Sviluppo di un numero in lettere

di Massimo Cremonesi
Canonica D'Adda (BG)

In un programma di contabilità non può mancare una routine di conversione per trasformare i risultati dei calcoli, necessariamente numerici, nella corrispondente stringa letterale. Un uso pratico molto frequente è quello della compilazione di assegni, tratte ecc. tipo in cui deve essere indicato sia l'importo in cifre che quello in lettere.

Il programma presente può essere comodamente inserito come subroutine di qualsiasi programma di contabilità o di compilazione di libri contabili.

Il funzionamento del programma è abbastanza semplice: il numero da trasformare viene scomposto in centinaia, decine e unità che vengono subito convertite in lettere utilizzando i dati delle righe da 1000 a 1300.

Per quello che riguarda invece i suffissi mille e un milione o gli affissi mila e milioni è direttamente il programma principale che si preoccupa di disporli adeguatamente.

Alcuni controlli aggiuntivi, eseguiti con delle sempici IF, verificano la correttezza sintattica di giunzioni tipo UNOMILIONE e simili convertendole direttamente nella forma più usuale.

Al programma è stata aggiunta la possibilità di scrivere in maiuscolo la prima lettera del numero e in minuscolo tutte le altre.

Dal momento però che per scrivere in minuscolo delle righe dentro a un programma in Basic è necessaria sia la routine del Minus.code di Bo Arnklit che la modifica che consiste nel collegare il piedino del tasto di Shift con l'ingresso del pulsante tre sullo zoccolo dei Game Control, nella figura 2 trovate le righe da sostituire per poter far funzionare il programma senza le minuscole se non avete l'Apple-minus.

Il campo di validità del programma va da zero (0) a novecentonovantanove milioni novecentonovantanovemilanovecentonovantanove (999.999.999) ovvero al massimo nove cifre.

```
> 10
Dieci
> 13
Tredici
> 20
Venti
> 21
Ventuno
> 27
Ventisette
> 55
Cinquantacinque
> 71
Settantuno
> 79
Settantanove
> 99
Novantanove
> 101
Centouno
> 135
Centotrentacinque
> 935
Novecentotrentacinque
> 1983
Millenovecentoottantatre
> 2017
Duemiladiciassette
> 4567
Quattromilacinquecentosessantasette
> 67890
Sessantasettemilaottocentonovanta
> 17987
Diciassettemilanovecentoottantasette
```

Figura 3 - Esempio di output del programma di conversione da cifre a lettere.

```
10      GOSUB 1000
20      INPUT "> ";TT
30      IF TT = 0 THEN  PRINT "Zero": GOTO 20
40      A$ = "":B$ = "":C$ = "":T$ =  STR$ (TT):H =  LEN (T$)
50      ON (H + 2) / 3 GOTO 60,70,80
60      C$ = T$:CF$ = C$: GOSUB 90:C$ = CF$: GOTO 250
70      B$ =  LEFT$ (T$,H - 3):CF$ = B$: GOSUB 90:B$ = CF$:C$ =
          RIGHT$ (T$,3):CF$ = C$: GOSUB 90:C$ = CF$: GOTO 250
80      A$ =  LEFT$ (T$,H - 6):CF$ = A$: GOSUB 90:A$ = CF$:B$ =
          MID$ (T$,H - 5,3):CF$ = B$: GOSUB 90:B$ = CF$:C$ =  RIGHT$
         (T$,3):CF$ = C$: GOSUB 90:C$ = CF$: GOTO 250
90      K =  LEN (CF$):X$ =  RIGHT$ (CF$,1): ON K GOTO 130,100
         ,140
100     Y$ =  LEFT$ (CF$,1)
110     X =  VAL (X$):Y =  VAL (Y$): IF Y < > 1 THEN X$ = W$(
         X):Y$ = Y$(Y): GOTO 190
120     Y$ = X$(X):X$ = "": GOTO 190
130     X =  VAL (X$):W = X:X$ = W$(W): GOTO 190
140     Z$ =  LEFT$ (CF$,1):Y$ =  MID$ (CF$,2,1)
150     Z =  VAL (Z$):Z$ = W$(Z) + "cento"
160     IF Z = 0 THEN Z$ = ""
170     IF Z = 1 THEN Z$ = "cento"
180     GOTO 110
190     ON K GOTO 200,210,230
200     CF$ = X$: RETURN
210     IF X$ = "uno" OR X$ = "otto" THEN Y$ =  LEFT$ (Y$, LEN
         (Y$) - 1)
220     CF$ = Y$ + X$: RETURN
230     IF (X$ = "uno" OR X$ = "otto") AND  LEN (Y$) > 1 THEN
         Y$ =  LEFT$ (Y$, LEN (Y$) - 1)
240     CF$ = Z$ + Y$ + X$: RETURN
250     ON (H + 2) / 3 GOTO 260,270,290
260     S$ = C$: GOTO 320
270     S$ = B$ + "mila" + C$: IF  LEFT$ (S$,7) = "unomila" THEN
         S$ = "mille" + C$
280     GOTO 320
290     MM$ = "mila"
300     IF  MID$ (T$,H - 5,3) = "000" THEN MM$ = ""
310     S$ = A$ + "milioni" + B$ + MM$ + C$: IF  LEFT$ (S$,2) =
         "un" THEN S$ = "unmilione" + B$ + MM$ + C$
320     S1$ =  CHR$ ( ASC ( LEFT$ (S$,1)) - 32)
330     PRINT S1$; RIGHT$ (S$, LEN (S$) - 1)
340     GOTO 20
```

Figura 1 - Listato del programma che permette di convertire in lettere un numero intero e positivo compreso tra zero e 999.999.999.

```
1000 W$(0) = ""
1010 W$(1) = "uno"
1020 W$(2) = "due"
1030 W$(3) = "tre"
1040 W$(4) = "quattro"
1050 W$(5) = "cinque"
1060 W$(6) = "sei"
1070 W$(7) = "sette"
1080 W$(8) = "otto"
1090 W$(9) = "nove"
1100 Y$(0) = ""
1110 Y$(1) = "dieci"
1120 Y$(2) = "venti"
1130 Y$(3) = "trenta"
1140 Y$(4) = "quaranta"
1150 Y$(5) = "cinquanta"
1160 Y$(6) = "sessanta"
1170 Y$(7) = "settanta"
1180 Y$(8) = "ottanta"
1190 Y$(9) = "novanta"
1200 X$(0) = "dieci"
1210 X$(1) = "undici"
1220 X$(2) = "dodici"
1230 X$(3) = "tredici"
1240 X$(4) = "quattordici"
1250 X$(5) = "quindici"
1260 X$(6) = "sedici"
1270 X$(7) = "diciassette"
1280 X$(8) = "diciotto"
1290 X$(9) = "diciannove"
1300  RETURN
```

Figura 2 - Modifiche da apportare al programma di figura 1 nel caso non si abbia un Apple con le minuscole direttamente accessibili da tastiera.

Hi speed HGR clear

Nella puntata precedente dell'articolo "Impariamo a programmare in Assembler" avevamo presentato un esercizio didattico per pulire la pagina di testo. Nonostante il fatto che fosse il classico programmino da esercitazione abbiamo notato una notevole velocità di esecuzione sì da indurci a provare lo stesso programma per pulire la pagina grafica in alta risoluzione. Per poter essere particolarmente veloce sono stati aboliti tutti i cicli inutili e si è provveduto ad azzerare contemporaneamente tutti e trentadue i blocchi da duecentocinquantasei byte che compongono la pagina grafica in alta risoluzione. Il programma, oltre ad azzerare la pagina grafica, la può riempire di un colore qualsiasi o di un qualsiasi pattern di punti cambiando solo il valore che viene caricato nell'accumulatore all'inizio del programma. Per pulire la pagina due è purtroppo necessario cambiare tutti gli indirizzi delle STA in modo che partano da $4000 e arrivino a $5F00.

Si può ottenere ciò raddoppiando da Basic il contenuto delle celle che vanno da 775 a 868 con passo 3. Il programma di figura 2 permette appunto ciò. Per cambiare il colore dei punti provate a POKARE in 769 vari numeri tra zero e 255 prima di lanciare in esecuzione il programma col CALL 768.

Figura 1

```
0300-  A9 00      LDA   #$00
0302-  A2 00      LDX   #$00
0304-  9D 00 20   STA   $2000,X
0307-  9D 00 21   STA   $2100,X
030A-  9D 00 22   STA   $2200,X
030D-  9D 00 23   STA   $2300,X
0310-  9D 00 24   STA   $2400,X
0313-  9D 00 25   STA   $2500,X
0316-  9D 00 26   STA   $2600,X
0319-  9D 00 27   STA   $2700,X
031C-  9D 00 28   STA   $2800,X
031F-  9D 00 29   STA   $2900,X
0322-  9D 00 2A   STA   $2A00,X
0325-  9D 00 2B   STA   $2B00,X
0328-  9D 00 2C   STA   $2C00,X
032B-  9D 00 2D   STA   $2D00,X
032E-  9D 00 2E   STA   $2E00,X
0331-  9D 00 2F   STA   $2F00,X
0334-  9D 00 30   STA   $3000,X
0337-  9D 00 31   STA   $3100,X
033A-  9D 00 32   STA   $3200,X
033D-  9D 00 33   STA   $3300,X
0340-  9D 00 34   STA   $3400,X
0343-  9D 00 35   STA   $3500,X
0346-  9D 00 36   STA   $3600,X
0349-  9D 00 37   STA   $3700,X
034C-  9D 00 38   STA   $3800,X
034F-  9D 00 39   STA   $3900,X
0352-  9D 00 3A   STA   $3A00,X
0355-  9D 00 3B   STA   $3B00,X
0358-  9D 00 3C   STA   $3C00,X
035B-  9D 00 3D   STA   $3D00,X
035E-  9D 00 3E   STA   $3E00,X
0361-  9D 00 3F   STA   $3F00,X
0364-  CA         DEX
0365-  D0 9D      BNE   $0304
0367-  60         RTS
```

Figura 2

```
5  INPUT "COLORE ? (0-255)";A
10  POKE 768,169: POKE 769,A
20  POKE 770,162: POKE 771,0
30  POKE 772,202
40  INPUT "PAGINA ? ";PG
50  IF PG = 1 OR PG = 2 THEN 70
60  GOTO 40
70 PG = PG * 32
80 K = 773
90  FOR I = 0 TO 31
100  POKE K,157
110 K = K + 1
120  POKE K,0
130 K = K + 1
140  POKE K,PG + I
150 K = K + 1: NEXT
160  POKE 869,208: POKE 870,157
170  POKE 871,96
180  CALL 768
```

Questo programma permette di creare tutta la routine di HGR.CLEAR sia per la pagina 1 che per la pagina 2. Provate a POKARE vari valori in 769 prima di dare il CALL 768.

Conversioni sempre pronte

Anche se può sembrare incredibile ecco un altro programma di conversione da esadecimale a decimale e viceversa. Quello che contraddistingue questa versione della più usata e più pubblicata routine è il fatto che è interamente scritta in linguaggio macchina e che si può attivare semplicemente battendo il simbolo &.

Si può usare la conversione da decimale a esadecimale anche dentro un programma in Basic in quanto essa accetta in ingresso qualsiasi tipo di espressione, anche quelle che contengono nomi di variabili o funzioni particolari come ad esempio INT o LOG e così via.

Per usarla, dopo aver dato un BRUN HEX-DEC, basta far seguire al simbolo & un numero decimale (o una espressione) o un numero esadecimale preceduto dal segno del dollaro $; una volta premuto il return verrà stampato sulla riga sottostante il valore convertito. Per caricare la routine, dopo essere passati al Monitor con il solito CALL-151, bisogna inserire a partire dalla solita locazione $300 il codice oggetto di figura 2. Dopo l'inserimento verificare il disassemblato ($300LL) con quello di figura 1 e salvare il programma battendo BSAVE HEX-DEC,A$300,L$4E.

La routine è divisa in tre sezioni: la prima da $300 a $31C inizializza il puntatore della & e prende un carattere dalla tastiera per decidere se deve effettuare la conversione in esadecimale o quella in decimale. La seconda che va da $31E a $326 esegue laconversione decimale > esadecimale; l'ultima da $329 a $34B esegue invece la conversione di un numero esadecimale (in formato ASCII) in uno decimale. Tutte e due le routine di conversione fanno largo uso di subroutine del Basic e del Monitor.

La DD67 è quella che analizza le espressioni numeriche e insieme alla E752 depone il risultato in $50 e $51; da qui viene passato alla F941 che è destinata a stampare due byte in esadecimale.

La routine ED28, sempre del Basic, preleva un numero esadecimale di due Byte da $9E e $9F e lo stampa in decimale. Per ultima la E003 ritorna al Basic senza modificare nulla. MC

Figura 1

```
0300-  A9 4C      LDA   #$4C
0302-  A2 10      LDX   #$10
0304-  A0 03      LDY   #$03
0306-  8D F5 03   STA   $03F5
0309-  8E F6 03   STX   $03F6
030C-  8C F7 03   STY   $03F7
030F-  60         RTS
0310-  AA         TAX
0311-  F0 38      BEQ   $034B
0313-  20 B1 00   JSR   $00B1
0316-  E0 24      CPX   #$24
0318-  F0 0F      BEQ   $0329
031A-  C6 B8      DEC   $B8
031C-  F0 F2      BEQ   $0310
031E-  20 67 DD   JSR   $DD67
0321-  20 52 E7   JSR   $E752
0324-  A6 50      LDX   $50
0326-  4C 41 F9   JMP   $F941
0329-  A2 00      LDX   #$00
032B-  86 9E      STX   $9E
032D-  86 9F      STX   $9F
032F-  C9 3A      CMP   #$3A
0331-  90 02      BCC   $0335
0333-  E9 07      SBC   #$07
0335-  0A         ASL
0336-  0A         ASL
0337-  0A         ASL
0338-  0A         ASL
0339-  A2 03      LDX   #$03
033B-  0A         ASL
033C-  26 9F      ROL   $9F
033E-  26 9E      ROL   $9E
0340-  CA         DEX
0341-  10 F8      BPL   $033B
0343-  20 B1 00   JSR   $00B1
0346-  D0 E7      BNE   $032F
0348-  4C 28 ED   JMP   $ED28
034B-  4C 03 E0   JMP   $E003
```

Figura 2

```
0300- A9 4C A2 10 A0 03 8D F5
0308- 03 8E F6 03 8C F7 03 60
0310- AA F0 38 20 B1 00 E0 24
0318- F0 0F C6 B8 F0 F2 20 67
0320- DD 20 52 E7 A6 50 4C 41
0328- F9 A2 00 86 9E 86 9F C9
0330- 3A 90 02 E9 07 0A 0A 0A
0338- 0A A2 03 0A 26 9F 26 9E
0340- CA 10 F8 20 B1 00 D0 E7
0348- 4C 28 ED 4C 03 E0 00 00
```

```
10  FOR I = 100 TO 110
20  & I: PRINT
30  NEXT

]RUN
0064
0065
0066
0067
0068
0069
006A
006B
006C
006D
006E

]&$300
768
]&768
0300
]&768+12
030C
]&3*10
001E
]A=15
]&A
000F
]&SQR(400)
0014
```

Esempi di utilizzo del programma di conversione.