COMANDOS 'verbos' de programación genérica

Cursa este comando para que el SuperBASIC...

ARC [#<nº ID canal>,] [<coord. usuario X,Y comienzo>]
TO <coord. usuario X,Y final> ,<ángulo> [; ...]

ARC_R ... (con coordenadas de Usuario Relativas:= Δ X, Δ Y)
trace una CURVA moviendo la 'pluma' de gráficos del punto actual [o desde ése] AL punto final dado; con esa 'curvatura'

AT <fila, columna> {↓ 0 a 19},{0 → 36 | 73} TV #1
sitúe el cursor de EXposición de texto **'en'** esas **fila** y **columna.**

BLOCK [#<nº ID canal>,]<anchura>, altura>
<coord. sistema X,Y base>, nº ID color>
pinte un **bloque** rectangular de esas medidas (en puntos) y con esa 'base' (arriba-izquierda) todo lleno de **motas** de ese color.

BORDER [#<nº ID canal>,]<grosor>[,<nº ID color>]
coloque un **marco** ('reborde') interno al [a ese] panel EXpositor.

CIRCLE [#<nº ID canal>,]
<coord. usuario X0,Y0>,<radio>
[<excentricidad> **(razón de eje X a Y)**
,<orientación>] [; ...] **(ángulo semiejeX/horiz.)**

CIRCLE_R ... (con coordenadas Usuario Relativas:= Δ X, Δ Y)
trace una curva cerrada ELIPTica centrada en ese punto y de ese tamaño, forma(1:=circunf.), inclinación y acorde a escala.

CLOSE #<nº ID canal>
cierre ese 'cauce' de transferencia de datos hacia el terminal.

CLS [#<nº ID canal>,][<nº ID parte>] (CLear Screen)
aclare pantalla (i.e. ponga color de 'papel'), eliminando:
{0:=todo; 1:=encima; 2:=debajo; 3:=línea; 4:=resto línea cursor}

CSIZE [#<nº ID canal>,] **('Character SIZE)**
<nº ID ancho>, {0:=6; 1:=8; 2:=12; 3:=16 puntos}
<nº ID alto> {0:=10; 1:=20 puntos}
adopte ese tamaño para los símbolos que vaya a EXponer.

CURSOR [#<nº ID canal>,]<coord. sistema X,Y>
CURSOR [#<nº ID canal>,]<coord. usuario X,Y>, <Δx, Δy>
sitúe el cursor para textos en ese punto respecto a origen ↑ ; o respecto a escala y origen ↑_ , desplazándolo esos puntos.

DATA <lista de: <constante>
cense estos **datos** (para que -en sucesión y marcando con su 'puntero'- los **tome** como valores para las variables dadas en READ). (Para que 'repunte', consulta RESTORE).

DEF FN nombre función [(<lista <argumento FORMAL>>)]
[LOC <lista de: <variable> (LOCales)
::: instrucciones del cómputo a efectuar
RET <expresión> **(DEVUELVA**
END DEF <nombre función> **resultado)**
sepa que tú **usas** en este programa, la FuNción ahora descrita.
(Cuando quieres que la evalúe, la **nombras** como una variable:
<nombre función>[(<lista <argumento ACTUAL>>)]
y el cómputo se hará sobre los datos **percibidos** 'aquí y ahora'
y el dato resultante lo **entregará** como valor de ese nombre .

DEF PROC <nombre verbal 'proce'dura>
[(<lista <parámetro | argumento | resultado FORMAL>)]
[LOC <lista de: <variable>>]
::: instrucciones del 'proce'dimiento a seguir
[RET] (VUELVA)
END PROC <nombre verbal 'proce'dura>
sepa que tú **usas** en este programa, la 'PROCe' ahora descrita.
(Cuando deba llevarla a cabo, la cursas como un comando:
<nombre verbal 'proce'dura>
[<lista <parámetro | argumento | resultado ACTUAL>]
y las acciones del procedimiento se cumplimentarán con los datos **percibidos** 'aquí y ahora'; y los datos conseguidos los **entregará** como valores de las variables mencionadas 'aquí y ahora' como <resultado> (i.e. nombres de variables globales).

DIM <lista de: <variable>>(<lista de <máx-subíndice>>)
ocupe un espacio en memoria de las DIMensiones necesarias para reseñar los elementos que componen estas **'tablas'.** (Con tablas literales, el subíndice final será la longitud del literal).

FILL [#nº ID canal>,]<llave 1 | 0>
re_**llene** sí | no el interior de las figuras cerradas posteriores.

FLASH [#<nº ID canal>,]<llave 1 | 0>
alterne (parpadee sí | no) los colores de pluma y papel.

FOR <conta ... ... salto> :<LINEA de instrucciones a reiterar>
FOR <contador>=<comienzoS> TO <finalES> [STEP <salto>]
::: <instrucciones a repetir> ... (un ciclo del bucle)
: ... [EXIT <contador>] SALGA de este bucle
: ... [NEXT <contador>] OTRA, si procede
::: <instrucciones 'epílogo'> ... (sólo con NEXT)
END FOR <contador> (indica final de bucle)
para... indica la entrada en el bucle y los valorES 'desde'... 'hasta' [y el paso] de la variable que cuenta las 'rondas' dadas.

GOSUB <numerAL línea> ___ **(comienzo de SUBrutina)**
'vayga' (i.e.: **'vaya y venga** aquí al acabar) :::
a ejecutar esa línea y las siguientes. RETURN
(Anote dónde está ahora, para cuando le diga VUELVA)

GOTO <numerAL línea>
vaya (i.e. salte, bifurque) a esa línea, y prosiga desde ahí.

IF <premisa lógica> [THEN | :] <párrafo1> [:ELSE <párrafo2>]
IF <premisa lógica> [THEN] Y SI también... **AND**
THEN <párrafo1> O SI también... **OR**
ELSE <párrafo2> o SI una 'U' otra... **XOR**
END IF si NO es... **NOT**
si evaluada la condición (o el dato) resulta ser 'cierta' <>0
entonces efectuará las acciones del párrafo1
y se saltará completamente el párrafo2
en los **demás** casos ('falsa'=0) realizará sólo el párrafo2.

INK [#<nº ID canal>,]<nº ID color>
adopte para la 'pluma' [de ese canal] ese color de **tinta.**

INPUT [#<nº ID canal>,]
[<mensaje EXponible (variable) o "constante">]
<lista de <variable>>
INponga (i.e. ponga 'dentro') los datos tecleados [o INgresados desde ese fichero-terminal] asignándolos como valor de cada variable dada en el comando. El 'aviso' puede incluir los mismos separadores que en PRINT y ¡admite EXponer datos variables!

[LET] <variable>=<fórmula> (asignación 'coercible')
'haga' que esa variable tenga como valor el obtenido después de computar la expresión dada. (No hay discordancia de tipos).

LINE <coord. usuario X,Y hasta>
mueve el cursor sin dejar marcado en pantalla ese 'trazo'.

LINE [#<nº ID canal>,] [<coord. usuario X,Y comienzo>]
TO <coord. usuario X,Y final> [...[;...]
LINE_R ... (con coordenadas de Usuario Relativas Δ x, Δ y)
trace una RECTA (o una serie de ellas) moviendo la 'pluma' de gráficos del punto actual [o desde ése] AL punto final dado.

MODE <nº colores | nº puntos Horiz> {4 ó 512 | 8 ó 256}
adopte para la gestión de pantalla esa 'textura', o **modo.**

MOVE [#<nº ID canal>,]<distancia> (tortuga LOGO)
avance (+) o **'retroceda'** (-) la 'pluma' esa cantidad de puntos.

ON <selector> GOSUB | GOTO <lista de <numerAL línea>>
con (i.e. según) ese valor {0 a 255} computado (y truncado).
{'vayga' | vaya} a la línea de programa cuyo número ocupa el correspondiente lugar ordinal en la lista mencionada.

OPEN #<nº ID canal>,< nombre ID terminal IN/EX>

abra un 'canal' para INtro/EXtro_ducción de datos hasta/desde el programa desde/hasta ese terminal (i.e.: establezca un 'cauce' o 'vía de acceso; y reserve un espacio en memoria -un **buzón**- como depósito regulador del flujo de información).

CON_<anchura>X<altura> (coordenadas sistema)
A<xbase>X<ybase> (esquina superior-izquierda)
[_<capacidad-buzónIN>] (:='buffer' length)

SCR_ <anchura>X<altura> (coordenadas sistema)
A<xbase>X<ybase> (esquina superior-izquierda)

NET<sentido IN/EX> { Input | Output }
_<nº ID estación> { 0, 1 a 63 } (0:=EXponga a todas)
('suyo':=acceso a red)
SER<nº ID portIN/EX> { 1 'surte' | 2 'draga' }
<paridad> { Even | Odd | Mark | Space | 'no' }
<control> { Ignore | Handshake:='apretón de manos' }
<marcas> { Raw, no EOF | Z, sí EOF | C, 'cr' → 'lf' }

OPEN_IN#<nº ID canal>,
MDV<nº ID ductora>_<título ID fichero>

abra un 'cauce' para INtroducir hasta el programa, desde esa MicroDriVe, los datos 'ya' grabados en ese fichero existente.

OPEN_NEW#<nº ID canal>,
MDV<nº ID ductora>_<título ID fichero>

abra un 'cauce' para EXtroducir los datos desde el programa hasta esa MicroDriVe y archivarlos en ese **nuevo** fichero.

GENERACION DE SONIDO

BEEP [<duración>,<tono-comienzo>
[,<tono-fin>,<escalón-tiempo>,<escalón-tono>
[,<perfil-envolvente> (elegible de 0 a 15)
[,<reverbero-azar> (elegible de 0 a 15)
[,<ruido-blanco>]]]]]

pite durante ese tiempo { 0 a 32767 } en unidades de 72 ms. (0:=por siempre o hasta cursar BEEP), desde la nota dada por ese 'período' de tono { 0 a 255 } agudo a grave. Aprox.:

C	C#	D	D#	E	F	F#	G	G#	A	A#	B
77	72	68	64	59	56	52	48	44	41	38	36
33	30	28	26	24	22	20	18	17	15	14	13
11	10	09	08	07	06	05	04	03			

[y variando gradualmente { 0 a 15; -8 a 7 } hasta ese tono final]
[agregando componentes aleatorias para efectos especiales].

BEEPING

el valor logical que atestigua sí | no está pitando.

OVER [# <nº ID canal>,]<llave 1 | 0 |-1> ('sobre'scribir)

EXponga encima | borrando todo | sólo donde no coinciden motas.

PAN [# <nºIDcanal>,]<Δ horiz-puntos>[,<nºIDparte>]

desplace la imagen [en ese panel] a derecha (+) o izquierda (-).

PAPER [# <nº ID canal>,] <nº ID color>

adopte ese color de **papel** en el próximo comando de gráficos.

PAUSE [<demora>] (períodos de 20 msegs.)

haga una **pausa** en la ejecución, de esa duración (o hasta pulsar).

PENUP | PENDOWN [# <nº ID canal>] (tortuga LOGO)

suba | **baje** la 'pluma' de gráficos (no | sí dejando trazo visible)

POINT[#<nº ID canal>,] <coord. usuario X,Y> [TO | ;...]

POINT_R ... (con coordenadas de usuario relativas:= Δ X, Δ Y)

pinte una(s) **mota**(s) de tinta en ese punto (1 ó 2 **puntos** Horiz.).

PRINT <nº ID canal>, < lista de <dato a EXponer>> (no ?)

EXponga (i.e. ponga 'fuera') por pantalla [o por ese canal]
esos datos separados por: (blanco) | (! blanco inteligente) |
(\ nueva línea) | (, cada 8 columnas) | (; sin separación)
(TO <columna> **tab**ulación, salta hasta esa columna).

RANDOMISE [<numeral>]

germine la serie pseudo**ale**atoria prescrita [o desde ese dado] .

READ <lista de <variable>>

tome el dato **'homólogo'** en el censo de constantes DATA y apúntelo como valor de la variable correspondiente dada aquí.

RECOL [# <nº ID canal>,] <nº ID color0>,... ,<nº ID color7>

cambie ahora todos los IDentificativos de los colores, **recoloree**.

REM <resto línea> (no ')

memorice este comentario y **recuérdemelo** en los listados.

REP <nombre ID bucle>:<línea de instrucciones a reiterar>
REP <nombre ID bucle> (indica el comienzo)
... <instrucciones a repetir> ... → (un ciclo del bucle)
: ... EXIT <nombre ID bucle> SALGA de este bucle
: ... [NEXT <nombre ID bucle> [OTRA 'ronda']
END REP <nombre ID bucle> (indica el final)

repita esa serie de instrucciones **'hasta que'** se cumpla la condición descrita para que **salga,** pasando entonces a la instrucción inmediatamente detrás de la culminación del bucle.

RESTORE <numerAL línea>

'repunte' su puntero DATA hacia la primera [esa] línea.

SCALE [# <nº ID canal>,]<alturapanel en coord. usuario>,
<coord. usuario X,Y para punto Inf-Izq-pantalla>

adopte esa **escala** vertical para la [esa] ventana de gráficos.

SCROLL [# <nºIDcanal>,]< Δ-vert-puntos>[,<nºIDparte>]

'desrrolle' la imagen hacia arriba (+) o hacia abajo (-).
[10:=toda; 1:=encima la línea; 2:=debajo la línea]

SEL ON <selector>=<numeral>:<LINEA de instrucciones>
SEL ON <selector> (indica comienzo)
[ON <selector>] =<numeral>
::: <instrucciones a ejecutar> (CON cada caso)
[[ON <selector>] = REMAINDER
::: <instrucciones a ejecutar> (CON los RESTANTES casos)
END SEL <selector> (indica final)

elija las acciones a ejecutar CON cada valor o **gama** (a **TO** b).

STRIP [# <nº ID canal>,]<nº ID color>

proyecte una **'banda'** como fondo del texto (color de PAPER).

TURN [# <nº ID canal>,] <cambio-rumbo> (tortuga LOGO)

gire a Derecha(-) o a iZquierda(+) esa cantidad de grados.

TURNTO [# <nº ID canal>,] <rumbo> (tortuga LOGO)

gire hasta orientarse así (Ref. semieje horizontal derecho).

UNDER [# <nº ID canal>,] <llave 1 | 0>

subraye sí | no, en los textos EXpuestos ulteriormente.

WIDTH [# <nº ID canal>,] <núm.columnas>

adopte esa **anchura** [para ese terminal] (¡no para CONsola!).

WINDOW [# <nº ID canal>,] <anchura>,<altura> (en puntos)
<coord. sistema X,Y para punto Sup-Izq-ventana>

use ese **panel** como terminal-EXpositor ligado [a ese canal] .

OPERADORES LOGICO-MATEMATICOS

=	igualdad	<	menor que	&	concatenación
< >	desigualdad	>	mayor que		literales
==	cuasi-igualdad	<=	menor o igual	INSTR	'pertenencia'
		>=	mayor o igual		lit1º a lit2º

		'logical'	OPERACION	'binaria'
+ -	adición sustrac.			
* /	multip. divis.	NOT	Negación	~~
^	exponenciación	AND	Y_L_iación	&&
DIV	cociente entero	OR	O_L_Iación	| |
MOD	residuo o resto	XOR	O_L_Eación	^ ^

Funciones 'nombres de PROGRAMACION GENERICA

Nombra esta FUNCION para que entregue el valor...

CHR$ (<numeral-entero>) **(CHaRacter)**

literal **caracter**ístico con ese código interno (octeto 0 a 255).

CODE (<literal>)

del **cód**igo interno equivalente al carácter **inicial** del literal.

DATE

resultante de convertir la "fecha" interna, a un número 'real'.

DATE$
DATE$ (<numeral-coma-flotante>)

literal "aaaa mmm dd hh:mm:ss" de la **fecha** interna (dada) (JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC).

DAY$
DAY$ (<numeral-coma-flotante>)

literal del **día** { MON TUE WED THU FRI SAT SUN } según la fecha interna (o el resultante de extraerlo de la fecha dada).

DIMN ([<nombretabla>,<ordinal-subíndice>])

numeral máximo del primer [ese] subíndice.

EOF [(#<nº ID canal>)] **(End Of File)**

logical 'testigo' del final del fichero vinculado a ese canal. EOF -sin argumento- da el final del censo de DATA constantes.

FILL$ (<literal 1|2 caracteres>,<longitud>)

literal **'llenado'** repitiendo hasta esa cantidad los 1 ó 2 dados.

INKEY$ [([#<nº ID canal>,]<demora>)] **(INput KEY)**

literal del Primer carácter INscrito vía el teclado [o vía ese canal]. Si la 'demora' es positiva espera esos períodos de 50 segs.; si -1:=espera por siempre; si 0:=no espera nada.

INT (<numeral>) **(INTeger)**

de la parte **en**tera (truncando hacia - ∞) del numeral 'real' dado.

LEN (<literal>) **(LENgth)**

de la cantidad de caracteres (la **'lon**gitud') del argumento dado.

RND [([<entero>] [TO <numeral>])] **RaNDom:='aZAR'**

numeral siguiente en la serie pseudo**ale**atoria interna.
[RND:=0 ó 1; RND(<n>):={0 al n}; RND (<m TO n>):={m AL n}

COMANDOS 'verbos' para CONFECCION DE PROGRAMAS

Cursa este comando al SuperBASIC para que...

ADATE <segundos> **(Adjust DATE)**

ajuste la 'fecha' adelantando (+) o atrasando (-) su cronómetro.

AUTO [<nº línea-comienzo>][,<incremento>]

reclame la entrada del **edit**ador y que **'auto**'máticamente numere cada nueva línea del programa. (Véase también EDIT) Se **corta** ('rumpe') pulsando CTRL y la tecla ESPACIADORA.

BAUD <cadencia en baudios>

adopte esa **rapidez** para las transferencias por los portIN/EX de acceso serie. {75 300 600 1200 2400 4800 9600 19200}

CLEAR

anule los valores dados a las variables y libre el espacio.

CONTINUE

siga la ejecución del programa parada con STOP o cortada (con ConTRoL y tecla ESPACIADORA). ('break':=inter**rupción**).

COPY <nombre ID term IN_>[<título ID fichero>]
TO <nombre ID term EX_>[<título ID fichero>]

lleve una 'copia' del texto contenido en el fichero-terminal IN ('origen' transferencia) AL fichero-terminal EX ('destino').

COPY_N ... ... **(COPY No heading)**

lleve..., **no** copiando la cabecera **-testero-** del fichero.

DELETE MDV<nº ID ductora>_<título ID fichero>

borre el fichero de ese nombre alojado en esa 'MicroDriVe'.

DIR MDV<nº ID ductora>_

muestre los **títulos** -o nombres de los ficheros- reseñados en el **dir**ectorio del cartucho de cinta alojado en esa 'MicroDriVe'. (Indica los <sectores libres>/<total sectores> de la cinta).

DLINE <lista de <nº de línea TO nº línea | nº línea>> **(Delete)**

tache del programa esos párrafos y/o esas líneas concretas.

EDIT <nº línea-comienzo>[,<incremento>]

reclame la entrada del **edit**ador para que **revise** a partir de esa línea, (ya sean existentes o nuevas). Usa CTRL → o CTRL ← para suprimir; ENTER cuando ya **vale;** las teclas ↑ y ↓ actúan como ENTER y además pasan a la anterior o posterior existente. Si la línea a revisar no existe, se trata como AUTO.

FORMAT MDV <nº ID 'drive'>_[<título ID cartucho>]

"con**forme** la superficie de la cinta en 'octetos' y 'segmentos'

LIST [#<nº ID canal>,]
<lista de <nº línea TO nº línea | nº línea>>

liste por pantalla [o por ese terminal] ese (esos) párrafo(s) y/o esa(s) línea(s) del programa en memoria (SER1 # impresora).

LOAD <nombre ID term IN>_[<título ID fichero>]

traiga de ese terminal (normalmente una de las 'microdrives' MDVn) el programa contenido en ese fichero, ('load':=cargar en)

LRUN <nombre ID term IN>_[<título ID fichero>] **(Load&RUN)**

traiga..., y **'rule'**lo inmediatamente después ('run':=correr, pasar)

MERGE <nombre ID term IN>_[<título ID fichero>]

mezcle al programa en memoria el programa traído de ese fichero-terminal. (Prevalecen las líneas del programa INtrante).

MRUN <nombre ID term IN>_[<título ID fichero>] **(Merge&RUN)**

mezcle... ...-terminal, y **'rule'** después el programa obtenido.

NEW

quite el programa presente en memoria. (Anule y **renueve** todo).

RENUM [<nº línea-comienzo>[TO<nº línea-final>;]
[<nuevo nº línea-comienzo>][,<incremento>]]

renumere las líneas de programa posteriores a esa de comienzo y hasta la última del programa [o sólo hasta ésa de final de párrafo], comenzando a numerar a partir de ese nuevo número.

RETRY

pruebe (i.e. vuelva a intentar la ejecución de) la línea errónea.

RUN [<numerAL de línea>]

'rule' el programa en memoria desde la primera [o desde ésa].

SAVE <nombre ID term EX>_[<título ID fichero>]
[,<lista de <nº línea TO nº línea | nº línea>>]

guarde en ese terminal (normalmente una de las 'microdrives' MDVn) alojándolo en ese fichero, el [esa parte del] programa.

SDATE <año>,<mes>,<día>,<hora>,<min>,<seg> **(Set DATE)**

fije la "fecha" actual según los datos mencionados. (año:=aaaa).

STOP (¡No admite END!)

pare la ejecución del programa en curso. (Se toma la última)

COMANDOS 'verbos' y FUNCIONES 'nombres' ESPECIFICOS

CALL <dirección>[,<lista de <parámetro | argumento>

ceda el control a ('**cite**' la) subrutina en Cod. Máq. que comienza en esa **dire**cción. Traspasables hasta 7 parámetros-dirección y hasta 6 parámetros-datos, en D1 a D7, A0 a A5 del µP68008. Se devuelve el control mediante 'MOVE Q # 0,D0' 'RTS'.

EXEC <lista de:<nombre ID termIN>_<título ID fichero-prog>
[,<lista de:<nombre ID termIN/EX>_<título ID fich-dat>]

ejecute el programa en Cod. Máq. (o en 'paralelo' los programas) trayéndolo de ese fichero [,y operando sobre ese (esos) fichero(s) de datos, y 'volviendo' en cuanto inicie el proceso(s).

EXEC_W ... ... **EXECute & Wait**

ejecute ... ... , y esperando hasta que concluyan ese proceso(s).

LBYTES <nombre ID termIN>_<título ID fich-cod> **(Load BYTES)**
,<dirección-comienzo>

traiga a la memoria desde ese terminal-fichero ese bloque consecutivo de **octe**tos, asentando el primero de ellos en esa dirección. (Guardado con SBYTES). **(dire 131072='clisé' imagen)**

NET <nº ID estación> (NETwork)

asigna el número { 0 a 127} propio de cada estación de la **red.**

PEEK (<dirección>) { **0 a 2^32=4 Megabytes** }

el octeto { 0 a 255} que **hay** en esa celdilla de memoria.

PEEK_L (<dirección>) **(PEEK_Long [word])**

el **cuádruple-octeto** ('palabra larga') { 0 a 4294967295} formado por el que **hay** en esa celdilla (0 a 2^32, 'par') de memoria señalada y los que hay en cada una de las tres sucesivas.

PEEK_W (<dirección>) **(PEEK_Word)**

el **doble-octeto** ('palabra') { 0 a 65535} formado por el que **hay** en esa celdilla (0 a 2^32, 'par') de memoria y el de la sucesiva.

POKE <dirección>,<octeto> { **0 a 2^32**},{ **0 a 255** }

meta **ahí** (en la celdilla de memoria señalada), ese valor.

POKE_L <dirección>,<cuádruple-octeto> **(POKE_Long [word])**

meta **ahí** (en la celdilla de memoria señalada) { 0 a 2^32} y en las tres sucesivas, ese valor { 0 a 424967295} ('palabra larga').

POKE_W <dirección>,<doble-octeto>

meta **ahí** (en la celdilla de memoria señalada) { 0 a 2^32} y en la sucesiva, ese valor { 0 a 65535} ('palabra').

RESPR (<espacioctetos>) **(RESident PRocedure)**

la dirección del bloque que -por nombrar esta **función**- queda reservado dentro del programa del usuario, para sus PROCEs.

SBYTES <nombre ID termEX>_<título ID fich-cod>
,<dirección-comienzo> **(Save**
,<cantidad-octetos> **BYTES)**

guarde en ese fichero-terminal el bloque consecutivo de **octe**tos asentados en memoria a partir de esa dirección y sucesivas hasta esa cantidad. (v.g. dire 131072='clisé' de imagen)

SEXEC <nombre ID termEX>_<título ID fichero-prog>
,<dirección-comienzo-programa> **(Save for**
,<cantidad-octetos-programa> **EXECution)**
,<cantidad-octetos-datos-programa>

guarde en ese fichero-terminal (en formato **eje**cutable ulteriormente con EXEC) ese programa en Cód. Máq. µP 68008.

KEYROW (<nº ID hilera-teclado>)

el valor { 2^n; n=0 a 7 } asociado a la pulsación de la tecla situada en esa ristra { 0 a 7 } horizontal según tabla:

	1	2	4	8	16	32	64	128
7	⇧	CTRL	ALT	X	V	<	N	,
6	8	2	6	Q	E	Ø	T	U
5	9	W	I	TAB	R	-	Y	O
4	L	3	H	I	A	P	D	J
3	'	⇕	K	S	F	=	G	Ñ
2	´	Z	.	C	B	[	M	;
1	↵	⇦	⇧	ESC	⇨	]	ESPACIO	⇩
0	F4	F1	5	F2	F3	F5	4	7

FUNCIONES MATEMATICAS

ABS (<x>)	la **magni**tud o valor absoluto del argumento
ACOS (<x>)	el Arco -en radianes- cuyo **coseno** es x
ACOT (<x>)	el Arco -en radianes- cuya **cotangente** es x
ASIN (<x>)	el Arco -en radianes- cuyo **seno** es x
ATAN (<x>)	el Arco -en radianes- cuya **tangente** es x
COS (<α>)	el **coseno** de ese ángulo (α radianes)
COT (<α>)	la **cotangente** de ese ángulo (α radianes)
DEG (<α>)	el equivalente en **grados** de α radianes
EXP (<x>)	el **antilogaritmo** en base **e** del argumento (e^x)
LN (<x>)	el **logaritmo neperiano** (base **e**) del argumento x
LOG10 (<x>)	el **logaritmo decimal** (base 10) del argumento x
PI	la constante universal π (:=3,141593)
RAD (<α>)	el equivalente en **radianes** de α grados
SIN (<α>)	el **seno** de ese ángulo (α radianes)
SQRT (<x>)	la **raíz** cuadrada de x (x positivo)
TAN (<α>)	la **tangente** de ese ángulo (α radianes)