto;
— verificare che l'ipotenusa sia maggiore anche di questo cateto;
— accettare l'ampiezza di un angolo;
— verificare che l'angolo sia minore di 90°.

Riflettiamo su questa lista per individuare ciò che ancora manca o va meglio precisato. Prima di tutto, non abbiamo ancora precisato quali sono gli elementi noti e quali, invece, quelli incogniti. Un modo semplice per farlo è questo: introdurre il valore zero per gli elementi incogniti. Dovremo avvertire l'utente di questa nostra scelta, facendo comparire un messaggio che lo avverta di battere zero per ogni elemento incognito.

Osserviamo poi che il triangolo è determinato solo se sono dati tre elementi, fra cui almeno un lato; ma poiché il triangolo è rettangolo e quindi un angolo è già noto, basta dare due elementi fra cui un lato. Converrà scrivere il programma in modo che verifichi questa condizione e sappia riconoscere se gli elementi noti bastano a determinare il triangolo.

Con queste osservazioni, la lista delle operazioni da compiere si modifica così:

— scrivere "introdurre 0 per gli elementi incogniti";
— accettare la lunghezza dell'ipotenusa;
— se l'ipotenusa è diversa da zero, incrementare di uno il numero dei lati noti;
— accettare la lunghezza di un cateto;
— se l'ipotenusa è diversa da zero, verificare che l'ipotenusa sia maggiore del cateto;
— se il cateto è diverso da zero, incrementare di uno il numero dei lati noti;

— se il numero dei lati noti è 2, allora passare alla risoluzione;
— accettare la lunghezza dell'altro cateto;
— se l'ipotenusa è diversa da zero, verificare che l'ipotenusa sia maggiore del cateto;
— se il cateto è diverso da zero, incrementare di uno il numero dei lati noti;
— se il numero dei lati noti è 2, passare alla risoluzione;
— se il numero dei lati noti è 0, avvertire che il triangolo è indeterminato;
— accettare l'ampiezza di un angolo;
— verificare che l'angolo sia minore di 90°;
— se l'angolo è uguale a zero, avvertire che il triangolo è indeterminato;
— passare alla risoluzione.

A questo punto, se avete una discreta conoscenza del BASIC potete direttamente saltare al listato della pagina seguente. Se invece avete dei dubbi, e volete evitare gli errori, vi conviene continuare a leggere senza salti.

Riscriviamo la lista delle operazioni da compiere in un linguaggio più vicino al BASIC, un linguaggio intermedio fra il nostro e quello del calcolatore. Questo linguaggio viene inventato dal programmatore; **non è** un linguaggio di programmazione ed è detto "pseudocodice". I bravi programmatori scrivono spesso i loro programmi in qualche pseudocodice, perché in questo modo evitano molti errori.

Ecco un esempio di pseudocodice che traduce la lista dei passi da compiere:

— print "introdurre 0 per gli elementi incogniti"
— input ipotenusa
— if ipotenusa>0 then latinoti=latinoti+1
— input primo cateto
— if ipotenusa >0 and primocateto>ipotenusa then impossibile
— if primocateto>0 then latinoti=latinoti+1
— if latinoti=2 then risoluzione
— input secondocateto
— if ipotenusa>0 and secondocateto>ipotenusa then impossibile
— if secondocateto>0 then latinoti=latinoti+1
— if latinoti=2 then risoluzione
— if latinoti=0 then indeterminato
— input primoangolo
— if primoangolo>90° then impossibile
— if primoangolo=0 then indeterminato
— goto risoluzione

Ora che abbiamo questo pseudocodice, esaminiamo tutte le variabili che vi compaiono e decidiamo un simbolo BASIC per ciascuna di esse, ricordando che gli interpreti BASIC più diffusi riconoscono solo simboli di uno o due caratteri. Ecco l'elenco delle variabili con una possibile scelta dei simboli:

VARIABILE	SIMBOLO
ipotenusa	IP
latinoti	LN
primocateto	C1
secondocateto	C2
primoangolo	A1

Esaminiamo ora di nuovo il nostro pseudocodice alla ricerca delle subroutine che abbiamo indicato sinteticamente con un nome, come "impossibile", ecc. Nel BASIC ogni subroutine deve essere individuata dal suo numero di linea, perciò elenchiamo queste subroutine e, per ciascuna di esse, scegliamo un numero di linea

SUBROUTINE	NUMERO DI LINEA
risoluzione	1000
indeterminato	2000
impossibile	3000

Possiamo ora scrivere le istruzioni BASIC che realizzano la fase d'inserimento dei dati:

```
10  REM  PROGRAMMA TRIANGOLI RETTANGOLI
100  REM  INSERIMENTO DATI************************
110  HOME
120  PRINT "INTRODURRE 0 PER GLI ELEMENTI INGOGNITI"
130  INPUT "IPOTENUSA=";IP
140  IF IP > 0 THEN LN = LN + 1
150  INPUT "PRIMO CATETO=";C1
160  IF IP > 0 AND C1 =  > IP THEN 3000
170  IF C1 > 0 THEN LN = LN + 1
180  IF LN = 2 THEN 1000
190  INPUT "SECONDO CATETO=";C2
200  IF IP > 0 AND C2 =  > IP THEN 3000
210  IF C2 > 0 THEN LN = LN + 1
220  IF LN = 2 THEN 1000
230  IF LN = 0 THEN 2000
240  INPUT "ANGOLO=";A1
250  IF A1 =  > 90 THEN 3000
260  IF A1 = 0 THEN 2000
270  GOTO 1000
```

II) Calcolo degli elementi incogniti. Ora che abbiamo memorizzato gli elementi noti, dobbiamo determinare quelli incogniti. Come sappiamo, possiamo distinguere due casi:

1) è noto un solo lato (e quindi anche un angolo oltre a quello retto);
2) sono noti due lati (e quindi solo l'angolo retto).

Inoltre, in ciascuno dei due casi possiamo distinguere due sottocasi:

A) fra i lati noti c'è l'ipotenusa;
B) fra i lati noti non c'è l'ipotenusa.

Abbiamo dunque 4 casi, che chiameremo 1A, 1B, 2A, 2B.

Ragionando come nel punto I), iniziamo con l'elencare i vari passi da compiere:

— distinguere fra i quattro casi;
— in ciascuno dei casi, calcolare gli elementi incogniti.

Per calcolare gli elementi incogniti, dobbiamo tenere presente il fatto che, a differenza del calcolatore tascabile, il calcolatore programmabile ha una sola funzione inversa: l'inversa della tangente, inv tan, che si scrive ATN. Dovremo dunque scegliere un metodo di risoluzione che si basi solo su questa funzione.

Ecco allora come procedere (Fig. 1):

Caso 1A (sono noti l'ipotenusa a e l'angolo β)
— si calcola il terzo angolo:

$$\gamma=90°-\beta$$

— si calcolano i cateti:

$$b=a\,\text{sen}\,\beta$$
$$c=a\cos\beta$$

Caso 1B (sono noti il cateto b e l'angolo β)
— si calcola il terzo angolo:

$$\gamma=90°-\beta$$

— si calcolano i restanti lati:

$$a=\frac{b}{\text{sen}\,\beta}$$
$$c=\frac{b}{\text{tg}\,\beta}$$

Caso 2A (sono noti l'ipotenusa a e il cateto b)
— si calcola l'altro cateto:

$$c=\sqrt{a^2-b^2}$$

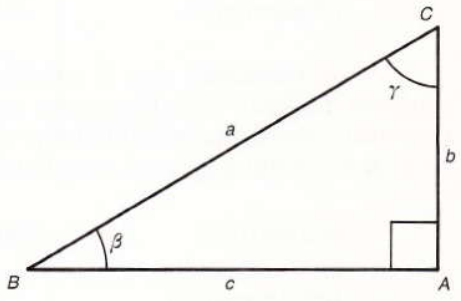

Fig. 1

— si calcolano gli angoli:

$$\beta=\text{ATN}\left(\frac{b}{c}\right)$$

$$\gamma=90°-\beta$$

Caso 2B (sono noti i cateti b e c)
— si calcola l'ipotenusa:

$$a=\sqrt{b^2+c^2}$$

— si calcolano gli angoli:

$$\beta=\text{ATN}\left(\frac{b}{c}\right)$$

$$\gamma=90°-\beta.$$

A questo punto siamo in grado di scrivere lo pseudocodice. Occorre però notare che le funzioni trigonometriche del BASIC (SIN, COS, TAN, ATN) usano un'unità di misura degli angoli che non è il grado ma il radiante (vedi cap. 2, n. 2); si passa dal grado al radiante moltiplicando per $\frac{\pi}{180}$. Dunque:

$$\text{misura in radianti}=\text{misura in gradi}\times\frac{\pi}{180}$$

$$\text{misura in gradi}=\text{misura in radianti}\times\frac{180}{\pi}.$$

Ecco l'esame dei quattro casi esposto con il nostro pseudocodice:

— if latinoti=1 and ipotenusa>0 then caso 1A
— if latinoti=1 and ipotenusa=0 then caso 1B
— if latinoti=2 and ipotenusa>0 then caso 2A
— if latinoti=2 and ipotenusa=0 then caso 2B

— *** caso 1A ***
  — secondoangolo=90−primoangolo
  — primocateto=ipotenusa×sen (primoangolo π/180)
  — secondocateto=ipotenusa×cos (primoangolo π/180)
  — scrivi i risultati

— *** caso 1B ***
  — secondoangolo=90−primoangolo
  — ipotenusa=primocateto/sen (primoangolo π/180)
  — secondocateto=primocateto/tg (primoangolo π/180)
  — scrivi i risultati

— *** caso 2A ***
  — secondocateto=$\sqrt{\text{ipotenusa}^2-\text{primocateto}^2}$
  — primoangolo=ATN (primocateto/secondocateto)×180/π
  — secondoangolo=90−primoangolo
  — scrivi i risultati

— *** caso 2B ***
  — ipotenusa=$\sqrt{\text{primocateto}^2+\text{secondocateto}^2}$
  — primoangolo=ATN (primocateto/secondocateto)×180/π
  — secondoangolo=90−primoangolo
  — scrivi i risultati

Esaminiamo ora questo pseudocodice alla ricerca di nuove variabili e nuove subroutine:

VARIABILE	SIMBOLO
secondoangolo	A2

SUBROUTINE	NUMERO DI LINEA
scrivi i risultati	1000

Siamo così in grado di tradurre lo pseudocodice in linguaggio BASIC:

```
1000  REM  CALCOLO ELEMENTI INCOGNITI**************
1010 PI = 3.141592653: REM  PI GRECO
1020  IF LN = 1 AND IP > 0 THEN 1060
1030  IF LN = 1 AND IP = 0 THEN 1110
1040  IF LN = 2 AND IP > 0 THEN 1160
1050  IF LN = 2 AND IP = 0 THEN 1210
1060  REM  CASO 1A
1070 A2 = 90 - A1
1080 C1 = IP *  SIN (A1 * PI / 180)
1090 C2 = IP *  COS (A1 * PI / 180)
1100  GOTO 4000
1110  REM  CASO 1B
1120 A2 = 90 - A1
1130 IP = C1 /  SIN (A1 * PI / 180)
1140 C2 = C1 /  TAN (A1 * PI / 180)
1150  GOTO 4000
1160  REM  CASO 2A
1170 C2 =  SQR (IP ^ 2 - C1 ^ 2)
1180 A1 =  ATN (C1 / C2) * 180 / PI
1190 A2 = 90 - A1
1200  GOTO 4000
1210  REM  CASO 2B
1220 IP =  SQR (C1 ^ 2 + C2 ^ 2)
1230 A1 =  ATN (C1 / C2) * 180 / PI
1240 A2 = 90 - A1
1250  GOTO 4000
```

III) Stampa dei risultati. Per completare il programma, ci restano da scrivere le istruzioni di stampa dei risultati. Si possono presentare tre eventualità:

— il triangolo è indeterminato (subroutine alla linea 2000);
— il triangolo è impossibile (subroutine alla linea 3000);
— il triangolo è stato risolto (subroutine alla linea 4000).

In tutti i casi si tratta di operazioni molto semplici, che possiamo scrivere direttamente in BASIC:

```
2000  REM  TRIANGOLO INDETERMINATO*****************
2010  PRINT "TRIANGOLO INDETERMINATO!"
2020  END
3000  REM  TRIANGOLO IMPOSSIBILE*******************
3010  PRINT "TRIANGOLO IMPOSSIBILE!"
3020  END
4000  REM  TRIANGOLO RISOLTO**********************
4010  PRINT "IPOTENUSA:";IP
4020  PRINT "CATETI:";C1;"  ";C2
4030  PRINT "ANGOLI:";A1;"  ";A2
4040  END
```

Il programma è così completato.

2. Un'osservazione sulla programmazione

Riflettiamo sul procedimento seguito per scrivere il programma del paragrafo precedente. Abbiamo iniziato con una breve descrizione: il programma doveva "risolvere i triangoli rettangoli". Abbiamo poi elencato i passi più importanti:

— accettare i valori degli elementi noti;
— calcolare gli elementi incogniti;
— mostrare il risultato.

Abbiamo quindi analizzato ciascuna di queste fasi, decomponendole in una serie di passi via via più semplici, fino a raggiungere un livello elementare in cui ad ogni frase potesse

corrispondere una linea di programma; abbiamo tradotto queste frasi in pseudocodice e, a partire da questo, abbiamo scritto il programma.

Questo cammino è rappresentato in Fig. 2: una serie di decomposizioni successive con un livello di dettaglio sempre maggiore, fino allo pseudocodice e al programma.

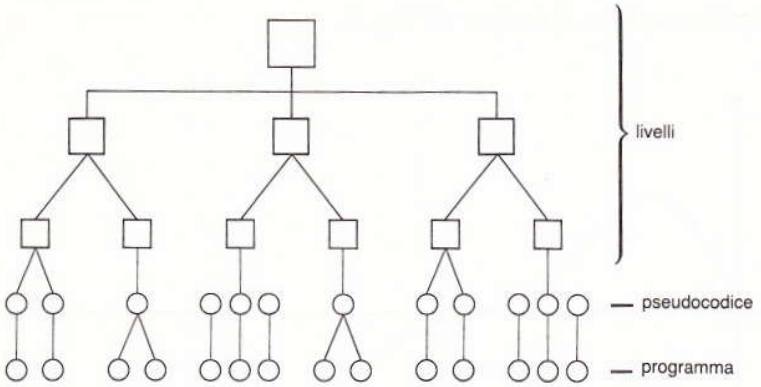

Fig. 2

Questa tecnica di programmazione è detta "top-down", cioè dall'alto in basso, ed è la più diffusa. Permette di evitare molti errori e di procedere in maniera sistematica; il tempo che si perde nello scrivere in dettaglio i vari livelli e lo pseudocodice è ampiamente ripagato dal tempo che si risparmia nel "debugging", cioè nella caccia agli errori. L'esperienza di programmazione di oltre 20 anni ha infatti dimostrato che circa metà del tempo necessario a completare un programma è perso nel debugging; perciò è importante ridurre gli errori fin dall'inizio.

Tutti i programmi che vi presenteremo nei prossimi paragrafi saranno sviluppati col metodo top-down; le spiegazioni saranno via via ridotte mentre rimarranno le liste dei passi da compiere, lo pseudocodice e, naturalmente, il programma.

3. Somma di armoniche

Come abbiamo detto nel cap. 4 (p. 98), una funzione periodica di frequenza f si può sempre esprimere come somma di sinusoidi aventi frequenze f, $2f$, $3f$...; queste sinusoidi sono spesso dette **armoniche**: prima armonica (o fondamentale), seconda armonica, terza armonica... Sempre nel cap. 4, abbiamo visto l'esempio dell'onda triangolare (Fig. 3), le cui armoniche sono:

$$y_1 = \operatorname{sen} \omega t$$

$$y_2 = \frac{1}{2} \operatorname{sen} 2\omega t$$

$$y_3 = \frac{1}{3} \operatorname{sen} 3\omega t$$

$$\vdots \qquad \vdots$$

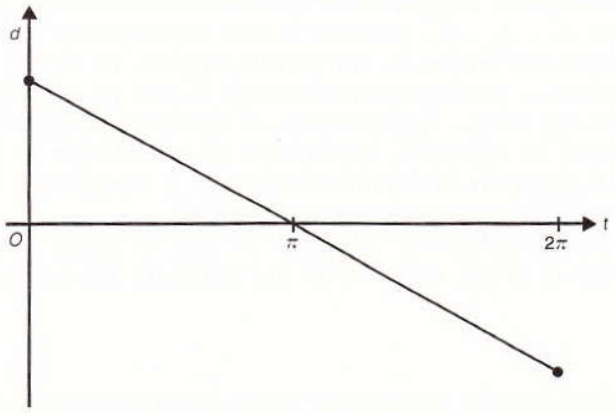

Fig. 3

Vogliamo ora scrivere un programma che ci permetta di vedere sullo schermo la somma di più funzioni armoniche. Ad esempio, vogliamo visualizzare la somma di y_1 e y_2:

$$\text{sen } \omega t + \frac{1}{2} \text{sen } 2\omega t;$$

vogliamo cioè che sul video appaia il grafico di Fig. 4.

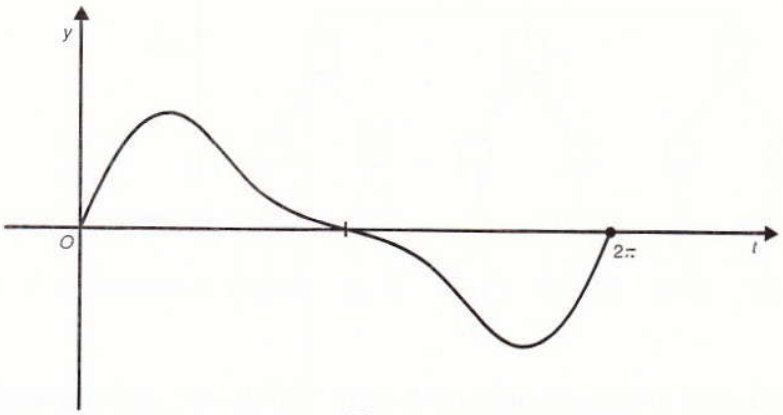

Fig. 4

Cerchiamo di essere più precisi. Scelta una pulsazione ω, per esempio $\omega=1$, la prima armonica da considerare è:

$$y_1 = A_1 \text{ sen } (t + \varphi_1)$$

dove A_1 è l'ampiezza e φ_1 la fase. La seconda armonica è:

$$y_2 = A_2 \text{ sen } (2t + \varphi_2)$$

dove l'ampiezza A_2 e la fase φ_2 sono, in genere, diverse da A_1 e φ_1.
L'armonica i-esima sarà dunque:

$$y_i = A_i \text{ sen } (it + \varphi_i)^{(1)}$$

e per determinarla occorrono due numeri: l'ampiezza A_i e la fase φ_i. È allora chiaro che il nostro programma deve accettare in ingresso, uno dopo l'altro, i valori delle ampiezze e delle fasi:

$$\begin{array}{ll} A_1 & \varphi_1 \\ A_2 & \varphi_2 \\ \vdots & \vdots \end{array}$$

È chiaro che il programma deve conservare in memoria questi dati, immagazzinandoli via via che li battiamo sulla tastiera.

Possiamo decidere di memorizzare questi dati sotto forma di **variabili con indice**; per esempio, le ampiezze A_1, A_2, A_3 possono essere memorizzate come A(1), A(2), A(3). In questo modo costruiamo un **vettore** o, con parola inglese, un **array**: l'array A(I).

Allo stesso modo, possiamo memorizzare le fasi in un altro array, che chiameremo F(I). Avremo dunque due array, A(I) ed F(I), di cui dovremo stabilire a priori le **dimensioni**, cioè il numero massimo di elementi. Decidiamo di considerare al massimo la somma di 30 armoniche e dunque fissiamo in 30 il numero massimo di elementi in ciascun array; questo vuol dire che l'indice I non dovrà mai superare il valore 30.

Dobbiamo poi decidere se misurare le fasi in gradi o in radianti; scegliamo di introdurle dalla tastiera in gradi, che ci sono più familiari. Ma poiché la funzione SIN accetta

(1) È ovvio che la lettera i che compare nella formula non ha alcuna relazione con l'unità immaginaria.

solo argomenti in radianti, dovremo passare da gradi a radianti moltiplicando per un fattore di conversione FC dato, come sappiamo, da

$$\frac{\pi}{180}.$$

Abbiamo così definito le strutture di dati di cui avremo bisogno nel programma; si tratta di passi molto semplici, che possiamo scrivere direttamente in BASIC senza passare per lo pseudocodice:

```
100  REM  STRUTTURE DATI
110 NM = 30: REM  NUMERO MASSIMO DI ARMONICHE
120  DIM A(NM),F(NM): REM  ARRAY DEI MODULI E DELLE FASI
130 PI = 4 *  ATN (1): REM  PI GRECO
140 FC = PI / 180: REM  FATTORE DI CONVERSIONE DA GRADI A RADIANTI
150 XM = 279:YM = 159: REM  VALORI MASSIMI DI X E DI Y
160 Y0 =  INT (YM / 2): REM  TRASLAZIONE ASSE DELLE ASCISSE
```

Veniamo ora alla parte di programma che chiede i dati e li memorizza. Le operazioni da compiere sono:

— indicare il numero dell'armonica di cui si chiedono i dati;
— verificare che questo numero non superi il numero massimo previsto;
— chiedere il valore dell'ampiezza;
— chiedere il valore della fase in gradi;
— convertire questo valore in radianti;
— chiedere se si vuole il grafico o se si debbono inserire altri dati.

Anche in questo caso, possiamo scrivere direttamente le istruzioni in BASIC:

```
200  REM  INSERIMENTO DATI
210  HOME : VTAB 21
220 I = I + 1: REM  I E' L'INDICE DELL'ARMONICA
230  IF I > NM THEN  PRINT "TROPPE ARMONICHE": END
240  HTAB 13: INVERSE : PRINT "ARMONICA N.";I: NORMAL
250  INPUT "AMPIEZZA:";A(I)
260 A = A + A(I): REM  AMPIEZZA MASSIMA
270  INPUT "FASE     :";G
280 F(I) = G * FC: REM  CONVERTE LA FASE IN RADIANTI
290  INVERSE : PRINT " X=FINE   N=NON DISEGNA  ALTRO=DISEGNA ": NORMAL
300  GET K$
310  IF K$ = "X" THEN  END
320  IF K$ = "N" THEN 200
```

Veniamo ora alla parte grafica. Dovremo:

— disegnare gli assi;
— calcolare i punti della curva;
— disegnare questi punti.

Iniziando dal disegno degli assi, osserviamo che nella maggior parte dei calcolatori gli assi sono disposti come nella Fig. 5; se li lasciassimo in questa posizione, apparirebbe sul video soltanto la parte della curva compresa nel primo quadrante e, per giunta, ribaltata rispetto all'usuale. Conviene perciò spostare gli assi disponendoli come in Fig. 6.

Osserviamo inoltre che l'ascissa può assumere solo valori compresi fra zero e un numero massimo che chiameremo XM (Fig. 5); analogamente, anche l'ordinata può variare solo da zero ad YM. Perciò, il nuovo asse delle ascisse (Fig. 6) dovrebbe avere equazione

$y=y_0$, cioè $y=\mathrm{YM}/2$.

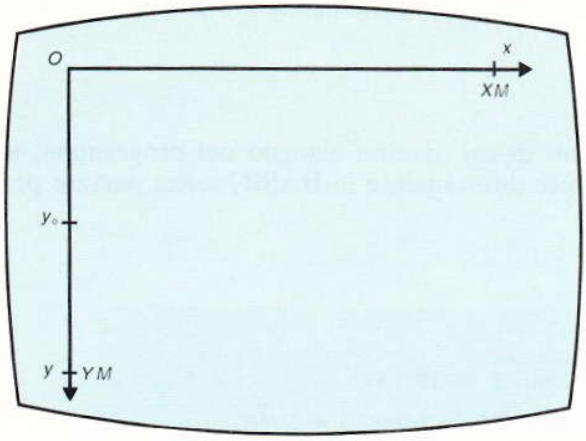

Fig. 5

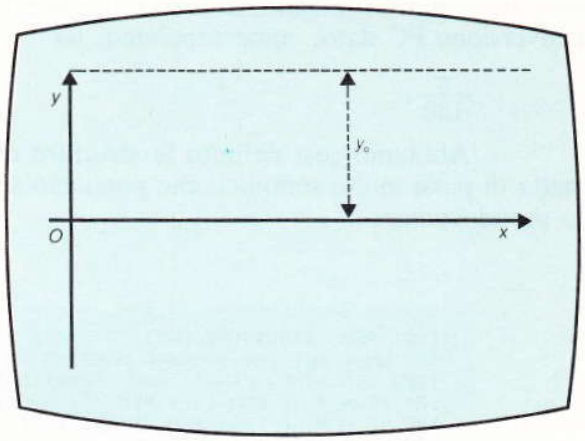

Fig. 6

Ma poiché il calcolatore disegna sullo schermo solo punti aventi coordinate intere, chiameremo y_0 la parte intera di YM/2 e scriveremo:

y_0=parte intera di YM/2.

Così, y_0 indica la traslazione subita dall'asse delle ascisse (Fig. 6).

Ecco allora la subroutine che disegna gli assi:

```
400  REM  TRACCIAMENTO DEGLI ASSI
410  HGR
420  HPLOT 0,Y0 TO XM,Y0: REM  ASSE X
430  HPLOT 0,0 TO 0,YM: REM   ASSE Y
```

Prima di disegnare la curva, dobbiamo tener presente il cambiamento di riferimento che abbiamo appena fatto: abbiamo invertito l'orientazione dell'asse delle y, cambiando quindi il segno delle ordinate, ed abbiamo eseguito una traslazione di y_0, sommando quindi y_0 a tutte le ordinate. Pertanto, abbiamo trasformato le ordinate y ottenendo ordinate y' date dalla formula:

$$y'=y_0-y.$$

Dobbiamo ancora scegliere le unità di misura sui due assi coordinati. Per le ascisse ragioniamo così: visto che x varia da 0 a XM, decidiamo che il segmento di lunghezza XM corrisponda a un'intero periodo T della prima armonica (Fig. 7). In questo modo avremo sul video esattamente un periodo completo della nostra curva.
Abbiamo dunque:

$$T=\text{XM}$$

e ricordando che risulta

$$\omega=\frac{2\pi}{T},$$

otteniamo:

$$\omega=2\pi/\text{XM}.$$

Pertanto, la prima armonica si scrive:

$$y_1=A_1\,\text{sen}\left(\frac{2\pi}{\text{XM}}t+\varphi_1\right)$$

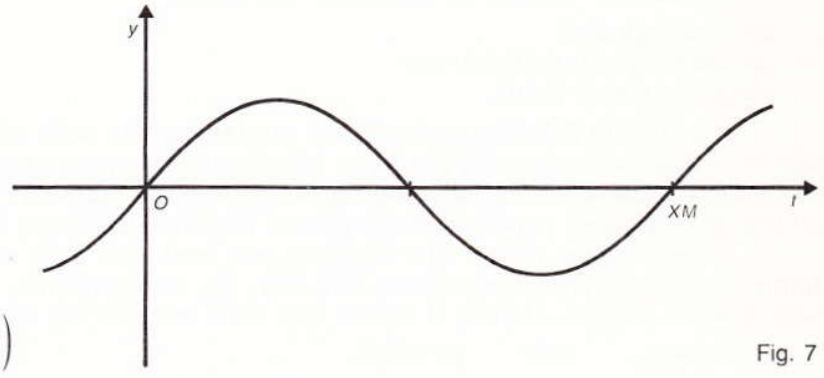

Fig. 7

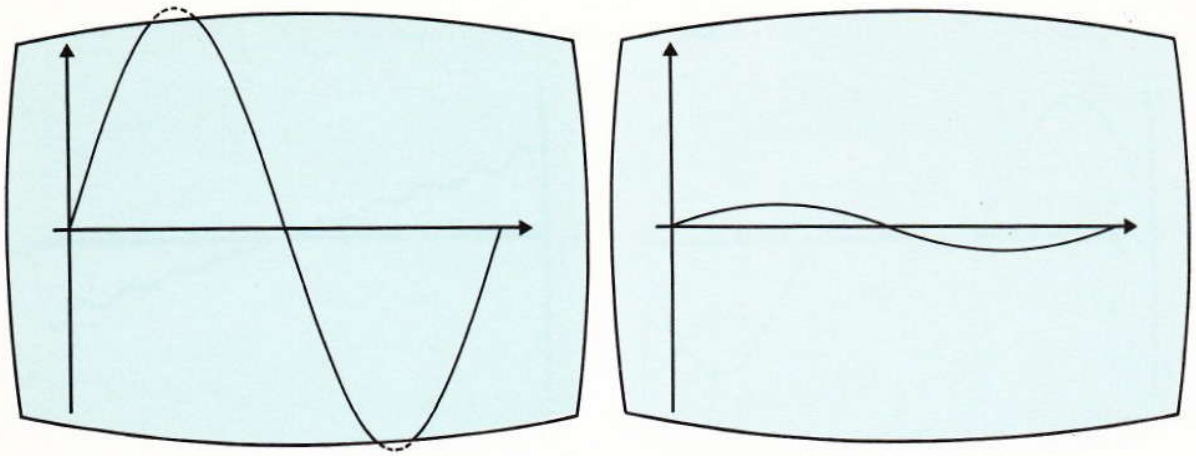

Fig. 8

e l'i-esima armonica diventa:

$$y_i = A_i \operatorname{sen}\left(i\frac{2\pi}{\mathrm{XM}}t+\varphi_1\right).$$

Venendo ora alle ordinate, il problema che si pone è di scegliere l'unità di misura in modo che il grafico non esca dallo schermo ma non sia neppure troppo appiattito (Fig. 8). Possiamo ragionare così: l'ampiezza massima di ciascuna armonica non supera mai il valore A(I) e dunque la somma di più armoniche non supera mai la somma delle corrispondenti ampiezze massime. Calcoliamo allora questa somma:

$$\mathrm{A}=\mathrm{A}(1)+\mathrm{A}(2)+\mathrm{A}(3)+\ldots$$

e scegliamo la scala delle ordinate in modo che al valore A corrisponda il massimo valore di y che entra nel video. Poiché questo massimo valore è y_0, basterà moltiplicare le ordinate y per il fattore y_0/A; si ottiene così

$$y' = y_0 - y \cdot y_0/\mathrm{A}.$$

In questo modo, se y dovesse raggiungere il suo valore massimo A, la y' raggiungerebbe il valore 0 (estremità superiore dello schermo in Fig. 5); se invece y dovesse raggiungere il suo valore minimo $-\mathrm{A}$, la y' raggiungerebbe il valore $2y_0$ (estremità inferiore dello schermo).

Visto che le operazioni da compiere non sono elementari, conviene procedere con cautela scrivendo una lista:

— calcolare la somma delle ampiezze A;
— per x che varia da 0 fino a XM:
  - calcolare y, somma delle armoniche;
  - calcolare $y' = y_0 - y \cdot y_0/\mathrm{A}$;
  - disegnare il punto di coordinate x,y';
— tornare alla fase di inserimento dati.

Le operazioni da compiere sono ora ben chiare e possiamo scriverle direttamente in BASIC:

```
500  REM  TRACCIAMENTO DEL GRAFICO
510  FOR X = 0 TO XM
520  REM   SOMMA DELLE PRIME I ARMONICHE
530 Y = 0
540  FOR J = 1 TO I
550 Y = Y + A(J) *  SIN (J * 2 * PI * X / XM + F(J))
560  NEXT J
570 Y = Y0 - Y * Y0 / A
580  HPLOT X,Y
590  NEXT X
600  GOTO 200
610  REM   ******************************************************
```

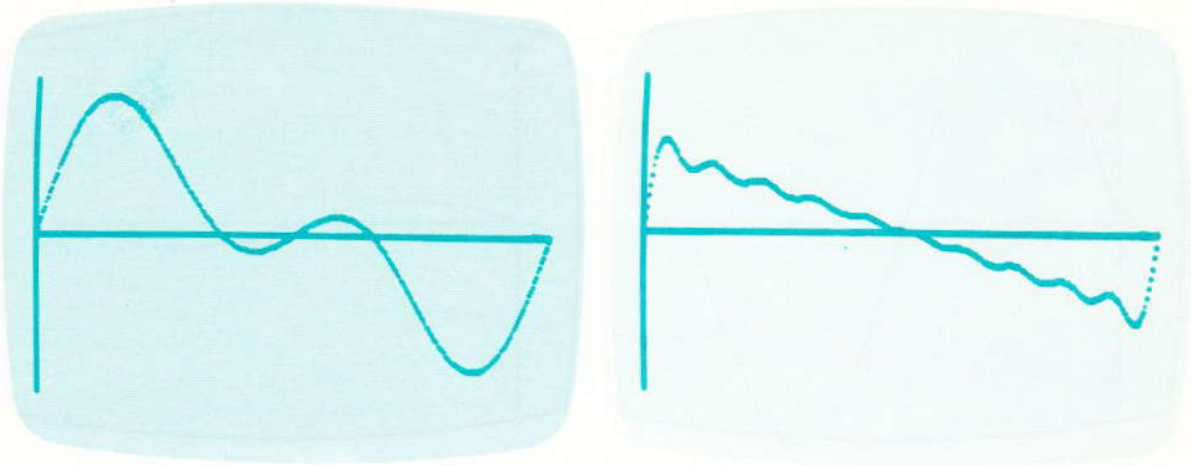

Fig. 9. Somma di armoniche ottenuta con il programma presentato in queste pagine: a sinistra le prime 3 armoniche, a destra le prime 11 della serie.

Il programma è così completato. Prima di copiarlo sul vostro calcolatore, dovete fare attenzione alle istruzioni non standard del BASIC, in particolare alle istruzioni grafiche che sono diverse da macchina a macchina.

La Fig. 9 mostra due grafici ottenuti con questo programma.

4. Risoluzione di equazioni trigonometriche

Sulla base di quanto detto nel cap. 5, consideriamo due tipi di equazioni trigonometriche:

1) $a \operatorname{sen} x + b \cos x = c$
2) $a \operatorname{sen}^2 x + b \operatorname{sen} x \cos x + c \cos^2 x = d.$

Ricordiamo brevemente come si risolvono queste equazioni, iniziando dalla prima. Ponendo

$$r = \sqrt{a^2+b^2} \qquad \varphi = \operatorname{arctg} \frac{b}{a}$$

si ha:

$$a \operatorname{sen} x + b \cos x = r \operatorname{sen} (x+\varphi);$$

pertanto, l'equazione 1) si riduce a:

$$r \operatorname{sen} (x+\varphi) = c.$$

Questa equazione è risolubile se e solo se

$$-1 \leq \frac{c}{r} \leq 1$$

cioè, tenendo presente che $r>0$, se e solo se:

$$|c| \leq r.$$

In tal caso risulta:

$$x+\varphi = \operatorname{arcsen} \frac{c}{r}$$

e infine:

$$x = \operatorname{arcsen} \frac{c}{r} - \varphi.$$

Ricordiamo che questa è solo una delle soluzioni; **tutte** le soluzioni sono date da:

$$x_k = 2k\pi + \arcsen\frac{c}{r} - \varphi$$

$$x'_k = (2k+1)\pi - \arcsen\frac{c}{r} + \varphi.$$

Passiamo ora alla seconda equazione. Con le formule di duplicazione si ottiene:

$$a\,\mathrm{sen}^2 x + b\,\mathrm{sen}\,x\cos x + c\cos^2 x = \frac{a}{2}(1-\cos 2x) + \frac{b}{2}\mathrm{sen}\,2x + \frac{c}{2}(1+\cos 2x) =$$

$$= \frac{b}{2}\mathrm{sen}\,2x + \left(\frac{c}{2} - \frac{a}{2}\right)\cos 2x + \frac{a}{2} + \frac{c}{2}$$

e quindi l'equazione assegnata diventa:

$$b\,\mathrm{sen}\,2x + (c-a)\cos 2x = 2d - a - c.$$

Ponendo

$$A = b \qquad B = c-a \qquad C = 2d-a-c \qquad X = 2x$$

l'equazione diviene:

$$A\,\mathrm{sen}\,X + B\cos X = C,$$

che è della stessa forma della 1) e dunque può essere risolta come abbiamo già visto. Una volta trovati i valori di X, si ottengono quelli di x, dato che risulta:

$$x = \frac{X}{2}.$$

Chiarito così il procedimento da seguire, iniziamo lo studio del programma con un primo elenco di passi da compiere:

*** programma principale ***
— mostra sul video i due tipi di equazione;
— chiede qual è il tipo da risolvere;
— chiede i valori di a, b, c ed eventualmente di d;
— salta alla subroutine corrispondente al tipo di equazione scelta;
— stampa i risultati

*** subroutine che risolve la prima equazione ***
— calcola $r = \sqrt{a^2+b^2}$;
— verifica che l'equazione sia risolubile;
— calcola $\varphi = \mathrm{arctg}\frac{b}{a}$;
— calcola $x = \arcsen\frac{c}{r} - \varphi$
— ritorna

*** subroutine che risolve la seconda equazione ***
— pone $A = b$, $B = c-a$, $C = 2d-a-c$;
— salta alla subroutine che risolve la prima equazione;
— pone $x = \frac{X}{2}$;
— ritorna

A partire da questo elenco, scriviamo uno pseudocodice:

*** programma principale ***
— print equazione 1
— print equazione 2
— input scelta
— input a, b, c
— if scelta=2 then input d
— if scelta=1 then prima equazione

— if scelta=2 then seconda equazione
— mostra i risultati

*** prima equazione ***
— $r=\sqrt{a^2+b^2}$
— if $|c|>r$ then impossibile
— $\varphi=\text{arctg}\frac{b}{a}$
— $x=\text{arcsen}\frac{c}{r}-\varphi$
— ritorna

*** seconda equazione ***
— A=b, B=$c-a$, C=$2d-a-c$
— a=A, b=B, c=C
— gosub primaequazione
— $x=\frac{X}{2}$
— ritorna

*** impossibile ***
— print equazione impossibile
— end

Ecco, infine, il programma in linguaggio BASIC:

```
100  HOME
110  REM  *** PROGRAMMA PRINCIPALE ****************
120  PRINT "1: asenx + bcosx = c"
130  PRINT
140  PRINT "       2                      2"
150  PRINT "2: asen x + bsenxcosx + ccos x = d"
160  PRINT
170  INPUT "QUALE EQUAZIONE ? (1 O 2)";SC
180  PRINT
190  INPUT "a=";A
200  INPUT "b=";B
210  INPUT "c=";C
220  IF SC = 2 THEN  INPUT "d=";D
230  IF SC = 1 THEN  GOSUB 1000
240  IF SC = 2 THEN  GOSUB 2000
244 PI = 3.141592653
245  PRINT
246 X = X * 180 / PI
250  PRINT "UNA SOLUZIONE E': ";X;" GRADI"
260  END
1000  REM  *** PRIMA EQUAZIONE ********************
1010 R =  SQR (A ^ 2 + B ^ 2)
1012  IF  ABS (R) <  =  ABS (C) THEN 3000
1015  IF A = 0 THEN FI = PI / 2: GOTO 1030
1020 FI =  ATN (B / A)
1030 X =  ATN ((C / R) /  SQR (1 - (C / R) ^ 2)) - FI
1040  RETURN
2000  REM  *** SECONDA EQUAZIONE ******************
2010 A1 = B:B1 = C - A:C1 = 2 * D - A - C
2020 A = A1:B = B1:C = C1
2030  GOSUB 1000
2040 X = X / 2
2050  RETURN
3000  REM  *** EQUAZIONE IMPOSSIBILE **************
3010  PRINT "EQUAZIONE IMPOSSIBILE!"
3020  END
```

Il programma ha un difetto: se scegliamo l'equazione 2) e poniamo a=1, b=0, c=1, d=1 otteniamo un'identità, cioè un'eguaglianza vera per ogni valore di x... e il calcolatore non se ne accorge. Siete capaci di modificare il programma in modo da tener conto anche di questa eventualità?

5. Risoluzione di triangoli qualunque

L'ultimo programma che vi proponiamo risolve i triangoli di qualunque tipo. È un programma lungo e impegnativo, che non vi consigliamo di affrontare prima di aver ben compreso quelli dei paragrafi precedenti; ma è anche un programma allettante, perché consente di risolvere all'istante un gran numero di problemi.

Come abbiamo visto nel cap. 2, nella risoluzione dei triangoli si possono presentare quattro casi diversi da risolvere, a seconda di quali siano gli elementi noti; il programma si divide in sottoprogrammi che affrontano e risolvono ciascuno di questi casi. Più in particolare, il programma è costituito dalle seguenti parti:

— intestazione;
— inserimento dati;
— verifica dei lati;
— riconoscimento del caso da risolvere;
— soluzione del primo caso (sono dati due lati e l'angolo compreso);
— soluzione del secondo caso (sono dati due angoli ed un lato);
— soluzione del terzo caso (sono dati tre lati);
— soluzione del quarto caso (sono dati due lati e l'angolo non compreso);
— stampa dei risultati.

Esaminiamo le varie parti, con la solita tecnica top-down.

Intestazione. Questa parte del programma serve a far comparire sul video un messaggio che informa l'utente circa lo scopo del programma. È chiaro che non si tratta di una parte necessaria e perciò potete anche saltarla; tuttavia dà un'aria professionale a tutto il programma.

L'intestazione è come la copertina di un libro e il modo in cui è progettata è questione di gusto; quella che noi vi presentiamo è presentata in Fig. 10 e viene ottenuta con le seguenti istruzioni:

```
100  REM  INTESTAZIONE
110  HOME : PRINT  CHR$ (7)
120  INVERSE : PRINT  TAB( 40);"": NORMAL
130  VTAB 9
140  PRINT  TAB( 13);"**************"
150  PRINT  TAB( 13);"*              *"
160  PRINT  TAB( 13);"* SOLUZIONE *"
170  PRINT  TAB( 13);"*     DEI      *"
180  PRINT  TAB( 13);"* TRIANGOLI *"
190  PRINT  TAB( 13);"*              *"
200  PRINT  TAB( 13);"**************"
210  VTAB 22
220  INVERSE : PRINT  TAB( 40);"": NORMAL
```

```
**************
*              *
* SOLUZIONE *
*     DEI      *
* TRIANGOLI *
*              *
**************
```

Fig. 10

Inserimento dati. Questo sottoprogramma chiede i valori dei lati noti, sceglie l'unità di misura degli angoli e chiede il valore degli angoli noti. Viene anche tenuto il conto del numero di lati noti e del numero di angoli noti, verificando che il triangolo sia determinato.

Ecco l'elenco delle operazioni:

*** inserimento dati ***
— scrivere "inserimento dati" e "premere 0 per gli elementi incogniti";
— chiedere il valore dei lati A, B, C;
— contare quanti sono i lati noti;
— se i lati noti sono 0 allora il triangolo è indeterminato;
— se i lati noti sono 3 allora verificare che il triangolo esista;
— chiedere se si scelgono i gradi o i radianti;
— chiedere il valore degli angoli ALFA e BETA;
— se ALFA e BETA sono diversi da 0 allora calcolare GAMMA;
— se ALFA e BETA non sono entrambi diversi da 0 allora chiedere GAMMA;
— contare quanti sono gli angoli noti;
— se lati noti+angoli noti<3 allora il triangolo è indeterminato.

A partire da questo elenco, è facile scrivere il programma in linguaggio BASIC:

```
230  DIM A2(2),A3(2),TL(2)
240 PI = 3.141592653: REM  PI GRECO
250  DEF  FN AS(X) =  ATN (X /  SQR (1 - X * X))
260  DEF  FN AC(X) = PI / 2 -  ATN (X /  SQR (1 - X * X))
270  GET K$: IF K$ = "" THEN 150
400  REM ************************************************
410  REM  INSERIMENTO DATI
420  HOME
430  HTAB 11
440  INVERSE : PRINT " INSERIMENTO DATI ": NORMAL
450  VTAB 3: PRINT "   INTRODURRE I VALORI NOTI; PREMERE"
460  PRINT "   ZERO  PER GLI ELEMENTI  INCOGNITI"
470  VTAB 6
480  HTAB 8: PRINT "******* L A T I *******"
490  PRINT : PRINT
500  INPUT "LATO A :";A
510 A =  ABS (A)
520  IF A > 0 THEN NL = NL + 1
530  PRINT
540  INPUT "LATO B: ";B
550 B =  ABS (B)
560  IF B > 0 THEN NL = NL + 1
570  PRINT
580  INPUT "LATO C: ";C
590 C =  ABS (C)
600  IF C > 0 THEN NL = NL + 1
610  PRINT : PRINT
620  IF NL = 0 THEN  PRINT "SENZA LA MISURA DI ALMENO UN LATO": PRINT "IL
      TRIANGOLO E' INDETERMINATO": END
630  IF NL = 3 THEN  GOSUB 6000: REM  VERIFICA LATI
640  INPUT "ANGOLI: IN GRADI (G) O RADIANTI (R) ?";A$
650  IF A$ = "G" THEN B$ = " GRADI"
660  IF A$ = "G" THEN FC = PI / 180
670  IF A$ = "R" THEN B$ = "RADIANTI"
680  IF A$ = "R" THEN FC = 1
690  VTAB 15: PRINT  CHR$ (11)
700  PRINT  TAB( 3);"******* ANGOLI IN ";B$;" *******
710  PRINT
720  INPUT "ANGOLO ALFA  (OPPOSTO AL LATO A):";ALFA
730 ALFA =  ABS (ALFA) * FC
740  IF ALFA > 0 THEN NA = NA + 1
750  PRINT
760  INPUT "ANGOLO BETA  (OPPOSTO AL LATO B):";BETA
770 BETA =  ABS (BETA) * FC
780  IF BETA > 0 THEN NA = NA + 1
790  PRINT
800  IF NA = 2 THEN GAMMA = PI - ALFA - BETA
810  IF NA = 2 THEN  PRINT "ANGOLO GAMMA (OPPOSTO AL LATO C):";GAMMA / FC
     : GOTO 840
820  INPUT "ANGOLO GAMMA (OPPOSTO AL LATO C):";GAMMA
830 GAMMA =  ABS (GAMMA) * FC
840  IF GAMMA > 0 THEN NA = NA + 1
850  IF NL + NA < 3 THEN  PRINT : PRINT "TRIANGOLO INDETERMINATO": END
860  HOME
900  REM ************************************************
```

Verifica dei lati. Nel caso in cui siano noti tutti e tre i lati, si deve verificare che la somma di due lati sia sempre maggiore del terzo; se questo non avviene, il triangolo è impossibile.

La verifica può essere scritta subito in BASIC:

```
6000  REM ************************************************
6010  REM  VERIFICA DEI LATI
6020  REM
6030  IF A + B <  = C THEN 6070
6040  IF B + C <  = (A) THEN 6070
6050  IF A + C <  = B THEN 6070
6060  RETURN
6070  HOME
6080  VTAB 12: HTAB 8
6090  INVERSE : PRINT " TRIANGOLO IMPOSSIBILE ": NORMAL
6100  END
7000  REM ************************************************
7010  REM  INCONGRUENZA FRA LATI ED ANGOLI
7020  PRINT "ERRORE: L'ANGOLO ";AE$;" NON E'"
7030  PRINT "CONGRUENTE CON I LATI ASSEGNATI"
7040  PRINT "E VIENE IGNORATO."
7050  RETURN
```

Riconoscimento del caso da risolvere. Osserviamo che, se siano arrivati a questo punto, il triangolo è certamente determinato, cioè sono noti almeno tre elementi fra cui un lato. Perciò, si possono verificare le seguenti possibilità:

— è noto un lato; allora sono noti anche due angoli e quindi ricadiamo nel secondo caso;
— sono noti due lati; allora è noto anche un angolo e quindi ricadiamo nel primo o nel quarto caso;
— sono noti tre lati; allora è il terzo caso.

Vediamo così che per distinguere fra queste alternative basta conoscere il numero dei lati noti, NL. Per scrivere il programma sarà sufficiente decidere i numeri di linea delle subroutine che risolveranno i vari casi:

SUBROUTINE	NUMERO DI LINEA
primo caso e quarto caso	1000
secondo caso	2000
terzo caso	3000

Ecco le istruzioni BASIC:

```
900  REM ************************************************
910  REM  SCELTA DEL METODO DI SOLUZIONE
920  IF NL = 3 THEN  GOSUB 3000: GOSUB 5000: GOTO 950
930  IF NA = 3 THEN  GOSUB 2000: GOSUB 5000: GOTO 950
940  IF NL = 2 THEN  GOSUB 1000: GOSUB 5000: GOTO 950
950  HTAB 4: INVERSE : PRINT " NON ESISTONO ALTRE SOLUZIONI. ": NORMAL
960  END
```

Soluzione del primo caso. Sono noti due lati e un angolo; per essere sicuri che siamo in presenza del primo caso, dovremo verificare che l'angolo sia compreso fra i lati noti. Se questo avviene, si calcola il terzo lato con il teorema del coseno e il secondo angolo con il teorema dei seni; il terzo angolo si ottiene come differenza dei primi due rispetto a π (o 180°). Se invece l'angolo noto non è compreso fra i lati noti, ricadiamo nel quarto caso, che verrà risolto nella subroutine alla linea 4000; decidiamo perciò:

SUBROUTINE	NUMERO DI LINEA
quarto caso	4000

Nello scrivere il programma che risolve il primo caso, dobbiamo distinguere fra tre sottocasi possibili:

— sono noti i lati A, B e l'angolo compreso GAMMA;
— sono noti i lati A, C e l'angolo compreso BETA;
— sono noti i lati B, C e l'angolo compreso ALFA.

Ecco il programma, scritto direttamente in linguaggio BASIC.

```
1000  REM ************************************************
1010  REM  DUE LATI E UN ANGOLO SONO NOTI
1020  REM  L'ANGOLO E' COMPRESO? (I CASO)
1030  IF A = 0 THEN 1100
1040  IF B = 0 THEN 1150
1050  IF GAMMA = 0 THEN 4030: REM  QUARTO CASO
1060 C =  SQR (A * A + B * B - 2 * A * B *  COS (GAMMA))
1070 ALFA =  FN AS( SIN (GAMMA) * A / C)
1080 BETA = PI - ALFA - GAMMA
1090  RETURN
1100  IF ALFA = 0 THEN 4250: REM  QUARTO CASO
1110 A =  SQR (B * B + C * C - 2 * B * C *  COS (ALFA))
1120 BETA =  FN AS( SIN (ALFA) * B / A)
1130 GAMMA = PI - ALFA - BETA
1140  RETURN
1150  IF BETA = 0 THEN 4470: REM  QUARTO CASO
1160 B =  SQR (A * A + C * C - 2 * A * C *  COS (BETA))
1170 GAMMA =  FN AS( SIN (BETA) * C / B)
1180 ALFA = PI - BETA - GAMMA
1190  RETURN
```

Soluzione del secondo caso. Si ha il secondo caso quando sono noti un lato e due angoli; in tal caso è noto anche il terzo angolo, che viene già calcolato dal sottoprogramma d'inserimento dei dati. Restano perciò da calcolare gli altri due lati, che si trovano con il teorema dei seni.

Anche ora possono presentarsi tre sottocasi, secondo che il lato noto sia A, B o C. Ecco allora le istruzioni BASIC:

```
2000  REM ************************************************
2010  REM  UN LATO E DUE ANGOLI SONO NOTI (II CASO)
2020 NS = 1
2030  IF A <  > 0 THEN B = A *  SIN (BETA) /  SIN (ALFA):C = A *  SIN (GA
     MMA) /  SIN (ALFA): RETURN
2040  IF B <  > 0 THEN A = B *  SIN (ALFA) /  SIN (BETA):C = B *  SIN (GA
     MMA) /  SIN (BETA): RETURN
2050  IF C <  > 0 THEN A = C *  SIN (ALFA) /  SIN (GAMMA):B = C *  SIN (B
     ETA) /  SIN (GAMMA): RETURN
```

Soluzione del terzo caso. Si ha il terzo caso quando sono noti i tre lati; per calcolare gli angoli è allora sufficiente applicare il teorema del coseno.

Ecco le istruzioni BASIC:

```
3000  REM ************************************************
3010  REM  TRE LATI SONO NOTI (III CASO)
3020 NS = 1
3030 CALFA = (B ^ 2 + C ^ 2 - A ^ 2) / (2 * B * C)
3040 AA =  FN AC(CALFA)
3050  IF ALFA = 0 THEN 3080
3060  IF  ABS (ALFA - AA) < 0.0001 THEN 3080
3070 AE$ = "ALFA": GOSUB 7000
3080 ALFA = AA
3090 CBETA = (A ^ 2 + C ^ 2 - B ^ 2) / (2 * A * C)
3100 AB =  FN AC(CBETA)
3110  IF BETA = 0 THEN 3140
3120  IF  ABS (BETA - AB) < 0.0001 THEN 3140
3130 AE$ = "BETA": GOSUB 7000
3140 BETA = AB
3150 CGAMMA = (A ^ 2 + B ^ 2 - C ^ 2) / (2 * A * B)
3160 AC =  FN AC(CGAMMA)
3170  IF GAMMA = 0 THEN 3200
3180  IF  ABS (GAMMA - AC) < 0.0001 THEN 3200
3190 AE$ = "GAMMA": GOSUB 7000
3200 GAMMA = AC
3210  RETURN
```

Soluzione del quarto caso. Si ha il quarto caso quando sono noti due lati e un angolo non compreso; come abbiamo visto nel cap. 2 (pp. 54-60), questo problema può avere una, due o nessuna soluzione. Per prima cosa, dobbiamo distinguere sei sottocasi possibili (Fig. 11):

— sono noti A, B e ALFA;
— sono noti A, B e BETA;
— sono noti A, C e ALFA;
— sono noti A, C e GAMMA;
— sono noti B, C e BETA;
— sono noti B, C e GAMMA.

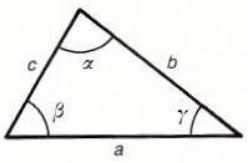

Fig. 11

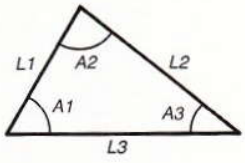

Fig. 12

È chiaro che dovremo scrivere delle istruzioni che permettano di riconoscere ciascuno di questi sei sottocasi; prima di farlo, però, riflettiamo sulla procedura da seguire per la risoluzione del triangolo. Per prima cosa, indichiamo (Fig. 12) con A1 l'angolo noto e con L1 quello fra i due lati noti che è adiacente all'angolo noto; indichiamo inoltre con L2 il secondo lato noto, quello cioè che non è adiacente all'angolo noto, con L3, A2 e A3 i restanti elementi. Ricordando quanto detto nel cap. 2, possono presentarsi le seguenti alternative:

— se L2<L1 sen A1, allora non esiste soluzione: il triangolo è impossibile;
— altrimenti, esiste almeno un triangolo, che si trova con il teorema dei seni;
— se poi A1<$\pi/2$ e L2<L1, allora esiste un altro triangolo in cui un angolo è il supplementare di quello della prima soluzione.

È chiaro che non conviene scrivere un'apposita parte di programma per risolvere ciascuno dei sei sottocasi che abbiamo visto prima; conviene scrivere una sola subroutine, in cui gli elementi noti sono L1, L2, A1 e quelli incogniti sono L3, A2, A3. Naturalmente, poi, dovremo restituire a ciascun elemento il suo nome originario, e cioè A, B, C per i lati e ALFA, BETA, GAMMA per gli angoli. Tutto ciò è piuttosto complicato in BASIC, mentre sarebbe molto più facile in linguaggi, come il Pascal, che permettono il «passaggio dei parametri» fra sottoprogrammi; in cambio, però, il BASIC è più semplice.

Ecco dunque (a pagina seguente) le istruzioni BASIC divise in due parti: la prima riconosce i sei sottocasi visti sopra, la seconda risolve il triangolo:

```
4000  REM ************************************************
4010  REM  DUE LATI E UN ANGOLO NON COMPRESO SONO NOTI (IV CASO)
4020  REM  SI DISTINGUONO SEI COMBINAZIONI POSSIBILI
4030  REM  A,B<>0,GAMMA=0
4040  IF BETA = 0 THEN 4150
4050  REM  A,B,BETA<>0
4060 LN = A:AN = BETA:SL = B
4070  GOSUB 4690
4080  FOR NS = 1 TO SOL
4090 ALFA = A2(NS)
4100 GAMMA = A3(NS)
4110 C = TL(NS)
4120  GOSUB 5000
4130  NEXT NS
4140  GOTO 950
4150  REM  A,B,ALFA<>0
4160 LN = B:AN = ALFA:SL = A
4170  GOSUB 4690
4180  FOR NS = 1 TO SOL
4190 BETA = A2(NS)
4200 GAMMA = A3(NS)
4210 C = TL(NS)
4220  GOSUB 5000
4230  NEXT NS
4240  GOTO 950
4250  REM  B,C<>0,ALFA=0
4260  IF BETA = 0 THEN 4370
4270  REM  B,C,GAMMA<>0
4280 LN = B:AN = GAMMA:SL = C
4290  GOSUB 4690
4300  FOR NS = 1 TO SOL
4310 BETA = A2(NS)
4320 ALFA = A3(NS)
4330 A = TL(NS)
4340  GOSUB 5000
4350  NEXT NS
4360  GOTO 950
4370  REM  B,C,BETA<>0
4380 LN = C:AN = BETA:SL = B
4390  GOSUB 4690
4400  FOR NS = 1 TO SOL
4410 GAMMA = A2(NS)
4420 ALFA = A3(NS)
4430 A = TL(NS)
4440  GOSUB 5000
4450  NEXT NS
4460  GOTO 950
4470  REM  A,C<>0,BETA=0
4480  IF ALFA = 0 THEN 4590
4490  REM  A,C,ALFA<>0
4500 LN = C:AN = ALFA:SL = A
4510  GOSUB 4690
4520  FOR NS = 1 TO SOL
4530 GAMMA = L2(NS)
4540 BETA = L3(NS)
4550 B = TL(NS)
4560  GOSUB 5000
4570  NEXT NS
4580  GOTO 950
4590  REM  A,C,GAMMA<>0
4600 LN = A:AN = GAMMA:SL = C
4610  GOSUB 4690
4620  FOR NS = 1 TO SOL
4630 ALFA = A2(NS)
4640 BETA = A3(NS)
4650 B = TL(NS)
4660  GOSUB 5000
4670  NEXT NS
4680  GOTO 950
4690  REM  SOLUZIONE QUARTO CASO*********
4700  REM   PARAMETRI:LN,AN,SL
4710  REM
4720  IF SL < LN *  SIN (AN) THEN 6070
4730 A2(1) =  FN AS( SIN (AN) * LN / SL)
4740 A3(1) = PI - AN - A2(1)
4750 TL(1) = SL *  SIN (A3(1)) /  SIN (AN)
4760  REM  ESISTE UN'ALTRA SOLUZIONE?
4770  IF AN < PI / 2 AND SL < LN THEN 4800
4780 SOL = 1
4790  RETURN
4800  REM  CALCOLO SECONDA SOLUZIONE
4810 SOL = 2
4820 A2(2) = PI - A2(1)
4830 A3(2) = PI - AN - A2(2)
4840 TL(2) = SL *  SIN (A3(2)) /  SIN (AN)
4850  RETURN
```

Stampa dei risultati. L'ultima subroutine esegue la stampa dei risultati: vengono scritti prima i lati, poi gli angoli, in gradi; se il numero di soluzioni NS è due, l'operazione viene ripetuta. Al termine, compare il messaggio «non esistono altre soluzioni».

```
5000  REM ************************************************
5010  REM  STAMPA DEI RISULTATI
5020  REM
5030  HTAB 12: INVERSE : PRINT " SOLUZIONE N.";NS;": ": NORMAL
5040  PRINT : PRINT "LATO A=";A
5050  PRINT "LATO B=";B
5060  PRINT "LATO C=";C
5070  PRINT
5080  PRINT "ANGOLO ALFA =";ALFA / FC
5090  PRINT "ANGOLO BETA =";BETA / FC
5100  PRINT "ANGOLO GAMMA=";GAMMA / FC
5110  PRINT
5120  RETURN
```

In Fig. 13 vedete il programma all'opera: sopra la fase d'inserimento dei dati, sotto i risultati.

```
          INSERIMENTO DATI
   INTRODURRE I VALORI NOTI; PREMERE
   ZERO  PER GLI ELEMENTI  INCOGNITI
      ******* L A T I *******

LATO A :8
LATO B: 12
LATO C: 0

  ******* ANGOLI IN  GRADI *******

ANGOLO ALFA  (OPPOSTO AL LATO A):40
ANGOLO BETA  (OPPOSTO AL LATO B):0
ANGOLO GAMMA (OPPOSTO AL LATO C):0
```

```
            SOLUZIONE N.1:
LATO A=8
LATO B=12
LATO C=11.3144826

ANGOLO ALFA =40
ANGOLO BETA =74.6185683
ANGOLO GAMMA=65.3814318

            SOLUZIONE N.2:
LATO A=8
LATO B=12
LATO C=7.07058402

ANGOLO ALFA =40
ANGOLO BETA =105.381432
ANGOLO GAMMA=34.6185683

   NON ESISTONO ALTRE SOLUZIONI.
```

Fig. 13

6. Le istruzioni BASIC

In quest'ultimo paragrafo abbiamo elencato le istruzioni BASIC che compaiono nei vari programmi dei paragrafi precedenti. Le istruzioni sono elencate in ordine alfabetico, con una breve illustrazione del loro impiego.

ABS

Calcola il valore assoluto di un numero o di un'espressione aritmetica. Ad esempio:

Z=ABS(−3)

assegna alla variabile Z il valore 3.

ATN

Calcola l'arcotangente, in radianti, di un numero o di un'espressione. Ad esempio:

X=ATN(1)

assegna ad X il valore $\pi/4$.

Nel BASIC non sono disponibili l'arcoseno e l'arcocoseno, che vengono perciò calcolati a partire dall'arcotangente con le seguenti formule (vedi pp. 301-302):

$$\arcsen x = \text{arctg}\frac{x}{\sqrt{1-x^2}}; \qquad \arccos x = \frac{\pi}{2} - \text{arctg}\frac{x}{\sqrt{1-x^2}}.$$

COS

Calcola il coseno di un angolo espresso in radianti. Ad esempio:

H=COS(3.1415927)

restituisce H=−1.

DEF FN

Permette di definire una funzione che comparirà nel seguito del programma. Ad esempio:

DEF FNY(X)=COS(X)−1

Ponendo Z=FNY(3.1415927)
si ottiene Z=−2.

DIM

Comunica al calcolatore quale sarà la lunghezza massima di una sequenza di numeri con indice. Ad esempio:

DIM A(20)

dice al calcolatore di riservare spazio in memoria per 21 numeri che saranno chiamati A(0), A(1), ... A(20). In alcune macchine è disponibile l'istruzione OPTION BASE: scrivendo

OPTION BASE 1

viene riservato spazio solo per 20 numeri che sono A(1), ... A(20).

GET

Dice al calcolatore di attendere che l'utente prema un tasto e di memorizzare il tasto premuto. Ad esempio:

GET K$

blocca il programma in attesa che un tasto sia premuto; premendo Z, risulterà

K$="Z".

In alcune macchine il programma non si ferma sull'istruzione GET; se nell'attimo in cui l'istruzione viene eseguita non è premuto alcun tasto, K$ è posto uguale alla stringa vuota. In tal caso, si può costringere la macchina ad attendere la pressione di un tasto con le istruzioni seguenti:

```
100 GET K$
110 IF K$="" THEN 100
```

GOSUB... RETURN

Trasferisce l'esecuzione delle operazioni ad un sottoprogramma che termina con RETURN; poi l'esecuzione riprende dalla linea successiva a GOSUB. Ad esempio:

```
100 X=3
110 GOSUB 1000
120 PRINT Y
1000 Y=X-2
1010 RETURN
```

Viene assegnato ad Y il valore 3−2=1 e quindi viene stampato il numero 1.

GOTO

Trasferisce l'esecuzione delle operazioni alla linea di programma indicata. Ad esempio:

```
10 PRINT "PROVA"
20 GOTO 10
```

viene stampato in continuazione il messaggio "PROVA", finché il programma non viene interrotto dall'esterno.

HGR

Pone il calcolatore nel modo grafico ad alta risoluzione, lasciando le ultime quattro righe del video libere per il testo. Questa istruzione varia da calcolatore a calcolatore.

HOME

Pulisce lo schermo e pone il cursore in alto a sinistra. Anche questa istruzione varia: alcune macchine hanno CALL CLEAR, CLR HOME o altro.

HPLOT

Può essere usata in due modi:

— scrivendo

```
HPLOT X,Y
```

si fa comparire sul video il punto di coordinate X,Y;

— scrivendo invece

```
HPLOT A,B TO C,D
```

si fa comparire il segmento i cui estremi hanno coordinate A,B e C,D.

È un'istruzione non standard: su alcune macchine si usano PLOT, POINT, DRAW o altro.

HTAB

Sposta il cursore alla colonna indicata. Ad esempio:

```
HTAB 13
```

porta il cursore sulla tredicesima colonna, senza cambiarne la riga.

IF... THEN

Se (IF) è verificata una condizione, allora (THEN) viene eseguito ciò che segue la parola THEN. Ad esempio:

```
100 IF A<B THEN A=B
110 PRINT A
```

Viene stampato il maggiore fra i numeri A e B.
Molti calcolatori permettono il completamento dell'istruzione con ELSE (altrimenti).

INPUT

Permette d'introdurre un dato dalla tastiera. Ad esempio:

```
INPUT "ALTEZZA"; H
```

Il calcolatore mostra sul video il messaggio "ALTEZZA" e attende che venga battuto sulla tastiera il valore di H.

INVERSE
Dopo questa istruzione, tutti i messaggi appariranno sullo schermo in modo inverso, cioè scuro su fondo luminoso. Alcuni calcolatori hanno l'istruzione REVERSE ON, o altro.

NORMAL
Con NORMAL si ripristina il modo di scrittura normale; perciò NORMAL annulla l'istruzione INVERSE. Alcuni calcolatori hanno REVERSE OFF.

PRINT
È l'istruzione di scrittura, ed è molto versatile; ecco alcuni esempi:

```
PRINT 3
```

viene scritto il numero 3;

```
PRINT X+Y
```

viene scritto il valore di X+Y;

```
PRINT X,Y
```

vengono scritti i valori di X e di Y separati da alcuni spazi;

```
PRINT X;Y
```

vengono scritti i valori di X e di Y uno di seguito all'altro.

REM
Istruzione di commento (REMark); permette d'inserire dei commenti nel programma in modo d'aumentarne la leggibilità; il calcolatore si limita a riscriverli senza interpretarne il significato.

SIN
Calcola il seno di un angolo espresso in radianti; ad esempio:

```
X=SIN(3.1415/2)
```

restituisce il valore X=1.

SQR
Calcola la radice quadrata di un'espressione; ad esempio:

```
X=3
Y=4
Z=SQR(X*X+Y*Y)
```

assegna a Z il valore 5.

TAB
Si inserisce nell'istruzione PRINT per specificare in quale posizione della riga deve iniziare la stampa. Ad esempio:

```
PRINT TAB(3); 5
```

scrive il numero 5 nella terza posizione della riga, cioè dopo 2 spazi.

VTAB
Porta il cursore alla riga indicata. Ad esempio:

```
VTAB 8
```

porta il cursore sull'ottava riga dello schermo, senza cambiarne la colonna.