REGOLE DI SCRITTURA COBOL :

colonna 1 fino a colonna 6 = una volta usati per numerare le righe del programma, adesso usate normalmente per marchiare le modifiche.
colonna 7 * = commento, - = continuazione riga precedente
colonna 8 fino a colonna 11 = margine oppure area A
colonna 12 fino a colonna 72 = margine oppure area B

prima delle DIVISION del COBOL

colonna
8

IDENTIFICATION DIVISION.

serve a definire il nome del Pgm

PROGRAM-ID. nome del programma.
AUTHOR. nome del programmatore.
INSTALLATION. commento. *i commenti **non** sono obbligatori*
DATE-WRITTEN. commento.

seconda delle DIVISION del COBOL

colonna
8

ENVIRONMENT DIVISION.

esprime collegamento tra elaboratore e Pgm

CONFIGURATION SECTION.
SOURCE-COMPUTER. sigla elaboratore.
OBJECT-COMPUTER. sigla elaboratore.

SPECIAL-NAMES.
colonna
12
DECIMAL-POINT IS COMMA.
CURRENCY SIGN IS $.

INPUT-OUTPUT SECTION.

FILE-CONTROL.

colonna
12

sequenziali

SELECT nome-interno ASSIGN TO ddname-1 [PRINTER]

FILE-STATUS IS nome-del-codice-ritorno-definito-in working-storage
PASSWORD IS password

indexed

SELECT nome-interno-2 ASSIGN TO ddname-2
ORGANIZATION INDEXED
ACCESS { SEQUENTIAL | RANDOM | DYNAMIC }
RECORD KEY nome-campo-chiave-1

ALTERNATE RECORD KEY nome-campo-chiave-2
WITH DUPLICATES

FILE STATUS IS nome-codice-ritorno-2-di area-working-storage

terza delle DIVISION del COBOL

colonna
8

DATA DIVISION. *descrive il formato dei dati*

FILE-SECTION. *dati di input-output*

colonna
8 12

```
FD   nome-interno-1

     BLOCK   numero-intero   RECORDS

             ⎡   Intero                                                    ⎤
     RECORD  ⎨ VARYING FROM intero-1 TO intero-2 DEPENDING ON  nome-campo-1⎬
             ⎣ Intero-1 TO intero-2                                        ⎦

             ⎡ STANDARD    ⎤
     LABEL   ⎨  OMITTED    ⎬
             ⎣ nome specifico ⎦

     RECORDING    [ F ]   [ V ]

   [ DATA RECORD   nome-record-input-output ]
```

colonna
8 12

```
01   nome-record-input-output                    PIC X(numero-intero).
```

WORKING-STORAGE SECTION. *dati di lavoro*

colonna
8 12

```
77   nome-codice-di-ritorno-1      PIC  X(2)   VALUE  '00'. ⎫
                                                           ⎬ → Aree non contigue di memoria
77   nome-codice-di-ritorno-2      PIC  X(2)   VALUE  '00'. ⎭
```

N.B. *I numeri di livello possono andare da 01 ¸ 49, 77,88,66. Normalmente i livelli 66* ***non*** *sono utilizzati.*

colonna
8 12

01 CAMPI-RECORD. → **sotto un livello 01 le Aree di memoria sono contigue**.

colonna
12

```
                                  ⎛ A ⎞
                                  ⎪ X ⎪
02   PRIMO                   PIC  ⎨   ⎬  (Numero).
                                  ⎪ 9 ⎪
                                  ⎝ Z ⎠

                                  ⎛                DISPLAY          ⎞
                                  ⎪                POINTER          ⎪
                                  ⎪ ⎛ COMP   ⎞                      ⎪
02   SECONDO                 PIC  S9(5)V9(2) ⎨ ⎨ COMP-2 ⎬  BINARY  ⎬
                                  ⎪ ⎝ COMP-4 ⎠                      ⎪
                                  ⎝  (COMP-3)    PACKED DECIMAL    ⎠

02   TERZO                   PIC X.
     88   SI                 VALUE  'Y'.
     88   NO                 VALUE  'N'.

02   QUARTO                  PIC  9999    BLANK WHEN ZERO.
```

COPY nome-tracciato. *inserisce a questo punto la descrizione di un tracciato dati che è registrato in libreria delle copy oppure anche libreria dei sorgenti.*

Clausola PICTURE

```
01   PLUTO                    PIC   XXX.
                                   A(3).
                                   S9(5)V9(3).
                                   Z(5)9V99.
                                   ZZ.ZZZ.ZZZ,99.
                                   $ZZZZ9    VALUE 5. ⇒     $....5
                                   $$$$$9    VALUE 5. ⇒       $5
                                   +ZZZZZ9   VALUE 5. ⇒        -5
                                   ZZZCR     VALUE 5. ⇒        5
                                   ZZZDB     VALUE 5. ⇒      5DB
                                   ******9   VALUE 5. ⇒  ******5

01   FILLER                   PIC X(n).

01   TABELLA.

colonna
8      12

     05  ELEM1  OCCURS   25                PIC X.

    03    AULA.
     05  BANCHI  OCCURS   6.
        10   PERSONA-DI-DESTRA            PIC X.(8)
        10   PERSONA-DI-SINISTRA          PIC X(8)

       INDEXED BY    nome-indice-1.
```

VALUE ed inizializzazione in PROCEDURE DIVISION con OPERAZIONI-INZIALI

SKIP SKIP2 SKIP3
EJECT
TITLE 'LITERAL' *Titolo da stampare sulla prima riga di ogni pagina.*

LINKAGE SECTION. *dati di altri programmi*

```
colonna
8      12

01   Area di comunicazione          PIC   X(n).

01.  PIPPO   REDEFINES  Area di comunicazione .
```

quarta delle DIVISION del COBOL

colonna
8

PROCEDURE DIVISION

Istruzioni del programma

[USING none campo 1, nome campo 2,.....] → Per pgm chiamati

LABEL o etichette o nomi di routine interne del PGM.

STRUTTURE DI CONTROLLO : LABEL **Sequenziale**

IF
THEN _____
_____
ELSE _____
_____.

Alternativa (NIDIFICAZIONE MAX 4 livelli (meglio di meno))

PERFORM **Ripetizione**

ISTRUZIONI

COPY nome-membro. *inserisce a questo punto le istruzioni di un membro che è registrato in libreria delle copy oppure anche libreria dei sorgenti.*

ACCEPT nome campo FROM { DATE | DAY | TIME | DAY-OFWEEK } Trasferimento dati dall' esterno del Pgm

{DISPLAY | EXHIBIT} nome campo 'b=bMESSAGGIO' [UPON { CONSOLE | SPECIAL-NAMES }]

STOP RUN.

{GO BACK . | EXIT .} *per programmi chiamati*

GO TO nome label .

ISTRUZIONI ARITMETICHE

SOMMA

(1) ADD var-1 var-2 var-n TO var-x [ROUNDED] var-y [ROUNDED]
[ON SIZE ERROR istruzione o perform]

(2) ADD var-1 var-2 var-n TO var-x GIVING var-y [ROUNDED]
[ON SIZE ERROR istruzione o perform]

```
01 ADD-1.
   05 A-1    PIC 9(5).
   05 A-2    PIC 9(5).
   05 A-3    PIC 9(5).

01 ADD-2.
   05 B-1    PIC 9(5).
   05 B-2    PIC 9(5).
   05 B-3    PIC 9(5).
```

(3) ADD CORRESPONDING ADD-1 TO ADD-2 [ROUNDED]

SOTTRAZIONE

(1) SUBTRACT var-1 var-2 var-n FROM var-x [ROUNDED] var-y [ROUNDED]
[ON SIZE ERROR istruzione o perform]

(2) SUBTRACT var-1 var-2 var-n FROM var-a var-b
GIVING var-x [ROUNDED] var-y [ROUNDED]
[ON SIZE ERROR istruzione o perform]

MOLTIPLICAZIONE

(1) MULTIPLY var-1 BY var-2 [ROUNDED] var-3 [ROUNDED]
[ON SIZE ERROR istruzione o perform]

(2) MULTIPLY var-1 BY var-2 GIVING var-3 [ROUNDED] var-4 [ROUNDED]
[ON SIZE ERROR istruzione o perform]

ISTRUZIONI ARITMETICHE

DIVISIONE

(1) DIVIDE var-1 INTO var-2 [ROUNDED] var-3 [ROUNDED]
[ON SIZE ERROR istruzione o perform]

(2) DIVIDE var-1 INTO var-2 GIVING var-3 [ROUNDED] var-4 [ROUNDED]
[ON SIZE ERROR istruzione o perform]

(3) DIVIDE var-1 BY var-2 GIVING var-3 [ROUNDED] var-4 [ROUNDED]
[ON SIZE ERROR istruzione o perform]

(4) DIVIDE var-1 INTO var-2 GIVING var-3 [ROUNDED] REMAINDER var-4
[ON SIZE ERROR istruzione o perform]

(5) DIVIDE var-1 BY var-2 GIVING var-3 [ROUNDED] REMAINDER var-4
[ON SIZE ERROR istruzione o perform]

COMPUTE

COMPUTE var-1 [ROUNDED] var-2 [ROUNDED] var-3 [ROUNDED]

= espressione aritmetica

[ON SIZE ERROR istruzione o perform]

+	addizione	-	sottrazione	*	moltiplicazione
/	divisione	( )	parentesi	**	elevamento a potenza

esempi :

X = radice cubica di ((A + B) / (A - B)) COMPUTE X = ((A + B) / (A - B)) ** (1 / 3)

E = (A + ((B - C) elevato 2) / B) * D COMPUTE E = (A + (B - C) ** 2) / B) * D

ISTRUZIONI

TRASFERIMENTO

MOVE [CORRESPONDING] var-1 TO var-2 var-3 .....

regole di trasferimento tra campi NUMERICI, ALFANUMERICI, DI GRUPPO, CON VIRGOLA.

CONTEGGIO E TRASFORMAZIONE

```
                                ⎡ costante figurativa 1 ⎤          ⎡ costante figurativa 2 ⎤
TRANSFORM var-1 FROM ⎨ identificatore 1 ⎬ TO ⎨ identificatore 2 ⎬
                                ⎣ LITERAL 1 ⎦          ⎣ LITERAL 2 ⎦

                                ⎡ ALL          ⎤
                ⎡ TALLYING ⎤ ⎪LEADING    ⎪          ⎡ REPLACING BY ⎤
EXAMINE var-1 ⎨              ⎬ ⎨              ⎬ LITERAL-1⎨                ⎬ LITERAL-2
                ⎣ REPLACING ⎦ ⎪ FIRST      ⎪          ⎣      BY      ⎦
                                ⎣UNTIL FIRST ⎦
```

di ESECUZIONE e / o RIPETIZIONE

(1) PERFORM label-1 [THRU label-2] .

(2) PERFORM label-1 **n** TIMES .

(3) PERFORM label-1 [THRU label-2] UNTIL condizione-1
AND / OR condizione-2.....AND / OR condizione-n.

(4) PERFORM label-1 [THRU label-2]

```
                                ⎡ valore-1 ⎤     ⎡ valore-2 ⎤
VARYING var-1 FROM ⎨          ⎬ BY ⎨          ⎬ UNTIL condizione-1
                                ⎣identific.1⎦     ⎣identific.2⎦

⎡                          ⎡ valore-1 ⎤     ⎡ valore-2 ⎤                    ⎤
```

```
⎪ AFTER var-2 FROM { } BY { } UNTIL condizione-2 ⎪.
⎩                  ⎩identific.1⎭ ⎩identific.2⎭                ⎭
```

ISTRUZIONI

GESTIONE ARCHIVI

```
IF codice-ritorno1 NOT = '00'                          verifica codice di ritorno

          ⎛ INPUT  ⎞                                   apertura archivi
          ⎪ OUTPUT ⎪
OPEN      { }      } nome-interno-1, nome-interno-2.......
          ⎪ I-O    ⎪
          ⎝ EXTEND ⎠      (VSAM ESDS)

CLOSE Nome-interno-1 Nome-interno-2 .............Nome-interno--n.      chiusura archivi
```

posizionamento su file relative e indexed per successive letture sequenziali

```
                    ⎡        ⎡ EQUAL      TO ⎤        ⎤
START Nome-interno-1 ⎪   KEY { GREATER THAN } var-1 ⎪
                    ⎣        ⎣ NOT LESS THAN ⎦        ⎦

                    ⎡                ⎡ DISPLAY ....... ⎤ ⎤
                    ⎪ INVALID KEY { }                 } ⎪
                    ⎣                ⎣ PERFORM ......⎦ ⎦
```

lettura archivi : (mette a disposizione il prossimo record logico)
lettura sequenziale

```
(1)   READ Nome-interno-1 [ NEXT ] [ INTO area-work ] [ AT END PERFORM label-1 ].
```

```
(2)   MOVE var-1 TO var-key.                                   lettura diretta
      READ Nome-interno-1 [ INTO area-work ] KEY var-key
                                   [ INVALID KEY PERFORM label-1 ].
```

modifica record esistente

```
REWRITE nome-record-1 [ FROM area-work-1 ] [ INVALID KEY PERFORM label-1 ].
```

N.B. *il record è quello corrente, cioè l' ultimo record letto di quel file.*

cancella record esistente

[MOVE var-1 TO var-key.]
DELETE nome-file-1 [INVALID KEY PERFORM label-1].
N.B. *il record è quello corrente, cioè l'ultimo record letto di quel file.*

ISTRUZIONI

GESTIONE ARCHIVI

scrittura archivi.

scrittura file sequenziali e stampe

(1) WRITE nome-record-1 [FROM area-work]

```
⌈                                ⌈⌈ var-1 ⌉  ⌈ LINE ⌉⌉⌉
|  ⌈ BEFORE ⌉                    |⌊intero-1⌋  ⌊ LINES ⌋||
|  {          } ADVANCING        {                      }|
|  ⌊ AFTER  ⌋                    | ⌈ nome mnemonico ⌉  ||
⌊                                ⌊ ⌊      PAGE      ⌋ ⌋⌋
```

scrittura file sequenziali e stampe

(2) WRITE nome-record-1 [FROM area-work] AFTER POSITIONING ⌈ var-1 ⌉ { } ⌊ intero-1⌋

(3) MOVE var-1 TO var-key. ***scrittura file indexed e relative.***
WRITE nome-record-1 [FROM area-work].
[INVALID KEY PERFORM label-1].

UNLOCK Nome-interno--1. ***rilascia tutti i LOCK su quel file.***

COMMIT. ***forza l'aggiornamento di tutti gli archivi e subito dopo rilascia tutti i LOCK su tutti gli archivi.***

ISTRUZIONI

CHIAMATA A ROUTINE ESTERNE (ALTRI PROGRAMMI).

```
        ⎡     var-1     ⎤ ⎡                ⎡ REFERENCE (default) ⎤                        ⎤
CALL    ⎨               ⎬ ⎢ USING  BY      ⎨                     ⎬ var-2  var-3  var-4 ..... ⎥
        ⎣ LITERAL-1     ⎦ ⎣                ⎣       CONTENT       ⎦                        ⎦
```

```
                          ⎡                ⎡       CONTENT       ⎤                        ⎤
                          ⎢ USING  BY      ⎨                     ⎬ var-x  var-y  var-z ..... ⎥
                          ⎣                ⎣ REFERENCE (default) ⎦                        ⎦
```

var-1 *nome definito in working-storage come PIC X(8) VALUE ‘nome-pgm-esterno’.* ***E’ utilizzato per le CALL dinamiche***.

LITERAL-1 *‘nome-pgm-esterno’* ***E’ utilizzato per le CALL statiche***.

REFERENCE *i dati var-2 var-3 .....* ***possono essere variati dai programmi chiamati***.

CONTENT *i dati var-x var-y .....* ***possono essere variati all’ interno dei Pgm chiamati, ma nel Pgm chiamante restano invariati***.

ISTRUZIONI DI DEBUG.

READY TRACE. ***attiva la stampa delle label per cui transita il Pgm***.

RESET TRACE. ***la disattiva***

ISTRUZIONI

TABELLE.

```
 colonna
 8     12
01  TABELLA.
                                  ⎡ ⎡ASCENDING  ⎤                ⎤
    05  ELEM1  OCCURS   n    TIMES ⎪ ⎨             ⎬   KEY  IS   var-1. ⎪
                                  ⎣ ⎣DESCENDING ⎦                ⎦


    05  ELEM2  OCCURS  [  n-1   TO ]  n-2    TIMES   DEPENDING  ON  var-x

                                  ⎡ ⎡ASCENDING  ⎤                ⎤
                                  ⎪ ⎨             ⎬   KEY  IS   var-2. ⎪
                                  ⎣ ⎣DESCENDING ⎦                ⎦

                                  [  INDEXED   BY   var-x .]
```

ricerca automatica .

```
   colonna
   12
    SET   ind-1  TO   1.
    SEARCH   ELEM1     VARYING  ind-1
                      AT  END     PERFORM    label-NON-TROVATO
                       WHEN   var-1 ( ind-1 )  =   valore-cercato                    condizione
                                 PERFORM    label-TROVATO.
    [  END-SEARCH.  ]
```

ricerca dicotomica .

possibile solo con *: elem occurs N ascending key is ..... indexed by indice*

```
    SEARCH   ALL   var-2
              AT  END    PERFORM        label-NON-TROVATO
               WHEN      condizione
                         PERFORM        label-TROVATO.
    [  END-SEARCH.  ]
```