DIALOG BOXES / CAJAS DE DIALOGO

(Guide under construction. Soon also in English)

By: Jaime Meza

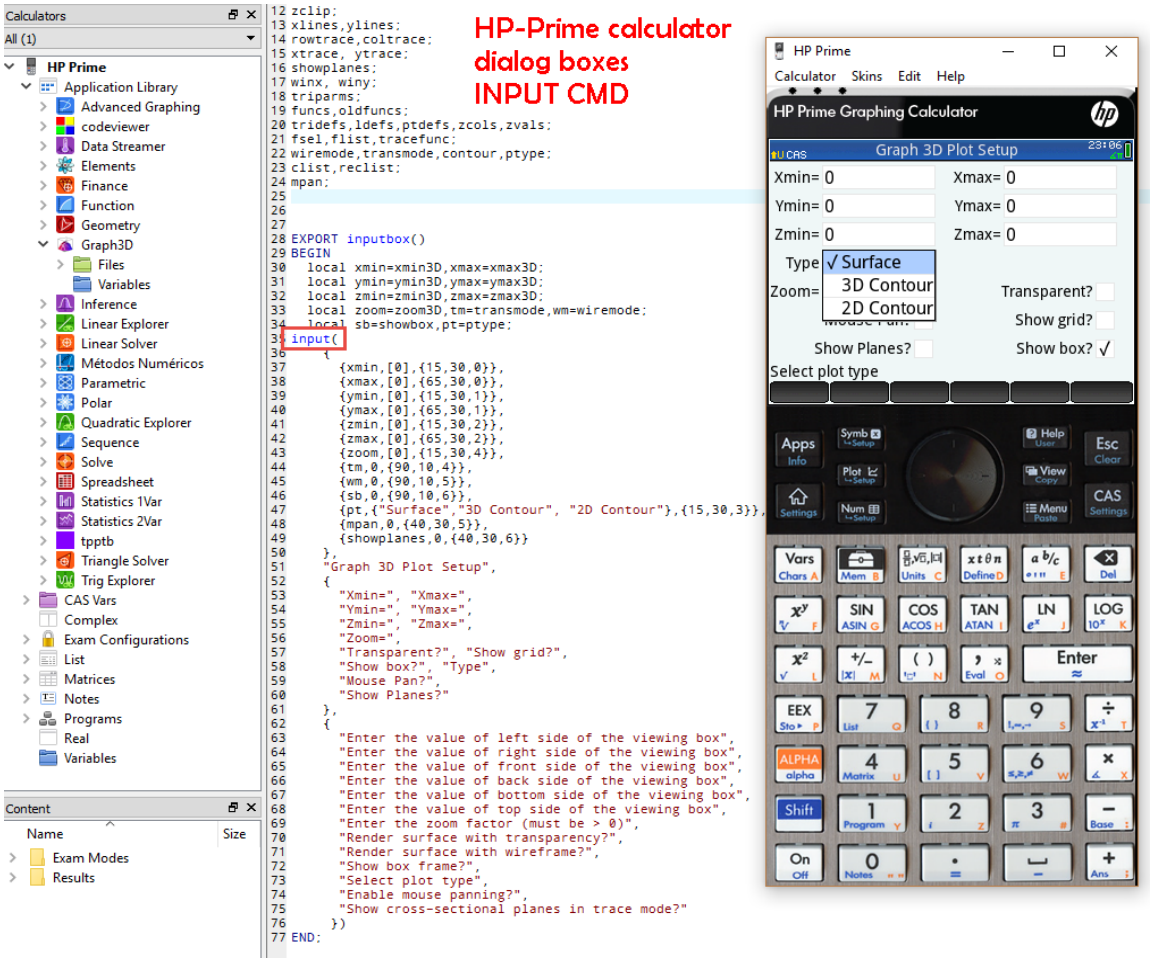
Version 1.0.0 Revision: 03, Mar 05 - 2017 ©

Comments, suggestions, etc. are welcome at eonicasys@gmail.com

Updates of this document go to: www.eonicasys.com.co ElectrONICs And SYStems (SISTEMAS COMPUTACIONALES, SISTEMAS EMBEBIDOS (microcontroladores), SISTEMAS ALGEBRAICOS COMPUTACIONALES (CAS/SAC)

Las ordenes de entrada/salida (E/S) nos permiten la interface entre usuario/calculadora, para introducir y desplegar información. EL CÓDIGO FUENTE PARA CREAR GUI EN LA HP-PRIME es un poco más complejo y laborioso, pero el siguiente tutorial muestra diversos ejemplos para adquirir en la HP-Prime dominio experto en la creación de CAJAS DE DIALOGO =)

Primer orden en la HP-PRIME: INPUT COMMAND [ENG]/ ORDEN DE ENTRADA [SPN], con esta orden podemos obtener un cuadro de dialogo de entrada de datos como el que se aprecia en la siguiente imagen, que es la interface inicial de la famosa aplicación GRAPH3D, su creador también, el famoso programador de calculadoras Hewlett Packard (HAN). Más información sobre Graph3D en http://www.hpmuseum.org/forum/thread-95.html



En los comentarios de cada ejemplo, se muestra el objetivo de cada programa

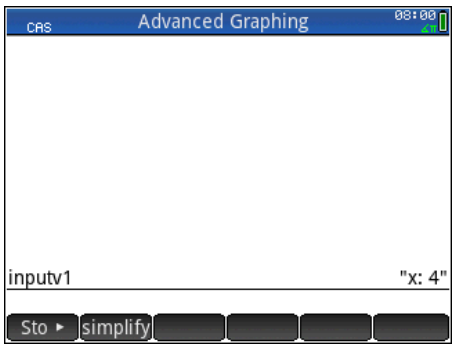
Ejemplo INPUT versión #01

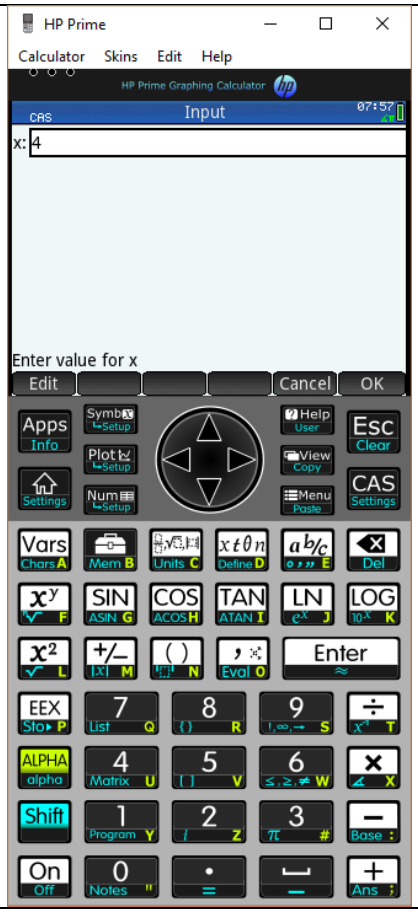
```
export inputv01()
BEGIN
    // I: forma más simple, requiere un solo argumento, el nombre de la variable a modificar

    // ARGUMENTO#1 variable(s) obligatoria(s): por ejemplo x
    // ARGUMENTO#2 etiqueta opcional: si no se incluye este argumento, se agrega automáticamente
 a la caja de dialogo
 el mismo nombre de la variable más ":" , en este caso "x:"
    // ARGUMENTO#3 titulo opcional de la caja de dialogo: por defecto coloca el nombre "Input"
    // ARGUMENTO#4 ayuda opcional: agrega la cadena "Enter value for" + nombre de la variable, e
n este caso
"Enter value for x" si la calculadora está en language/English
    // ARGUMENTO#5 valores opcionales de reinicio de cada variable
    // ARGUMENTO#6 valores iniciales que se muestran en cada variable (opcionales)
    local x;
    x := 4; // valor inicial siempre que se llame a la orden de usuario inputv1()
    input( x ); // un solo argumento
    // [enter] ó [ok] ejecuta los campos de la caja de dialogo
    // [esc] ó [cancel] sale de la caja de dialogo sin modificar los campos de entrada.
    return "x: "+x;
END;
```

Código anterior sin comentarios

```
export inputv1()
BEGIN
    local x;
    x := 4;
    input( x );
    return "x: "+x;
END
```

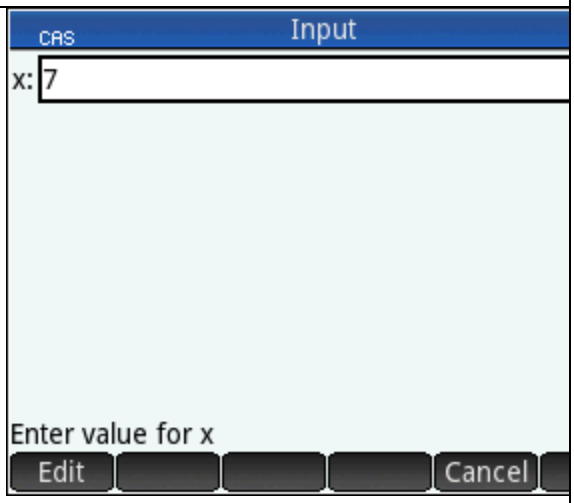




Ejemplo INPUT versión #1a/1b

```
export x := 7;
export inputv1a()
BEGIN
  // para RECUPERAR EL ULTIMO VALOR ASIGNADO, se debe cr
ear x como
 variable global "export x := 7;"
  input( x );
  return "x: "+x;
END;
```

```
export inputv1b()
BEGIN
  // para RECUPERAR EL ULTIMO VALOR ASIGNADO sin usar EX
PORT CMD,
también puede usar USER VARIABLES EN MAYÚSCULAS [A,B,...
,Z]
  input( X );
  return "X: "+X;
END;
```

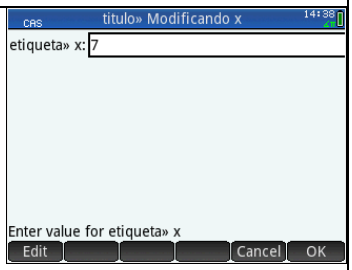


Ejemplo INPUT versión #1c

```
export x := 7;
export inputv1c()
BEGIN
  // agregando TÍTULO PERSONALIZADO, argumento #2
  input( x, "titulo» Modificando x" );
  return "x: "+x;
END;
```

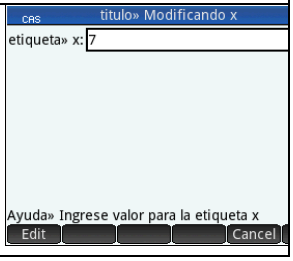
Ejemplo INPUT versión #1d

```
export x := 7;
export inputv1d()
BEGIN
  // agregando TÍTULO Y NOMBRE DE LA ETIQUETA PERSONALIZADA, argumentos
 #2-3
  input( x, "titulo» Modificando x", "etiqueta» x:" );
  // note que la ayuda automáticamente elimina los dos puntos
 si los hay al final de la etiqueta mostrando solo
"Enter value for etiqueta X" y no "Enter value for etiqueta X:"
  return "x: "+x;
END;
```



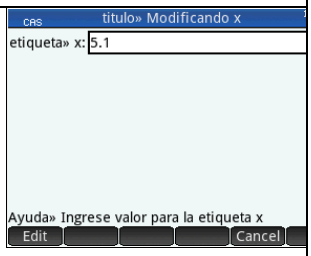
Ejemplo INPUT versión #1e

```
export x := 7;
export inputv1e()
BEGIN
  // agregando TÍTULO, ETIQUETA Y AYUDA PERSONALIZADA, argumentos #2-4
  input( x, "titulo» Modificando x","etiqueta» x:","Ayuda» Ingrese valor pa
ra la etiqueta x" );
  return "x: "+x;
END;
```



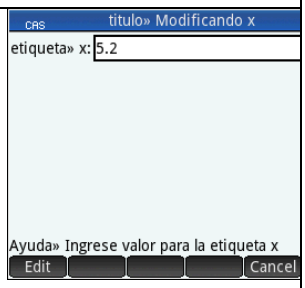
Ejemplo INPUT versión #1f

```
export x := 7;
export inputv1f()
BEGIN
  // agregando TITULO, ETIQUETA, AYUDA, Y VALOR DE REINICIO PERSONALIZADO,
 argumentos #2-5
  local valorDeReinicio := 5.1;
  input(
  x,
  "titulo» Modificando x",
  "etiqueta» x:",
  "Ayuda» Ingrese valor para la etiqueta x",
  valorDeReinicio
  );
  // el valor de reinicio se coloca con la secuencia [shift]+[esc] ó solo
[backspace]
  return "x: "+x;
END;
```



Ejemplo INPUT versión #1g

```
export x := 7;
export inputv1g()
BEGIN
  // agregando TÍTULO, ETIQUETA, AYUDA, VALOR DE REINICIO E INICIAL PERSONA
LIZADO, argumentos #2-5
  //x := 5.2; // ya no es necesario definir un valor inicial en esta forma
  local valorDeReinicio := 5.1234;
  local valorInicial := 5.2;
  input(
  x,
  "titulo» Modificando x",
  "etiqueta» x:",
  "Ayuda» Ingrese valor para la etiqueta x",
  valorDeReinicio,
  valorInicial
  );
  return "x: "+x;
END;
```

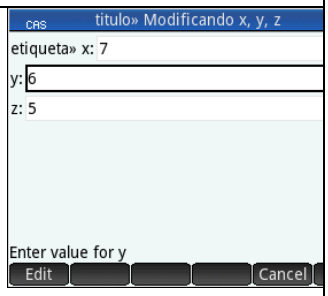


Ejemplo INPUT versión #1h

```
export x := 7;
export inputv1h()
BEGIN
  // ARGUMENTOS POR DEFECTO se agrega {} y para campo vacío ""
// {} para nombre de etiqueta por defecto y ayuda vacía o sin texto
  input( x, "titulo» Modificando x", {}, "" );
  return "x: "+x;
END;
```

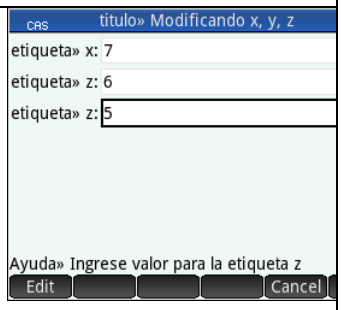
Ejemplo INPUT versión #02

```
export x := 7; export y := 0; export z := 0;
export inputv02()
BEGIN
  // para un CONJUNTO DE VARIABLES se abarcan entre {}
  input( { x, y, z }, "titulo» Modificando x, y, z", "etiqueta» x:",
 "Ayuda» Ingrese valor para la etiqueta x" );
  // note que las cadenas de texto para etiqueta y ayuda solo afectan a l
a primera variable
  return { "x: "+x, "y: "+y, "z: "+z };
END;
```



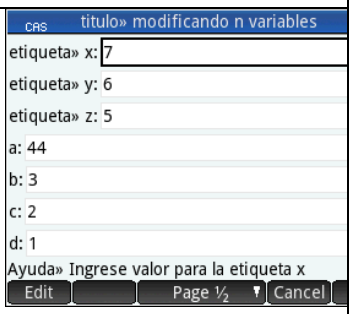
Ejemplo INPUT versión #2a

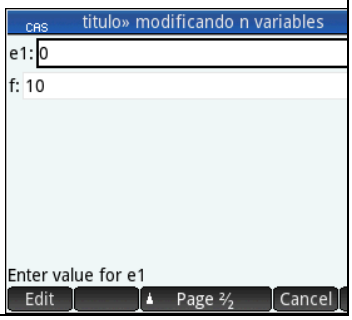
```
export x := 7; export y := 0; export z := 0;
export inputv2a()
BEGIN
  // para un CONJUNTO DE ETIQUETAS Y AYUDAS SE ABARCAN ENTRE {}
  input( { x, y, z }, "titulo» Modificando x, y, z",
  { "etiqueta» x:", "etiqueta» z:", "etiqueta» z:" },
  { "Ayuda» Ingrese valor para la etiqueta x", "Ayuda» Ingrese valor par
a la etiqueta y",
"Ayuda» Ingrese valor para la etiqueta z" }
  );
  return { "x: "+x, "y: "+y, "z: "+z };
END;
```



Ejemplo INPUT versión #2b

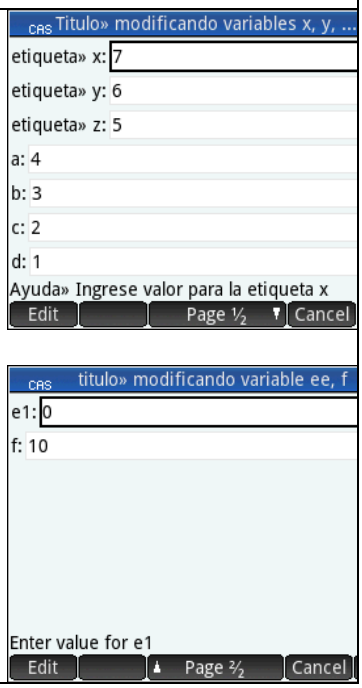
```
export x := 7; export y := 0; export z := 0;
export a := 4, b := 3, c := 2, d := 1, e1 := 0;
export f := 10;
export inputv2b()
BEGIN
  // Si hay más de 7 variables LOS CAMPOS SE MUESTRAN EN N PANTALLAS
  input(
  { x, y, z, a, b, c, d, e1, f },
  "titulo» modificando n variables",
  { "etiqueta» x:", "etiqueta» y:", "etiqueta» z:" },
  { "Ayuda» Ingrese valor para la etiqueta x", "Ayuda» Ingrese valor pa
ra la etiqueta y",
 "Ayuda» Ingrese valor para la etiqueta z" }
  );
  // puede completar las cadenas de texto para las etiquetas y ayuda pa
ra A,B,C,D,EE
  // con la tecla virtual [page #/#] se avanza ó REGRESA entre pantalla
s
  return { x, y, z, a, b, c, d, e1, f };
END;
```





Ejemplo INPUT versión #2c

```
//export x := 7; export y := 0; export z := 0;
//export a := 4, b := 3, c := 2, d := 1, e1 := 0;
//export f := 10;
export inputv2c()
BEGIN
  // para un CONJUNTO DE TÍTULOS se abarcan entre {}
  input(
  { x, y, z, a, b, c, d, e1, f },
  { "Titulo» modificando variables x, y, ... d", "titulo» modificando v
ariable ee, f" },
  { "etiqueta» x:", "etiqueta» y:", "etiqueta» z:" },
  { "Ayuda» Ingrese valor para la etiqueta x", "Ayuda» Ingrese valor pa
ra la etiqueta y",
"Ayuda» Ingrese valor para la etiqueta z" }
  );
  return { x, y, z, a, b, c, d, e1, f };
END;
```

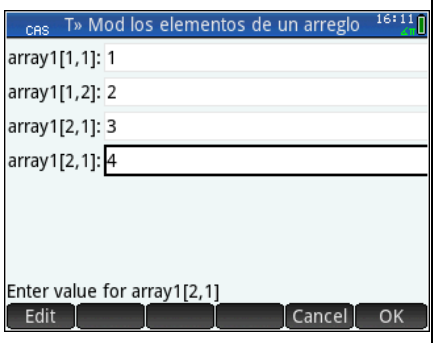


Ejemplo INPUT versión #03

```
export inputv03()
BEGIN
  // Variable con n elementos tipo LISTA
  local list1 := { 5, 4, 3, 2};
  input( {list1[1], list1[2], list1[3], list1[4]},
  "T» Mod los elementos de una lista",
  { "list[1]:", "list[2]:", "list[3]:", "list[4]:" }
  );
  return list1;
END;
```

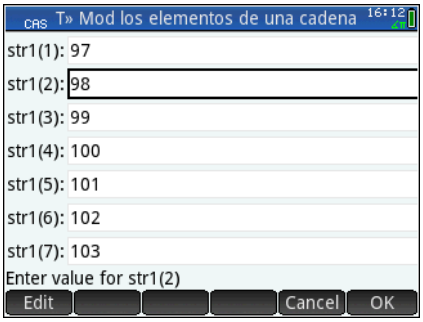
Ejemplo INPUT versión #3a

```
export inputv3a()
BEGIN
  // Variable con n elementos tipo ARRAY
  local array1 := [[1,2],[3,4]];
  input(
  {
  array1[1,1],
  array1[1,2],
  array1[2,1],
  array1[2,2]
  },
  "T» Mod los elementos de un arreglo",
  { "array1[1,1]:", "array1[1,2]:", "array1[2,1]:", "array1[2,1]:"
 }
  );
  return array1;
END;
```



Ejemplo INPUT versión #3b

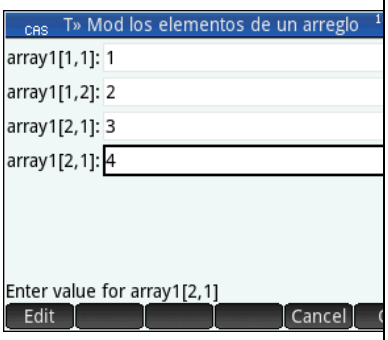
```
export inputv3b()
BEGIN
  // Variable con n elementos tipo CADENA DE CARACTERES
  local str1 := "abcdefgh";
  input(
  {
  str1[1],
  str1[2],
  str1[3],
  str1[4],
  str1[5],
  str1[6],
  str1[7]
  },
  "T» Mod los elementos de una cadena"
  );
  return str1;
END;
```

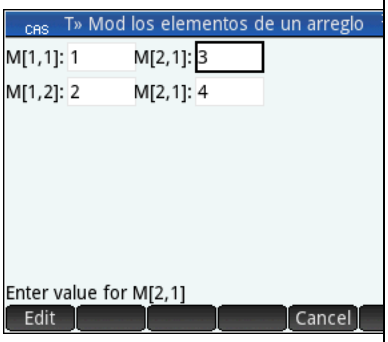


Ejemplo INPUT versión #3c

```
export inputv3c()
BEGIN
  // POSICIONANDO LOS CAMPOS DE ENTRADA ó desfasándolos
x% hacia la derecha, con x1% de ancho
  local offsetField0 := 0; //0%
  local offsetField5 := 5; //10%
  local offsetField10 := 10; //%
  local offsetField15 := 15; //%
  local offsetField20 := 20; //%
  local offsetField25 := 25; //%
  local offsetField30 := 30; //%
  local offsetField40 := 40; //%
  local offsetField45 := 45; //%
  local offsetField50 := 50; //%
  //...
  local offsetField100 := 100; //100%
  local widthField0 := 0; //0%
  local widthField10 := 10; //%
  local widthField15 := 15; //%
  local widthField20 := 20; //%
  local widthField25 := 25; //%
  local widthField30 := 30; //%
  local widthField40 := 40; //%
  local widthField50 := 50; //%
  // ...

  // page 1
  local lineField0 := 0;
  local lineField1 := 1;
  local lineField2 := 2;
  local lineField3 := 3;
  local lineField4 := 4;
  local lineField5 := 5;
  local lineField6 := 6;
  // page 2
  local lineField7 := 7;
  local lineField8 := 8;
  // ...
```

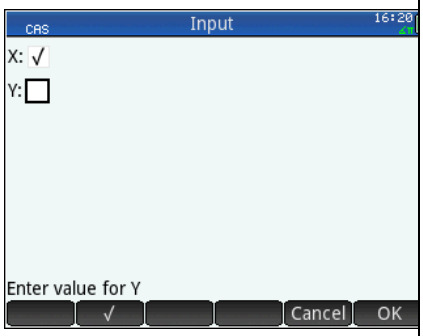




```
  local ObjectType_Real :=  0;
  local array1 := [[1,2],[3,4]];
  input(
  {
  {array1[1,1], [ ObjectType_Real ], { offsetField15, widthField15, l
ineField0} },
  {array1[1,2], [ ObjectType_Real ], { offsetField15, widthField15, l
ineField1} },
  {array1[2,1], [ ObjectType_Real ], { offsetField45, widthField15, l
ineField0} },
  {array1[2,2], [ ObjectType_Real ], { offsetField45, widthField15, l
ineField1} }
  },
  "T» Mod los elementos de un arreglo",
  { "M[1,1]:", "M[1,2]:", "M[2,1]:", "M[2,1]:" }
  );
  return array1;
END;
```

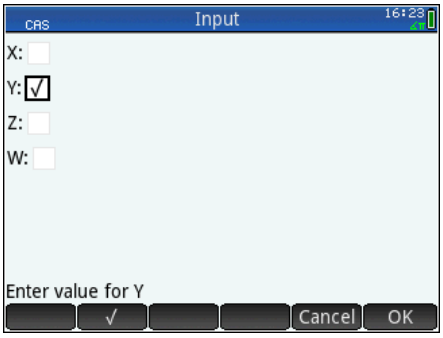
Ejemplo INPUT versión #04

```
export inputcheckv04()
BEGIN
  // Seleccion de una dato por
  local checkbox := 0;
  local checkbox_true := true;
  local checkbox_false := false;
  input(
  {
  { X, checkbox },
  { Y, checkbox }
  }
  );
  return [ X, Y ];
  // las teclas [±] ó [enter] ó [CHECK] cambian el estado de la ca
silla
END;
```



Ejemplo INPUT versión #4a

```
export inputcheckv4a()
BEGIN
  // AGRUPANDO CAMPOS, solo un campo se puede chulear
  local checkbox := 0;
  local checkbox_true := true;
  local checkbox_false := false;
  local group2_checkbox := 2;
  local group3_checkbox := 3;
  local group4_checkbox := 4;
  input(
  {
  { X,group2_checkbox },
  { Y,checkbox },
  { Z,checkbox },
  { W,checkbox }
  }
  );
  return [ X, Y, Z, W ];
END;
```



Ejemplo INPUT versión #05

```
export posList := 2;
export inputDropDownv05()
BEGIN
  // Selección de una dato por menú desplegable
  local list1 := { 1, 2, 3, 4 };
  input( {{ posList, list1 }},
  "Titulo» menú desplegable",
  "list"
  );
  return "opción "+posList+": "+list1[posList];
END;
```

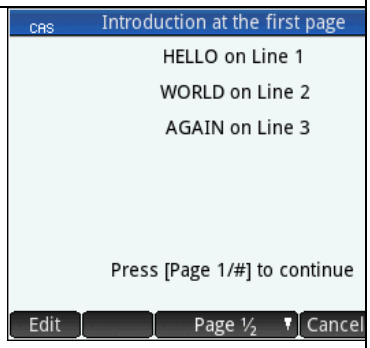
Ejemplo INPUT versión #6

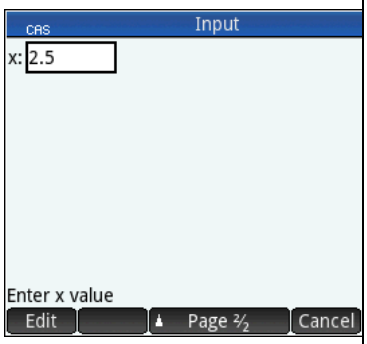
```
export inputIntroduction()
BEGIN
  // truco para mostrar una introducción

  local offsetField0 := 0; //0%
  local offsetField5 := 5; //10%
  local offsetField10 := 10; //%
  local offsetField15 := 15; //%
  local offsetField20 := 20; //%
  local offsetField25 := 25; //%
  local offsetField30 := 30; //%
  local offsetField40 := 40; //%
  local offsetField45 := 45; //%
  local offsetField50 := 50; //%
  //...
  local offsetField100 := 100; //100%
  local widthField0 := 0; //0%
  local widthField10 := 10; //%
  local widthField15 := 15; //%
  local widthField20 := 20; //%
  local widthField25 := 25; //%
  local widthField30 := 30; //%
  local widthField40 := 40; //%
  local widthField50 := 50; //%
  // ...

  // page 1
  local lineField0 := 0;
  local lineField1 := 1;
  local lineField2 := 2;
  local lineField3 := 3;
  local lineField4 := 4;
  local lineField5 := 5;
  local lineField6 := 6;
  // page 2
  local lineField7 := 7;
  local lineField8 := 8;
  // ...

  local ObjectType_Real  :=  0;
  local empty, x := 2.5;
  input(
  {
  { empty, [ ObjectType_Real ], { offsetField100, widthField0, lineFie
```





```
ld0 } },
  { empty, [ ObjectType_Real ], { offsetField100, widthField0, lineFie
ld1 } },
  { empty, [ ObjectType_Real ], { offsetField100, widthField0, lineFie
ld2 } },
  { empty, [ ObjectType_Real ], { offsetField100, widthField0, lineFie
ld3 } },
  { empty, [ ObjectType_Real ], { offsetField100, widthField0, lineFie
ld4 } },
  { empty, [ ObjectType_Real ], { offsetField100, widthField0, lineFie
ld5 } },
  { empty, [ ObjectType_Real ], { offsetField100, widthField0, lineFie
ld6 } },
  { x, [ ObjectType_Real ], { offsetField5, widthField20, lineField7 }
 }
  },
  { "Introduction at the first page", "Input" },
  { "HELLO on Line 1                          ", "WORLD on Line 2
                ",
"AGAIN on Line 3                          ", "", "", "",
"Press [Page 1/#] to continue               ", "x:" },
  { "", "", "", "", "", "", "", "Enter x value" }
  );
  return "x: "+x;
END;
```

Ejemplo INPUT versión #

```
export testInputString()
begin
local a, b, c;
local resetVAL_a, resetVAL_b, resetVAL_c;
local initVAL_a, initVAL_b, initVAL_c;
initVAL_a := "abc"; initVAL_b := "123"; initVAL_c := "\"X\"";
resetVAL_a := ""; resetVAL_b := ""; resetVAL_c := "";
local ObjectType_String := 2;
local ObjectType := ObjectType_String;
local title := "Type Allow: String";
local ok := true;
local cancel := false;
local keyPressedOnMenu := cancel;
keyPressedOnMenu := input
(
{
{a, [ObjectType] },
{b, [ObjectType] },
{c, [ObjectType] }
},
title,
{ "label_a:", "label_b:","label_c:" },
{ "help_a", "help_b", "help_c" },
{ resetVAL_a, resetVAL_b, resetVAL_c },
{ initVAL_a, initVAL_b, initVAL_c }
);
if keyPressedOnMenu == ok then
return ({a, b, c});
```

```
else
return "Done";
end;
end;
```

Frases: “Tree with few fruits, but mature / Arbol con pocos frutos, pero maduros” decía uno de los más grandes matemáticos, CARL FRIEDRICH GAUSS cuando una publicación era excelente.

Este primer lanzamiento de esta publicación, es prácticamente un sketch, en cada actualización ira mejorando e inclusive con la ayuda de vosotros, no dudes en contactarme