# Uso de variables en Maple

José Luis Torres Rodríguez*

Enero 2011

Maple nos permite asignar a cualquier tipo de dato, estructura o resultado, un nombre formado por una palabra sintácticamente válida. Este tipo de expresiones se conocen con el nombre de ***"variables"***. Las variables son útiles ya que nos permiten ***"almacenar"*** datos para su uso posterior. Por ejemplo, considérese que para hacer referencia a resultados anteriores por medio de los operadores "%", "%%" "%%%", no podemos referenciar más alla del antepenúltimo valor; en cambio, al utilizar variables, podemos almacenar un dato a lo largo de toda una sesión y utilizarlo, modificarlo o eliminarlo en cualquier momento, sin importar cuando fue generado.

## 1. Variables

Existen varios detalles que deben tenerse en cuenta al utilizar variables; respecto a la definición, nombres que se pueden usar, forma de asignar datos y de ***"desasignar"*** una variable previamente creada.

### 1.1. Definición de una variable

Para poder utilizar una expresión como variable y asignarle un dato o alguna otra expresión, se utiliza el operador **":="**, de la siguiente forma:

**expresión1 := dato o expresión2;**

donde **"expresión1"** es el nombre que le daremos a la variable. Por ejemplo, asignemos a la expresión **"a"** el valor **"34"**.

```
> a := 34;
```

$$a := 34$$

Una vez que se crea una variable, el valor asignado a ésta permanecerá inalterado hasta que se le asigne uno nuevo. Además, cada vez que hacemos referencia a una variable, Maple automáticamente invoca al último valor que se le asignó. Por ejemplo:

```
> a;
```

$$34$$

Nótese que Maple nos muestra como resultado el valor asignado. De la misma forma, al incluir el nombre de una variable dentro de una expresión algebraica, Maple evalua esta expresión tomando en cuenta el valor que contiene la variable.

```
> 3*a + 25*a^2 + 39*a^3;
```

$$1561858$$

Si a esta variable le asignamos otro dato, el valor asignado previamente se pierde:

```
> a := x^2 + alpha + 3;
```

$$a := x^2 + \alpha + 3$$

*Coordinación de Cómputo, Facultad de Ciencias, UNAM

```
> 3*a + 25*a^2 + 39*a^3;
```

$$3\,x^2 + 3\,\alpha + 9 + 25\,(x^2 + \alpha + 3)^2 + 39\,(x^2 + \alpha + 3)^3$$

## 1.2. Asignación de valores a una variable

A una variable de Maple se le puede asignar practicamente cualquier dato válido. Por ejemplo, se pueden asignar números, expresiones aritméticas, relaciones, cadenas, conjuntos, listas, etc. Veamos algunos ejemplos:

```
> var1 := 235^3 + sqrt(29);
```

$$var1 := 12977875 + \sqrt{29}$$

```
> com1 := 23^(1/3) + 8*I;
```

$$com1 := 23^{(1/3)} + 8\,I$$

```
> inq1 := 4*x + 5*x^2 < 10;
```

$$inq1 := 4\,x + 5\,x^2 < 10$$

```
> sec1 := dato1, a, 13, 4*25 + 8;
```

$$\mathrm{sec1} := dato1,\ x^2 + \alpha + 3,\ 13,\ 108$$

```
> lis1 := [x^2 + y^2, 8!, 34*29^2, 5.234];
```

$$lis1 := [x^2 + y^2,\ 40320,\ 28594,\ 5{,}234]$$

```
> cnj1 := {sqrt(23), `hola a todos`, [a, b, c]};
```

$$cnj1 := \{hola\ a\ todos,\ \sqrt{23},\ [x^2 + \alpha + 3,\ b,\ c]\}$$

Estas variables pueden ser operadas de la misma forma que el valor que contienen. Por ejemplo:

```
> com1 + 5*com1;
```

$$6\,23^{(1/3)} + 48\,I$$

```
> evalf[30](var1);
```

$$0{,}129778803851648071345040312507\,10^8$$

También se pueden definir variables a partir de otras variables previamente definidas:

```
> c1 := 92; exp1 := r^3 + p^4 - 35;
```

$$c1 := 92$$

$$\mathrm{exp1} := r^3 + p^4 - 35$$

```
> comb := c1 + c1*exp1 + c1^2*exp1^2;
```

$$comb := -3128 + 92\,r^3 + 92\,p^4 + 8464\,(r^3 + p^4 - 35)^2$$

Otro tipo de dato que puede ser asignado a una variable son las gráficas, como veremos a continuación.

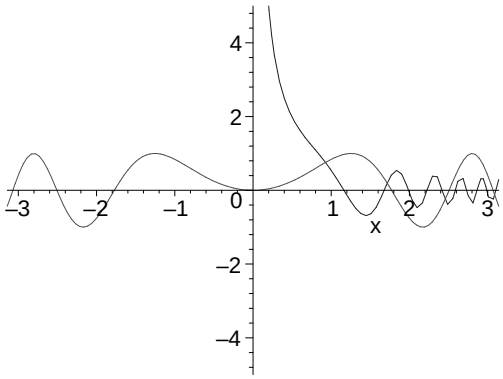
```
> grafica1 := plot(sin(x^2), x=-Pi..Pi, numpoints=5):

> grafica2 := plot(cos(x^3)/x, x=0.01..Pi, y=-5..5, color=blue):
```

Al hacer este tipo de asignación Maple muestra todos los datos creados, necesarios para desplegar la gráfica. En este caso se hace evidente la conveniencia de utilizar el caracter **":"** como terminador de instrucción, ya que de esta forma la asignación se lleva a cabo pero no se muestran todos los datos generados.

Este tipo de variables, al tener asignadas gráficas de funciones, podemos referenciarlas por medio de una instrucción que pueda manipular los datos de la gráfica. Una de las funciones de Maple que nos permiten

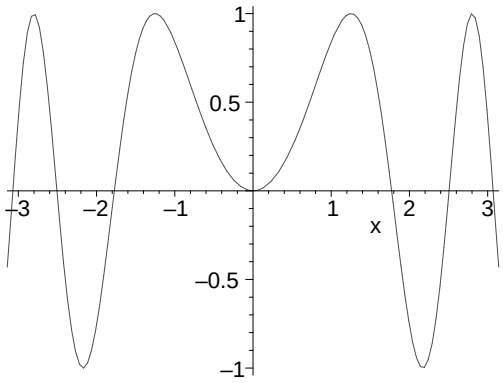
hacer esto es **display**, la cual puede tomar tales datos y desplegar la gráfica correspondiente. Esta función forma parte del paquete **plots** y puede usarse de la siguiente forma.

```
> plots[display](grafica1, grafica2);
```




Cuando se crean variables cuyo valor asignado es una gráfica, al evaluarlas en la linea de comandos obtenemos un despliegue de la gráfica. Por ejemplo:

```
> grafica1;
```




Los detalles acerca del uso de la función **display** serán tratados posteriormente en las secciones de gráficas en dos y tres dimensiones.

## 1.3. Nombres de variables

A continuación revisaremos algunas de las reglas que deben seguirse al hacer uso de variables en Maple.

### 1.3.1. Nombres válidos

En principio, toda expresión de Maple, sintácticamente bien formada, puede ser utilizada como nombre de variable, aunque se deben tener en cuenta algunas restricciones en las palabras utilizadas, como se verá a continuación.

Usualmente los caracteres que se usan para formar nombres de variables son:

- Letras (recuerdese que las mayúsculas son diferentes de las minúsculas).
- Números.
- Guiones de subrayado **"_"**, así como el signo **"?"**, siempre que no aparezca como primer caracter.
- Espacios en blanco, siempre y cuando el nombre sea tratado como una cadena. (Veanse los ejemplos más adelante).
- También es posible incluir subindices en las variables, aunque se debe tener cuidado con las expresiones usadas dentro de los corchetes.

Otros detalles que deben tenerse en cuenta al asignar un nombre de variable son los siguientes:

- El primer caracter debe ser una letra o un guión bajo, no puede ser un número o el signo **"?"**. Por ejemplo:

```
> variable1? := 9;
```

$$variable1? := 9$$

```
> _var2 :=%*23;
```

$$\_var2 := 207$$

```
> 12var1 := 13; # este no es un nombre válido
Error, missing operator or ‘;‘
```

---

**Nota:** aunque son validos, es recomendable evitar en lo posible el uso de nombres cuyo primer caracter sea un guión de subrayado, ya que Maple utiliza variables internas con esta forma.

---

- Los corchetes (**"[ ]"**), solo deben usarse para indicar subíndices, no deben colocarse como caracteres intermedios. Por ejemplo:

```
> num[1] := 23;
```

$$num_1 := 23$$

```
> num[2] :=%^2;
```

$$num_2 := 529$$

```
> var[23 := 24; # nombre no valido
Error, ‘:=‘ unexpected
```

---

**Nota:** las letras o palabras que se usan como subíndices, deben manejarse como cadenas. Esto nos permite evitar efectos colaterales provocados por el uso posterior de la misma expresión. Por ejemplo:

```
> solucion[x=1] := sqrt(29)*5!;
```

$$solucion_{x=1} := 120\sqrt{29}$$

En esta expresión es recomendable encerrar **“x=1"** entre acentos agudos, pues de lo contrario Maple puede considerarla como una asignación, lo cual puede provocar problemas al utilizar posteriormente **“x"**. La forma más recomendable es:

```
> solucion[‘x=1‘] := sqrt(29)*5!;
```

$$solucion_{x=1} := 120\sqrt{29}$$

De cualquier manera se obtiene el mismo resultado, pero en el segundo caso se evita cualquier problema por el uso posterior de **“x"**.

---

- Se pueden colocar espacios en blanco en los nombres de variables, siempre y cuando dicho nombre sea tratado como cadena (delimitado por acentos agudos). Por ejemplo:

```
> ‘solucion 1‘ := 97;
```

$$solucion\ 1 := 97$$

Al hacer referencia a esta variable también es necesario colocar los acentos.

```
> ‘solucion 1‘^2 + 2*‘solucion 1‘^3;
```

$$1834755$$

Cuando se manejan este tipo de nombres es más recomendable usar guiones de subrayado en lugar de espacios en blanco:

```
> solucion_1 := 23;
```

$$solucion\_1 := 23$$

```
> solucion_1^2 + 2*solucion_1^3 - 24;
```

$$24839$$

- Otro tipo de nombres válidos son aquellos en los cuales se incluyen paréntesis de la siguiente forma:

```
> dato1(x=0) := 25.35*4.2 + Pi;
```

$$\text{dato1}(x = 0) := 106{,}470 + \pi$$

Al invocar esta variable también deben usarse los paréntesis:

```
> 76*dato1(x=0);
```

$$8091{,}720 + 76\,\pi$$

Este tipo de variables son válidas, como se mencionó anteriormente; sin embargo, no es recomendable su uso ya que pueden causar confusión con las funciones del sistema y las definidas por el usuario.

### 1.3.2. Nombres no válidos

En la sección anterior se trató el caso de los tipos de expresiones válidas como nombres de variables en Maple. A continuación se muestran algunas de las expresiones que no deben usarse para este fin:

- No se pueden usar números, constantes o funciones predefinidas por Maple, como nombres de variables. Por ejemplo:

```
> 34 := 6!;

Error, invalid left hand side of assignment
```

```
> Pi := 24;

Error, attempting to assign to ‘Pi‘ which is protected

> cos := 34;

Error, attempting to assign to ‘cos‘ which is protected

> evalf := sin(x)*sqrt(x^2 - 25);

Error, attempting to assign to ‘evalf‘ which is protected
```

- Nombres que contengan símbolos de relación, operadores aritméticos, o bien parentesis o corchetes incompletos. Por ejemplo:

```
> dato<1 := -4.3;

Error, invalid left hand side of assignment

> dato>=5 := 7.9;

Error, invalid left hand side of assignment

> cuatro+5 := 9;

Error, invalid left hand side of assignment

> numero(1 := 34;

Error, ‘:=‘ unexpected

> cadena[2a:=‘hola a todos‘;

Error, missing operator or ‘;‘
```

## 1.4. Variables simbólicas

Generalmente se conceptualiza a una variable como un dato o conjunto de datos que pueden ser referenciados a través de un nombre; sin embargo, Maple también nos permite hacer uso de variables simbólicas, es decir, variables a las cuales no se les ha asignado ningun dato. Por ejemplo, consideremos la siguiente instrucción:

```
> a^2 + 23^3*(5! - 4) - 5*a^2;
```

$$-4\,(x^2+\alpha+3)^2+1411372$$

Nótese que, la expresión formada por la letra “$a$" en realidad no tiene asignado ningun valor que pueda ser evaluado, pero Maple de cualquier forma la opera. En este caso, dicha expresión es considerada una variable simbólica, es decir, una variable cuyo valor aun no ha sido definido.

Las variables símbolicas pueden estar dadas por cualquier nombre de variable válido.

## 1.5. Desasignación de una variable.

Cada vez que se define una variable ésta permanece hasta el fin de la sesión. Por ejemplo, definamos a continuación la variable ***“x"***.

```
> x := 34;
```

$$x := 34$$

¿Qué sucede si a continuación deseamos utilizar “$x$" como variable simbólica?

```
> a*x^2 + b*x + c = 0;
```

$$1339804 + 1156\,\alpha + 34\,b + c = 0$$

Obviamente no es posible, pues Maple sustituye la variable por el valor asignado. Para usarla de esta manera, deberiamos poder ***"eliminar"*** o ***"desasignar"*** su valor. Existen dos formas de hacerlo.

Una manera de eliminar una variable es a través de la instrucción **restart** (vease su página de ayuda), la cual también puede ser invocada desde la barra de herramientas por medio del botón . El problema es que esta instrucción elimina toda la información generada desde el inicio de la sesión hasta el momento que es ejecutada. Por lo cual, además de eliminar ***"x"***, eliminamos cualquier otro resultado presente en ese momento.

Una manera más conveniente en este caso, es utilizar lo que se conoce como ***"desasignación"*** de una variable. La sintaxis para esta ***"desasignación"*** es la siguiente:

**var:='var'**;

Es decir, a la variable le asignamos su mismo nombre pero encerrado entre apóstrofos. Por ejemplo, asignemos un nuevo valor a *"x"*.

```
> x := 345;
```

$$x := 345$$

A continuación desasignemos esta variable.

```
> x := 'x';
```

$$x := x$$

Ahora, introduzcamos la siguiente expresión.

```
> d*x^2 + f*x + h = 0;
```

$$d\,x^2 + f\,x + h = 0$$

Como puede verse, despues de ***"desasignarla"***, Maple elimina el valor asignado a la variable. Por lo cual, en la instrucción anterior, ***"x"*** es manejada como una variable simbólica.

Esta desasignación de variables también puede llevarse a cabo por medio de la instrucción **unassign**, de la siguiente forma:

**unassign('x');**

La ventaja de hacer uso de esta función es que nos permite ***"desasignar"*** más de una variable al mismo tiempo. Por ejemplo:

**unassign('x', 'y', 'grafica', 'solucion' );**

Esta instrucción desasigna todas las variables recibidas como argumento.