Guía del editor de programas TI-Nspire™

Vea más información acerca de la tecnología de TI en la ayuda en línea en education.ti.com/eguide.

Información importante

Excepto que se indique expresamente lo contrario en la Licencia que acompaña a un programa, Texas Instruments no ofrece ninguna garantía, ya sea expresa o implícita, incluyendo pero no limitado a cualquier garantía implícita de comerciabilidad y aptitud para un propósito particular, con respecto a cualquier programa o libro materiales disponibles únicamente en la forma en que se encuentren. En ningún caso Texas Instruments será responsable ante nadie por daños especiales, colaterales, incidentales o consecuenciales relacionados con o derivados de la compra o uso de estos materiales y la única y exclusiva responsabilidad de Texas Instruments, independientemente de la forma de acción, no excederá la cantidad establecida en la licencia para el programa. Además, Texas Instruments no será responsable de ninguna reclamación de ningún tipo contra el uso de estos materiales por cualquier otra parte.

El software TI-Nspire™ utiliza Lua como entorno de secuencias de comandos. Para obtener información sobre derechos de autor y licencias, consulte http://www.lua.org/license.html.

El software TI-Nspire™ utiliza Chipmunk Physics versión 5.3.4 como entorno de simulación. Para obtener información sobre la licencia, consulte http://chipmunk-physics.net/release/Chipmunk-5.x/Chipmunk-5.3.4-Docs/.

Microsoft® y Windows® son marcas comerciales registradas de Microsoft Corporation en los Estados Unidos y / o en otros países.

Mac OS®, iPad® y OS X® son marcas registradas de Apple Inc.

Unicode® es una marca registrada de Unicode, Inc. en los Estados Unidos y otros países.

Los productos reales pueden ser ligeramente distintos de las imágenes proporcionadas.

Contents

Cómoutilizar el Editor de Programas

Puede crear funciones o programas definidos por el usuario ingresando enunciados de definiciones en la línea de ingreso de la Calculadora o utilizando el Editor de Programas. El Editor de Programas le ofrece algunas ventajas, y se cubre en esta sección. Para obtener más información, consulte la sección *Calculadora*.

- El editor tiene plantillas de programación y cuadros de diálogo para ayudarle a definir funciones y programas utilizando la sintaxis correcta.
- El editor le permite ingresar enunciados de programación de varias líneas sin requerir una secuencia de teclas especial para agregar cada línea.
- Usted puede crear con facilidad objetos de librería privada y pública (variables, funciones y programas). Para obtener más información, consulte *Bibliotecas*.

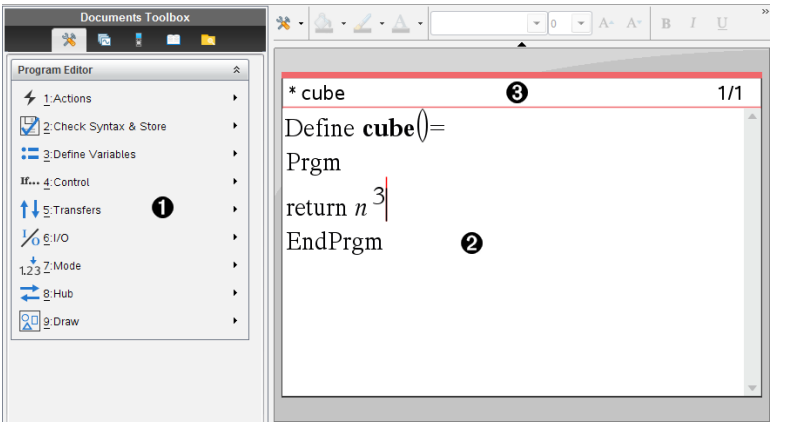
Cómo iniciar el Editor de programas

▶ Para agregar una nueva página de Editor de programas en el problema actual:

En la barra de herramientas, haga clic en **Insertar > Editor de programas> Nuevo**.

Dispositivo portátil: Presione doc, y selecciones **Insertar>Editor de Programas > Nuevo**.

Nota: También puede accederse al editor desde el menú **Funciones & Programas** menú de una página de Calculadora.



❶ Menú Editor de programas: Este menú se encuentra disponible siempre que se encuentre en el área de trabajo del Editor de programas con el modo de vista Normal.

❷ Área de trabajo del Editor de programas

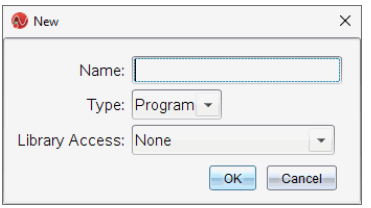
❸ La línea de estado muestra la información del número de línea y el nombre de la función o programa que se define o edita. Un asterisco (*) indica que esta función

está “sucia”, lo que significa que ha cambiado desde la última vez que se verificó su sintaxis y ha sido guardado.

Cómo definir un programa o una función

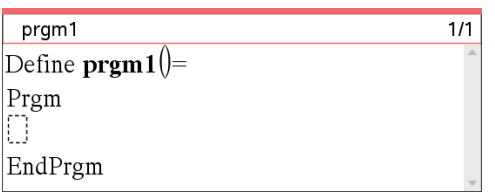
Comenzando con un nuevo Editor de programas

1. Asegúrese de estar en el documento y el problema en donde desea crear el programa o la función.
2. Haga clic en el botón **Insertar** en la barra de herramientas de la aplicación y seleccione **Editor de programas > Nuevo**. (En el dispositivo portátil, presione doc y seleccione **Insertar > Editor de programas > Nuevo**).



3. Escriba un nombre para la función o el programa que va a definir.
4. Seleccione el **Tipo** (**programa** o**función**).
5. Establezca el **Acceso a la biblioteca**:
   - Para usar solo la función o el programa del documento y el problema actual, seleccione **Ninguno**.
   - Para hacer que la función o el programa estén accesibles desde cualquier documento, pero no sean visibles en el Catálogo, seleccione **LibPriv**.
   - Para hacer que la función o el programa estén accesibles desde cualquier documento y visibles en el catálogo, seleccione **LibPub (mostrar en el catálogo)**. Para obtener más información, consulte *Bibliotecas*.
6. Haga clic en **Aceptar**.

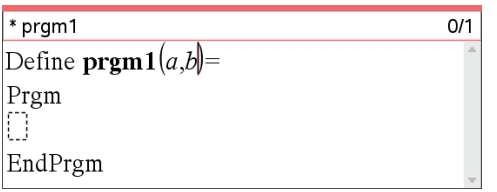
   Se abre una nueva instancia del Editor de programas con una plantilla que concuerda con las selecciones que realizó.



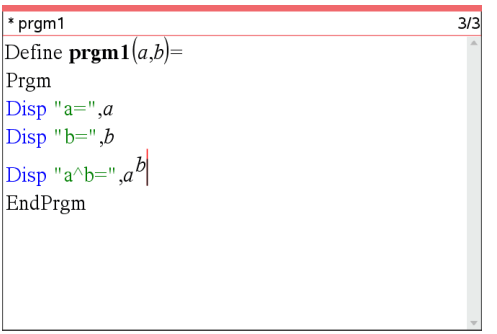
Ingresar líneas en una Función o Programa

El Editor de programas no ejecuta los comando ni evalúa las expresiones a media que las escribe. Solo se ejecutan cuando evalúa la función o ejecuta el programa.

1. Si la función o programa requiere que el usuario proporcione argumentos, escriba los nombres de los parámetros en paréntesis después del nombre. Separe los parámetros con una coma.



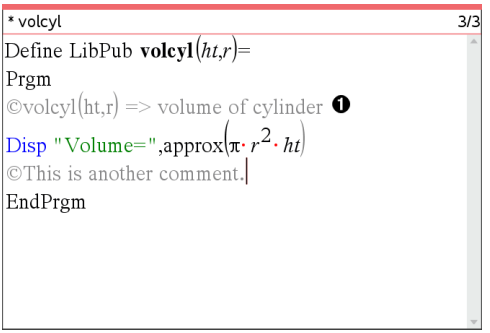
2. Entre las líneas Func y EndFunc (o Prgm y EndPrgm), escriba las líneas de los enunciados que componen su función o programa.



- Puede escribir los nombres de la función y comandos o insertarlos desde el Catálogo.
- Una línea puede ser más larga que el ancho de la pantalla; de ser así, tal vez tenga que desplazarse para ver el enunciado completo.
- Después de escribir cada línea, presione **Intro**. Esto inserta una nueva línea en blanco y le permite agregar otra línea.
- Use las teclas de flecha ◄, ►, ▲ y ▼ para desplazarse por la función o el programa para ingresar o editar los comandos.

Insertar comentarios

Los comentarios pueden ser útiles para alguien que vea o edite el programa. No se muestran cuando se ejecuta el programa y no afectan el flujo del programa. El símbolo © se muestra al inicio de la línea con el comentario.



❶ El comentario muestra que se requiere sintaxis. Como este objeto de la biblioteca es público y este comentario está en la primera línea en un bloque Func o Prgm, el comentario se muestra en el Catálogo como ayuda. Para obtener más información, consulte *Bibliotecas*.

Para insertar un comentario:

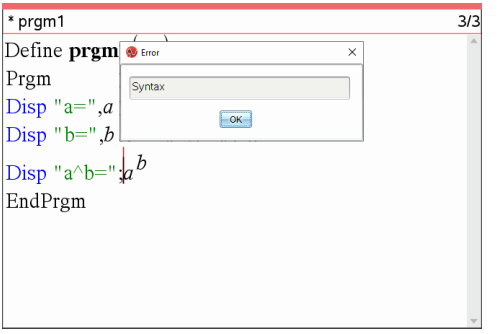
1. Coloque el cursor al final de la línea donde desee insertar un comentario.
2. En el menú **Acciones**, haga clic en **Insertar comentario** o presione **Ctrl+T**.
3. Escriba el texto del comentario después del símbolo ©.

Verificar la sintaxis

El Editor de programas le permite verificar la sintaxis correcta de la función o programa.

- En el menú **Verificar sintaxis y almacenar**, haga clic en **Verificar sintaxis**.

  Si el verificador de sintaxis detecta cualquier error de sintaxis, se muestra un mensaje de error e intenta colocar el cursor cerca del primer error para que usted lo pueda corregir.



Almacenar una función o programa

Debe almacenar la función o programa para hacerlo accesible. El Editor de programas verifica la sintaxis automáticamente antes de almacenar.

Se muestra un asterisco (*) en la esquina superior izquierda del Editor de programas para indicar que la función o el programa no se ha almacenado.

- En el menú **Verificar sintaxis y almacenar**, haga clic en **Verificar sintaxis y almacenar**.

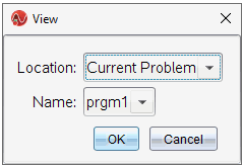
  Si el verificador de sintaxis detecta cualquier error de sintaxis, se muestra un mensaje de error e intenta colocar el cursor cerca del primer error.

  Si no se detectan errores de sintaxis, aparece el mensaje “Stored successfully” en la línea de estado en la parte superior del Editor de programas.

Nota: Si la función o el programa se definen como objetos de biblioteca, también debe guardar el documento en la carpeta de la biblioteca designada y actualizar las bibliotecas para que el objeto sea accesible para otros documentos. Para obtener más información, consulte *Bibliotecas*.

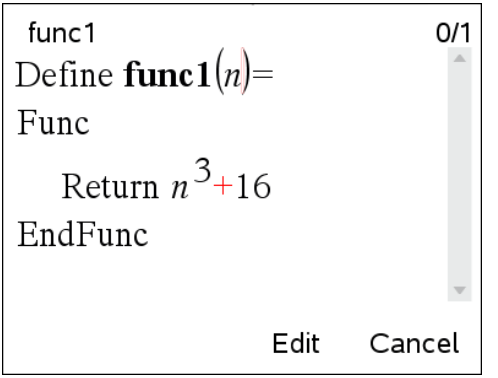
Cómo ver un programa o una función

1. Desde el menú **Acciones** , seleccione **Ver**.



2. Si la función o el programa es un objeto de librería, seleccione su librería desde la lista de **Ubicación** .
3. Seleccione el nombre de la función o del programa desde la lista **Nombre** .

   La función o el programa se despliega en un visor.



4. Use las teclas de flecha para ver la función o el programa.
5. Si usted desea editar el programa, haga clic en **Editar**.

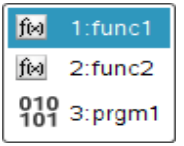
   Nota: La selección **Editar** está disponible únicamente para las funciones y los programas definidos en el problema actual. Para editar un objeto de librería, usted debe abrir primero su documento de librería.

Cómo abrir una función o un programa para edición

Usted puede abrir una función o un programa únicamente desde el problema actual

Nota: Usted no puede modificar un programa o una función bloqueada. Para desbloquear el objeto, vaya a la página de la Calculadora y use el comando **Desbloquear** .

1. Despliegue la lista de funciones y programas disponibles.
   - Desde el menú **Acciones** , seleccione **Abrir**.

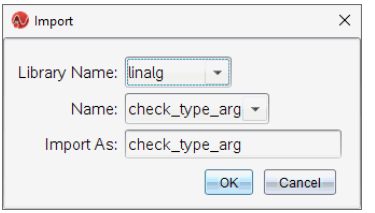


2. Seleccione el elemento a abrir.

Cómo importar un programa desde una librería.

Usted puede importar una función o un programa definido como un objeto de librería al Editor de Programas dentro del problema actual. La copia importada no está bloqueada, incluso si el original está bloqueado.

1. Desde el menú **Acciones** , seleccione **Importar**.



2. Seleccione el **Nombre de Librería**.
3. Seleccione el **Nombre** del objeto.
4. Si desea que el objeto importado tenga un nombre distinto, escriba el nombre bajo **Importar como**.

Cómo crear una copia de una función o un programa

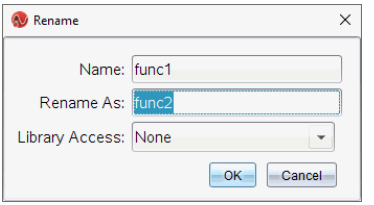
Al crear una nueva función o un nuevo programa, usted podría considerar más fácil comenzar con una copia del actual. La copia que usted crea no está bloqueada, incluso si el original está bloqueado.

1. Desde el menú **Acciones** , seleccione **Crear Copia**.
2. Escriba un nuevo nombre o haga clic en **OK** para aceptar el nombre propuesto.
3. Si usted desea cambiar el nivel de acceso, seleccione **Acceso a Librería**y seleccione un nuevo nivel.

Cómo renombrar un programa o una función

Usted puede renombrar y (en forma opcional) cambiar el nivel de acceso de la función o del programa actual.

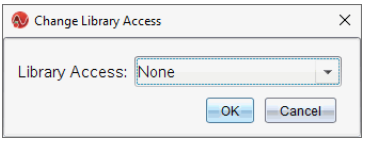
1. Desde el menú **Acciones** , seleccione **Renombrar**.



2. Escriba un nuevo nombre o haga clic en **OK** para aceptar el nombre propuesto.
3. Si usted desea cambiar el nivel de acceso, seleccione **Acceso a Librería**y seleccione un nuevo nivel.

Cómo cambiar el nivel de acceso a librería

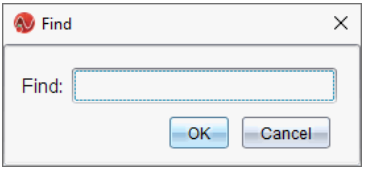
1. Desde el menú **Acciones** , seleccione **Cambiar Acceso a Librería**.



2. Seleccione **Acceso a Librería**:
   - Para usar la función o el programa sólo desde el problema actual de la Calculadora, seleccione **Ninguno**.
   - Para hacer que la función o el programa sea accesible desde cualquier documento pero que no esté visible en el Catálogo, seleccione **LibPriv**.
   - Para hacer que la función o el programa sea accesible desde cualquier documento y que también esté visible en el Catálogo, seleccione **LibPub**.

Cómo encontrar texto

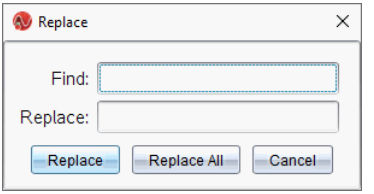
1. Desde el menú **Acciones** , seleccione **Encontrar**.



2. Escriba el texto que desea encontrar y haga clic en **OK**.
   - Si se encuentra el texto, éste se resalta en el programa.
   - Si no se encuentra el texto, se desplegará un mensaje de notificación.

Cómo encontrar y reemplazar texto

1. Desde el menú **Acciones** , seleccione **Encontrar y Reemplazar**.



2. Escriba el texto que desea encontrar.
3. Escriba el texto de reemplazo.
4. Haga clic en **Reemplazar** para reemplazar la primera ocurrencia después de la posición del cursor, o bien haga clic en **Reemplazar Todo** para reemplazar cada ocurrencia.

   Nota: Si el texto se encuentra en una plantilla de matemáticas, se desplegará un mensaje para advertirle que su texto de reemplazo reemplazará la plantilla entera, no tan sólo el texto encontrado.

Cómo cerrar la función o el programa actual.

- Desde el menú **Acciones** , seleccione **Cerrar**.

  Si la función o el programa tiene cambios no almacenados, usted recibe una indicación para revisar la sintaxis y almacenar antes de cerrar.

Cómo ejecutar programas y cómo evaluar funciones

Después de definir y almacenar un programa o función, puede usarlo desde una aplicación. Todas las aplicaciones pueden evaluar funciones, aunque solo las aplicaciones Calculadora y Notas pueden ejecutar programas.

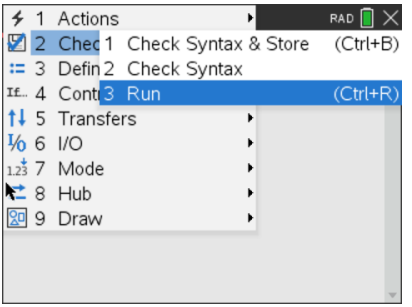
Los enunciados del programa se ejecutan en orden secuencial (aunque algunos comandos alteran el flujo del programa). La salida, si acaso existe una, se muestra en el área de trabajo de la aplicación.

- La ejecución del programa continúa hasta que se alcance el último enunciado o un comando **Stop**.
- La ejecución de la función continúa hasta que alcanza un comando **Return**.

Cómo ejecutar un programa o función desde el Editor de programas

1. Asegúrese de haber definido un programa o una función y que el Editor de programas sea el panel (computadora) o la página (dispositivo portátil) que se activó.
2. En la barra de herramientas, haga clic en el botón **Herramientas de documentos** y seleccione **Verificar sintaxis y almacenar > Ejecutar**.
   — o bien—
   Presione **Ctrl+R**.

   Dispositivo portátil: Presione menu 2 3 o presione ctrl R.



   Esto automáticamente hará lo siguiente:

   - Verificará la sintaxis y almacenará el programa o la función,
   - Pegará el nombre del programa o la función en la primera línea disponible de la aplicación Calculadora inmediatamente después del Editor de programas. Si no existe la Calculadora en esa posición, se insertará una nueva.

3. Si el programa o la función requiere que suministre uno o más argumentos, escriba los valores o los nombres de las variables dentro de paréntesis.
4. Presione enter.

Nota: También puede ejecutar un programa o función en las aplicaciones Calculadora o Notas si escribe el nombre del programa en paréntesis y cualquier otro argumento requerido y luego presione enter.

Cómo usar los nombres cortos y largos

Siempre que se encuentre en el mismo problema donde se define un objeto, puede tener acceso a él si escribe su nombre corto (el nombre que se le da al objeto cuando en el comando **Define**). Es el mismo caso para todos los objetos definidos, incluso los objetos de bibliotecas privadas, públicas y objetos que no son de biblioteca.

Puede tener acceso al objeto de la biblioteca desde cualquier documento al escribir el nombre largo del objeto. El nombre largo consiste en el nombre del documento de la biblioteca de objetos, seguido de una barra diagonal inversa "\" seguida del nombre del objeto. Por ejemplo, el nombre largo del objeto definido como **func1** en el documento de biblioteca **lib1** es **lib1\func1**. Para escribir el carácter "\" en el dispositivo portátil, presione shift ÷.

Nota: Si no recuerda el nombre exacto o el orden de argumentos requerido para un objeto de una biblioteca privada, puede abrir el documento de la biblioteca o usar el Editor de programas para ver el objeto. También puede usar **getVarInfo** para ver una lista de los objetos en una biblioteca.

Cómo utilizar un programa o una función de una biblioteca pública

1. Asegúrese de haber definido el objeto en el primer problema del documento, almacenado el objeto, guardado el documento de biblioteca en la carpeta MyLib y actualizado las bibliotecas.
2. Abra la aplicación TI-Nspire™ en la que desea usar el programa o la función.

   Nota: Todas las aplicaciones pueden evaluar funciones, aunque solo las aplicaciones Calculadora y Notas pueden ejecutar programas.
3. Abra el Catálogo y utilice la pestaña de la biblioteca para encontrar e insertar el objeto.
   — o bien—
   Escriba el nombre del objeto. En el caso de funciones o programas, siempre agregue paréntesis después del nombre.

   ```
   libs2\func1()
   ```
4. Si el programa o la función requiere que suministre uno o más argumentos, escriba los valores o los nombres de las variables dentro de paréntesis.

   ```
   libs2\func1(34,energía)
   ```
5. Presione enter.

Cómo usar un programa o una función de una biblioteca privada

Para usar el objeto de biblioteca privada, debe conocer su nombre largo. Por ejemplo, el nombre largo del objeto definido como **func1** en el documento de biblioteca **lib1** es **lib1\func1**.

Nota: Si no recuerda el nombre exacto o el orden de argumentos requerido para un objeto de una biblioteca privada, puede abrir el documento de la biblioteca o utilizar el Editor de programas para ver el objeto.

1. Asegúrese de haber definido el objeto en el primer problema del documento, almacenado el objeto, guardado el documento de biblioteca en la carpeta MyLib y actualizado las bibliotecas.
2. Abra la aplicación TI-Nspire™ en la que desea usar el programa o la función.

   Nota: Todas las aplicaciones pueden evaluar funciones, aunque solo las aplicaciones Calculadora y Notas pueden ejecutar programas.
3. Escriba el nombre del objeto. En el caso de funciones o programas, siempre agregue paréntesis después del nombre.

   ```
   libs2\func1()
   ```
4. Si el objeto requiere que suministre uno o más argumentos, escriba los valores o los nombres de las variables dentro del paréntesis.

   ```
   libs2\func1(34,energía)
   ```
5. Presione enter.

Cómo interrumpir un programa o una función que se está ejecutando

Mientras un programa o una función se está ejecutando, se muestra el puntero ocupado ⏲.

- Para detener un programa o función,
  - Windows®: Mantenga presionada la tecla **F12** y presione **Intro** varias veces.
  - Mac®: Mantenga presionada la tecla **F5** y presione **Intro** varias veces.
  - Dispositivo portátil: Mantenga presionada la tecla ⌂ on y presione enter varias veces.

  Se muestra un mensaje. Para editar el programa o la función en el Editor de programas, seleccione **Ir a**. El cursor aparece en el comando donde ocurrió el salto.

Cómo insertar valores en un programa

Usted puede elegir entre varios métodos para proporcionar los valores que usa una función o un programa en los cálculos.

Cómo incrustar los valores dentro del programa o la función

Este método es útil principalmente para los valores que deben ser los mismos cada vez que se usa el programa o la función.

1. Cómo definir el programa.

```
* calculatearea                    4/4
Define calculatearea()=
Prgm
w:=3
h:=23.64
area:=w·h
Disp area
EndPrgm
```

2. Ejecute el programa.

```
calculatearea()
                                  70.92
                                  Done
```

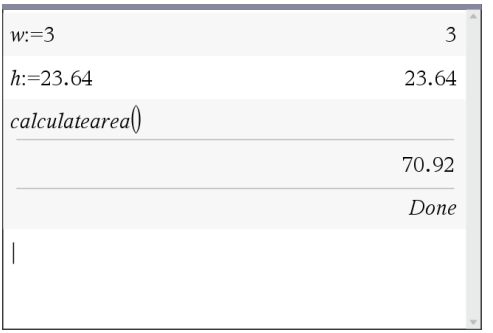
Cómo permitir que el usuario asigne los valores a las variables

Un programa o una función puede referirse a variables creadas con anterioridad. Este método requiere que los usuarios recuerden los nombres de variables y que asignen valores a los mismos antes de usar el objeto.

1. Cómo definir el programa.

```
* calculatearea                    2/2
Define calculatearea()=
Prgm
area:=w·h
Disp area
EndPrgm
```

2. Proporcione las variables y luego ejecute el programa.

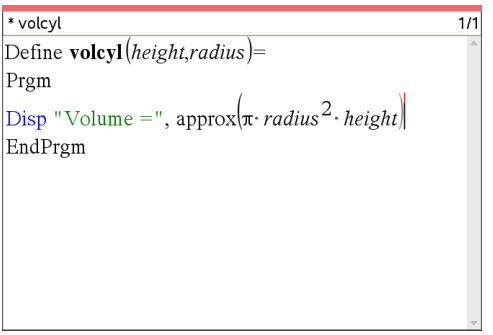


Cómo permitir que el usuario proporcione los valores como argumentos

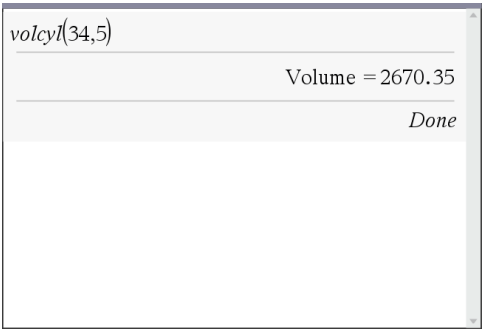
Este método permite que los usuarios pasen uno o más valores como argumentos dentro de la expresión que llama al programa o a la función.

El siguiente programa, **volcyl**, calcula el volumen de un cilindro. Requiere que el usuario proporcione dos valores: altura y radio del cilindro.

1. Defina el programa **volcyl**.



2. Ejecute el programa para desplegar el volumen de un cilindro con una altura de 34 mm y un radio de 5 mm.



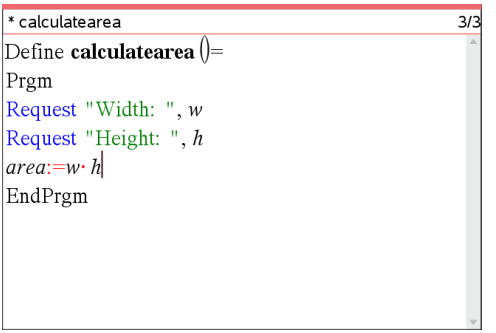
Nota: Usted no tiene que usar los nombres de parámetro cuando ejecute el programa **volcyl** aunque deberá proporcionar dos argumentos (como valores, variables o expresiones). El primero debe representar la altura y el segundo debe representar el radio.

Cómo solicitar los valores del usuario (sólo programas)

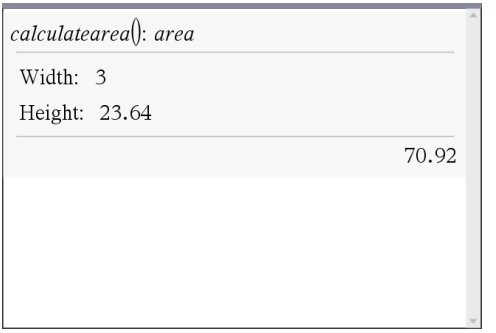
Usted puede usar los comandos **Request** y **RequestStr** en un programa para hacer que el programa entre en pausa y despliegue un cuadro de diálogo indicándole al usuario que proporcione información. Este método no requiere que los usuarios recuerden los nombres de variable ni el orden en el que se necesitan.

Usted no puede usar el comando **Request** o **RequestStr** en una función.

1. Cómo definir el programa.



2. Ejecute el programa y responda a las solicitudes.

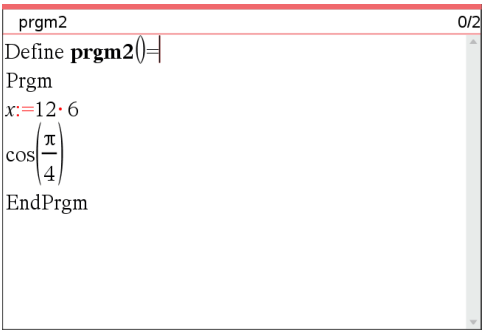


Use **RequestStr** en lugar de **Request** cuando usted desee que el programa interprete la respuesta del usuario como una cadena de caracteres en lugar de como una expresión matemática. Esto evita solicitar que el usuario encierre la respuesta entre comillas (“”).

Cómo desplegar información

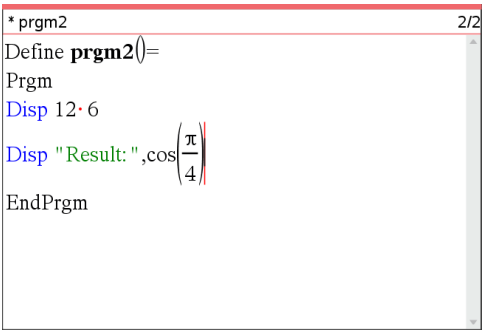
Una función o un programa en ejecución no despliega resultados calculados intermedios, a menos que usted incluya un comando para desplegarlos. Esta es una diferencia importante entre realizar un cálculo en la línea de ingreso y realizarlo en una función o un programa.

Por ejemplo, los siguientes cálculos no despliegan un resultado en una función o un programa (aunque sí lo hacen desde la línea de ingreso).



Cómo desplegar información en el historial

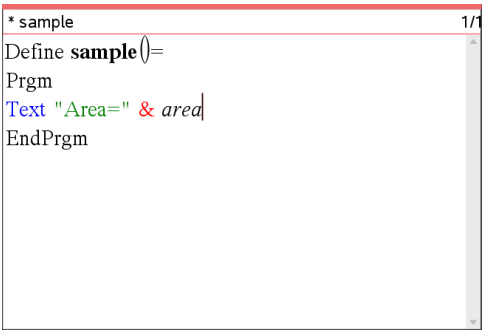
Usted puede usar el comando **Disp** en un programa o una función para desplegar información, incluyendo resultados intermedios, en el historial.



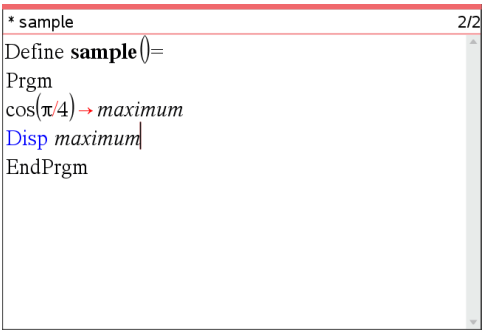
Cómo desplegar información en un cuadro de diálogo

Usted puede usar el comando **Text** para pausar un programa en ejecución y desplegar información en el cuadro de diálogo. El usuario selecciona **OK** para continuar o selecciona **Cancelar** para detener el programa.

Usted no puede usar el comando **Text** en una función.



Nota: Desplegar un resultado con **Disp** o **Text** no almacena ese resultado. Si usted espera consultar un resultado más adelante, almacénelo en una variable global.

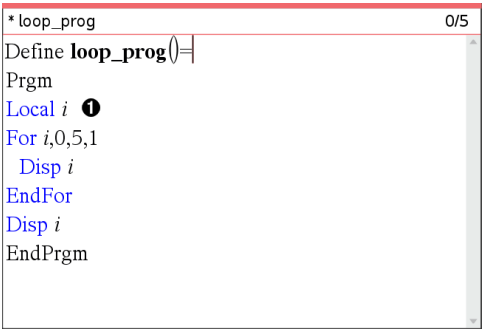


Cómo usar variables locales

Una variable local es una variable temporal que existe sólo mientras se está evaluando una función definida por el usuario o cuando se está ejecutando un programa definido por el usuario.

Ejemplo de una variable local

El siguiente segmento de programa muestra un bucle **For...EndFor** (el cual se analiza más adelante en este módulo). La variable *i* es el contador de bucles. En la mayoría de los casos, la variable *i* se usa sólo mientras el programa se está ejecutando.



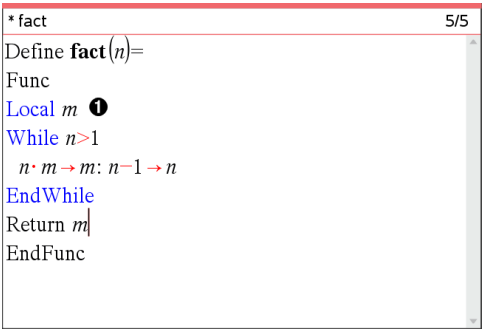
❶ Declara la variable *i* como local.

Nota: Cuando sea posible, declare como local cualquier variable que se use sólo dentro del programa y que no necesite estar disponible después de que el programa se detenga.

¿Qué causa un mensaje de error de variable indefinida?

Se desplegará un mensaje de error de variable **Indefinida** cuando usted evalúe una función definida por el usuario o cuando ejecute un programa definido por el usuario que se refiera a una variable local que no se ha inicializado (asignado un valor).

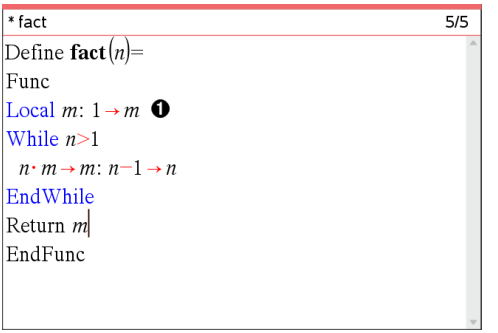
Por ejemplo:



❶ La variable local *m* no tiene asignado un valor inicial.

Inicialice las variables locales

A todas las variables locales se les debe asignar un valor inicial antes de que se referencien.



❶ 1 se almacena como el valor inicial para *m*.

Nota (CAS): Las funciones y los programas no se pueden usar como una variable local para realizar cálculos simbólicos.

CAS: Cómo realizar cálculos simbólicos

Si usted desea que una función o un programa realice cálculos simbólicos, deberá usar una variable global en lugar de una local. Sin embargo, usted debe estar seguro de que la variable global no exista ya fuera del programa. Los siguientes métodos le pueden ayudar.

- Refiérase a un nombre de variable global, por lo general con dos o más caracteres, que no sea probable que exista fuera de la función o del programa.
- Incluya **DelVar** dentro de un programa para borrar la variable global, si es que existe, antes de referirse a ella. (**DelVar** no borra las variables bloqueadas o enlazadas).

Diferencias entre funciones y programas

Una función definida en el Editor de Programas es similar a las funciones que se crean en el software TI-Nspire™ .

- Las funciones deben regresar un resultado, el cual se puede graficar o ingresar en una tabla. Los programas no pueden regresar un resultado.
- Usted puede usar una función (pero no un programa) dentro de una expresión. Por ejemplo: **3 • func1(3)** es válido, pero no **3 • prog1(3)**.
- Puede ejecutar programas solamente desde las aplicaciones Calculadora y Notas. Sin embargo, puede evaluar funciones en Calculadora, Notas, Listas y Hoja de Cálculo, Gráficos y Geometría y Datos y Estadísticas.

- Una función se puede referir a cualquier variable; sin embargo, puede almacenar un valor únicamente en una variable local. Los programas se pueden almacenar en variables locales y globales.

  Nota: Los argumentos que se usan para pasar valores a una función se tratan como variables locales en forma automática. Si usted desea almacenar en cualquier otra variable, deberá declararlas como **Local** desde dentro de la función.

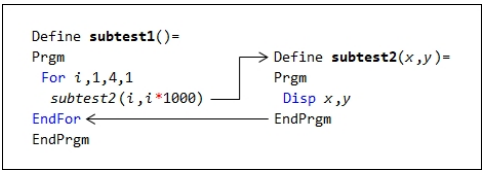
- Una función no puede llamar a un programa como una subrutina, pero sí puede llamar a otra función definida por el usuario.
- Usted no puede definir un programa dentro de una función.
- Una función no puede definir una función global, pero sí puede definir una función local.

Cómo llamar un programa desde otro

Un programa puede llamar otro programa como una subrutina. La subrutina puede ser externa (un programa independiente) o interna (incluida en el programa principal). Las subrutinas son útiles cuando un programa necesita repetir el mismo grupo de comandos en varios lugares diferentes.

Cómo llamar un programa independiente

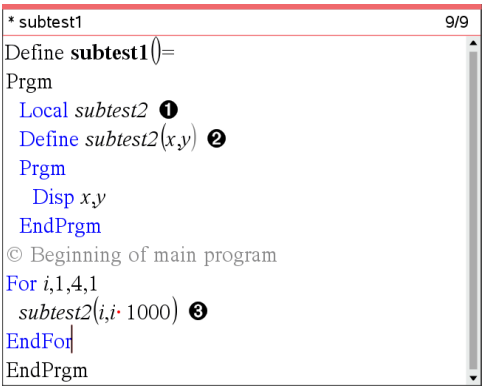
Para llamar un programa independiente, use la misma sintaxis que usted usa para ejecutar el programa desde la línea de ingreso.



Cómo definir y llamar a una subrutina interna

Para definir una subrutina interna, use el comando **Definir** con **Prgm...TerminarPrgm**. Dado que una subrutina se debe definir antes de que se pueda llamar, es una buena práctica definir las subrutinas al principio del programa principal.

Una subrutina interna se llama y ejecuta de la misma manera que un programa independiente.



❶ Declara la subrutina como una variable local.

❷ Define la subrutina.

❸ Llama a la subrutina.

Nota: Use el menú **Var** del Editor de Programas para ingresar los comandos **Definir** y **Prgm...TerminarPrgm** .

Notas acerca de cómo usar las subrutinas

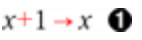
Al final de una subrutina, la ejecución regresa al programa que llama. Para salir de una subrutina en cualquier otro momento, use **Return** sin ningún argumento.

Una subrutina no puede acceder a las variables locales declaradas en el programa que llama. Asimismo, el programa que llama no puede acceder a las variables locales declaradas en una subrutina.

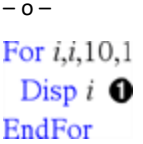
Lbl son locales para los programas en los cuales se localizan. Por lo tanto, un comando de **Goto** en el programa que llama no se puede ramificar en una etiqueta en una subrutina o viceversa.

Cómo evitar errores de definición circular

Cuando se evalúa una función definida por el usuario o se ejecuta un programa, usted puede especificar un argumento que incluya la misma variable que se usó para definir la función o para crear el programa. No obstante, para evitar errores de definición circular, usted debe asignar un valor para las variables que se usan al evaluar la función o al ejecutar el programa. Por ejemplo:



– o –



❶ Causa un mensaje de error de **definición Circular** si x o i no tiene un valor. El error no ocurre si a x o i ya se les ha asignado un valor.

Cómo controlar el flujo de una función o un programa

Cuando usted ejecuta un programa o evalúa una función, las líneas del programa se ejecutan en orden secuencial. Sin embargo, algunos comandos alteran el flujo del programa. Por ejemplo:

- Las estructuras de control como los comandos **If...EndIf** usan una prueba condicional para decidir qué parte de un programa ejecutar.
- Los comandos de bucle como **Para...TerminarPara** repiten un grupo de comandos.

Cómo usar If, Lbl e Goto a para controlar el flujo del programa

El comando **Si** y varias estructuras de **For...EndFor** le permiten ejecutar una sentencia o bloque de sentencias en forma condicional; esto es, con base en el resultado de una prueba (como x>5). **Lbl** (etiqueta) e **Goto** le permiten ramificar, o saltar, desde un lugar hasta otros en una función o un programa.

El comando **If** y varias estructuras **If...EndIf** residen en el menú de **Control** del Editor de Programas.

Cuando usted inserta una estructura como **If...Then...EndIf**, se inserta una plantilla en la ubicación del cursor. El cursor se posiciona de manera que usted puede ingresar una prueba condicional.

Comando If

Para ejecutar un comando sencillo cuando una prueba condicional es verdadera, use la forma general:

```
If x>5
  Disp "x is greater than 5" ❶
Disp x ❷
```

❶ Se ejecuta sólo si x>5; de otro modo, se salta.

❷ Siempre despliega el valor de x.

En este ejemplo, usted debe almacenar un valor para x antes de ejecutar el comando **If** .

Estructuras If...Then...EndIf

Para ejecutar un grupo de comandos si una prueba condicional es verdadera, use la estructura:

```
If x>5 Then
  Disp "x is greater than 5 " ❶
  2·x→x ❶
EndIf
Disp x ❷
```

❶ Se ejecuta sólo si x>5.

❷ Despliega el valor de:
2x si x>5
x si x≤5

Nota: EndIf marca el final del bloque **Then** que se ejecuta si la condición es verdadera.

Estructuras If...Then...Else...EndIf

Para ejecutar un grupo de comandos si una prueba condicional es verdadera y un grupo diferente si la condición es falsa, use esta estructura:

```
If x>5 Then
  Disp "x is greater than 5 " ❶
  2·x→x ❶
Else
  Disp "x is less than or equal to 5 " ❷
  5·x→x ❷
EndIf
Disp x ❸
```

❶ Se ejecuta sólo si x>5.

❷ Se ejecuta sólo si x≤5.

❸ Despliega el valor de:
2x si x>5
5x si x≤5

Estructuras If...Then...ElseIf... EndIf

Una forma más compleja del comando **If** le permite probar varias condiciones. Supongamos que usted desea un programa para probar un argumento suministrado por el usuario que significa una de cuatro opciones.

Para probar cada opción (Si Opción=1, Si Opción=2 y así sucesivamente), use la estructura **If...Then...ElseIf...EndIf**.

Comandos Lbl y Goto

Usted también puede controlar el flujo al usar los comandos **Lbl** (etiqueta) e **Goto** (Ir a) . Estos comandos residen en el menú **Transferencias** del Editor de Programas.

Use el comando **Lbl** para etiquetar (asignar un nombre a) una ubicación en particular en la función o el programa.

Lbl *nombre de Etiqueta*	nombre para asignar a esta ubicación (use la misma norma de nombrado como un nombre de variable)

Entonces usted puede usar el comando **Ir a** en cualquier punto de la función o del programa para ramificar hacia la ubicación que corresponde a la etiqueta especificada.

Goto *nombre de Etiqueta*	especifica a cuál comando **Lbl** a ramificar

Dado que un comando **Goto** es incondicional (siempre se ramifica a la etiqueta especificada), con frecuencia se usa con un comando **If** de manera que usted puede especificar una prueba condicional. Por ejemplo:

```
If x>5
  Goto GT5 ❶
Disp x

.....

Lbl GT5 ❷
Disp "The number was > 5"
```

❶ Si x>5, se ramifica directamente a la etiqueta GT5.

❷ Para este ejemplo, el programa debe incluir comandos (como **Stop**) que previenen que **Lbl** GT5 se ejecute si x≤5.

Cómo usar bucles para repetir un grupo de comandos.

Para repetir el mismo grupo de comandos en forma sucesiva, use una de las estructuras de bucle. Hay varios tipos de bucles disponibles. Cada tipo le brinda una manera distinta de salir del bucle, con base en una prueba condicional.

Los comandos de bucle y relacionados con bucle residen en los menús **Control** y **Transferencias** del Editor de Programas.

Cuando usted inserta una de las estructuras de bucle, se inserta su plantilla en la ubicación del cursor. Entonces usted puede comenzar a ingresar los comandos que se ejecutarán dentro del bucle.

Bucles For...EndFor (Para...TerminarPara)

Un bucle **For...EndFor** usa un contador para controlar el número de veces que se repite el bucle. La sintaxis del comando **For** es:

Nota: El valor final puede ser menor que el valor inicial, siempre que el incremento sea negativo.

Para *variable*, *iniciar*, *terminar* [, *incremento*]

❶ ❷ ❸ ❹

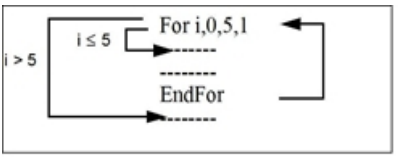
❶ *Variable* usada como un contador

❷ El valor del contador usado por primera vez **For** se ejecuta

❸ Sale del bucle cuando la *variable* excede este valor

❹ Se agrega al contador cada vez subsiguiente que se ejecuta **For** (Si este valor opcional se omite, el *incremento* es 1.)

Cuando se ejecuta **For** , el valor de la *variable* se compara con el valor *final* . Si la *variable* no excede el *final*, se ejecuta el bucle; de otra manera, el control salta al comando después de **EndFor**.



Nota: El comando **For** incrementa en forma automática la variable del contador, de manera que la función o el programa puede salir del bucle después de un cierto número de repeticiones.

Al final del bucle (**EndFor**), el control salta de regreso al comando **For** , donde la variable se incrementa y compara con el *final*.

Por ejemplo:

```
For i,0,5,1
  Disp i ❶
EndFor
Disp i ❷
```

❶ Despliega 0, 1, 2, 3, 4 y 5.

❷ Despliega 6. Cuando la *variable* se incrementa a 6, el bucle no se ejecuta.

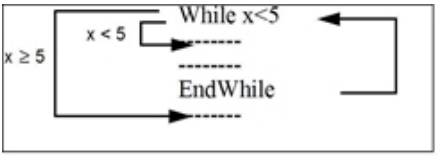
Nota: Usted puede declarar la variable del contador como local si no necesita guardarse después de que la función o el programa se detiene.

Bucles While...EndWhile (Mientras...TerminarMientras)

Un bucle **While...EndWhile** repite un bloque de comandos siempre y cuando una condición específica sea verdadera. La sintaxis del comando **While** es:

While *condición*

Cuando se ejecuta **While** , *la condición* se evalúa. Si la *condición* es verdadera, se ejecuta el bucle; de otra manera, el control salta al comando después de **EndWhile**.



Nota: El comando **While** no cambia la condición en forma automática. Usted deberá incluir comandos que permitan que la función o el programa salga del bucle.

Al final del bucle (**EndWhile**), el control salta de regreso al comando **While** , donde la condición se vuelve a evaluar.

Para ejecutar el bucle por primera vez, la condición debe ser verdadera inicialmente.

- Cualquier variable referenciada en la condición debe estar configurada antes del comando **While** . (Usted puede crear los valores en la función o el programa, o bien puede indicarle al usuario que ingrese los valores).
- El bucle debe contener comandos que cambien los valores en la condición, que a la larga causen que sea falsa. De otro modo, la condición siempre será verdadera y la función o el programa no podrá salir del bucle (llamado bucle infinito).

Por ejemplo:

```
0→x  ❶
While x<5
  Disp x  ❷
  x+1→x  ❸
EndWhile
Disp x  ❹
```

❶ Configura x inicialmente.

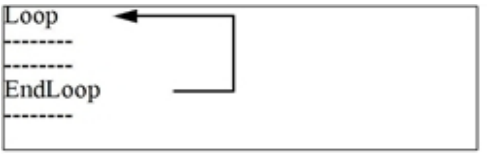
❷ Despliega 0, 1, 2, 3 y 4.

❸ Incrementa x.

❹ Despliega 5. Cuando x se incrementa a 5, el bucle no se ejecuta.

Bucles Loop...EndLoop (Bucle...TerminarBucle)

Un **Loop...EndLoop** crea un bucle infinito, el cual se repite sin parar. El comando **Loop** no tiene ningún argumento.



Por lo general, usted inserta comandos en el bucle que permiten que el programa salga del bucle. Los comandos que se usan comúnmente son: **If**, **Exit**, **Goto**, y **Lbl** (etiqueta). Por ejemplo:

```
0→x
Loop
  Disp x
  x+1→x
  If x>5 ❶
    Exit
EndLoop
Disp x ❷
```

❶ Un comando **If** revisa la condición.

❷ Sale del bucle y salta hasta aquí cuando x se incrementa a 6.

Nota: El comando **Exit** sale del bucle actual.

En este ejemplo, el comando **If** puede estar en cualquier parte del bucle.

Cuando el comando If es:	El bucle es:
En el inicio del bucle	Se ejecuta sólo si la condición es verdadera.
Al final del bucle	Se ejecuta al menos una vez y se repite sólo si la condición es verdadera.

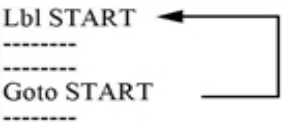
El comando **If** también podría usar un comando **Goto** para transferir el control del programa a un comando de **Lbl** (etiqueta) especificado.

Cómo repetir un bucle de inmediato

El comando **Cycle** transfiere de inmediato el control del programa a la siguiente iteración de un bucle (antes de que la iteración actual esté completa). Este comando funciona con **For...EndFor**, **While...EndWhile** y **Loop...EndLoop**.

Bucles Lbl y Goto

A pesar de que los comandos **Lbl** (etiqueta) e **Goto** no son estrictamente comandos de bucle, se pueden usar para crear un bucle infinito. Por ejemplo:



Al igual que con **Loop...EndLoop**, el bucle deberá contener comandos que permitan que la función o el programa salga del bucle.

Cómo cambiar las configuraciones del modo

Las funciones y los programas pueden usar la función **setMode()** para configurar en forma temporal los modos de cálculo o resultado específico. El menú **Modo** del Editor de Programas facilita el ingreso de la sintaxis correcta sin requerir que usted memorice códigos numéricos.

Nota: Los cambios de modo que se hacen dentro de una definición de función o programa no persisten afuera de la función o el programa.

Cómo configurar un modo

1. Posicione el cursor donde usted desea insertar la función **setMode** .
2. Desde el menú **Modo** , seleccione el modo a cambiar y seleccione la nueva configuración.

   La sintaxis correcta se inserta en la ubicación del cursor. Por ejemplo:

```
setMode(1,3)
```

Cómo depurar programas y manejar errores

Después de que usted escribe una función o un programa, podrá usar varias técnicas para encontrar y corregir errores. Usted también puede construir un comando de manejo de errores en la función o en el programa en sí.

Si su función o programa permite que el usuario seleccione de entre varias opciones, asegúrese de correrlo y de probar cada opción.

Técnicas para depurar

Los mensajes de error en tiempo de ejecución pueden localizar errores de sintaxis, pero no los errores en la lógica del programa. Las siguientes técnicas pueden ser útiles.

- Inserte temporalmente comandos **Disp** para desplegar los valores de las variables críticas.
- Para confirmar que un bucle se ejecuta el número de veces correcto, use **Disp** para desplegar la variable del contador o los valores en la prueba condicional.

- Para confirmar que una subrutina se ejecuta, use **Disp** para desplegar mensajes como “Ingresando a subrutina” y “Saliendo de subrutina” al inicio y al final de la subrutina.
- Para detener un programa o función de forma manual:
  - **Windows®:** Mantenga presionada la tecla **F12** y presione **Enter** varias veces.
  - **Macintosh®:** Mantenga presionada la tecla **F5** y presione **Enter** varias veces.
  - **Dispositivo portátil:** Mantenga presionada la tecla [on] y presione [enter] varias veces.

Comandos de manejo de errores

Comando	Descripción
Try...EndTry	Define un bloque que permite que una función o un programa ejecute un comando y, si es necesario, que se recupere de un error generado por ese comando.
ClrErr	Borra el estado de error y establece la variable *errCode del sistema en cero. Para ver un ejemplo del uso de errCode, consulte el comando* ***Try en la Guía de Referencia****.*
PassErr	Pasa un error al siguiente nivel del bloque **Try...EndTry**.

Información general

Ayuda en línea

education.ti.com/eguide

Seleccione su país para obtener más información del producto.

Comuníquese con Asistencia de TI

education.ti.com/ti-cares

Seleccione su país para obtener recursos técnicos y otro tipo de ayuda.

Información sobre el servicio y la garantía

education.ti.com/warranty

Seleccione su país para obtener información acerca de la duración de los términos de la garantía o sobre el servicio para productos.

Garantía limitada. Esta garantía no afecta a sus derechos legales.

Texas Instruments Incorporated

12500 TI Blvd.

Dallas, TX 75243