Uso de funciones en Maple

José Luis Torres Rodríguez*

Enero 2011

Un elemento importantisimo en Maple son las funciones. El sistema por sí mismo ya contiene un conjunto predefinido de éstas, pero también le proporciona al usuario varios mecanismos para poder definirlas suyas propias.

Al igual que con las variables, las funciones permanecen en la memoria desde su definición y pueden ser invocadas a lo largo de toda la sesión.

1. Funciones definidas por el usuario

La manera de crear funciones dadas por el usuario es por medio de la definición de operadores. Esto se hace utilizando la siguiente sintaxis:

nom := var -> regla;

Donde **nom** es el nombre que se le asignará a la función, **var** es la variable de la cual dependerá, y ***regla*** es la regla de correspondencia que se aplicará al invocar dicha función. Entre el nombre de la variable y la regla de correspondencia se debe colocar el operador **"->"**, formado por un signo menos (**"-"**) y un mayor que (**">"**). Por ejemplo, definamos la función **f**:

```
> f := x -> x + x^2;
```

$$f := x \to x + x^2$$

Este tipo de funciones pueden ser evaluadas en la forma: **f(n)**, donde **n** es una expresión aritmética, simbólica o una función.

Antes de continuar, definamos la función **g**:

```
> g := x -> 2*x + 3*x^2;
```

$$g := x \to 2\,x + 3\,x^2$$

Ahora, para evaluar **f** en $x = 25$, $x = \sqrt{23} + 18^2$ y $x = g(34)$, procedemos de la siguiente manera:

```
> f(25);
```

$$650$$

```
> f(sqrt(23) + 18^2);
```

$$\sqrt{23} + 324 + (\sqrt{23} + 324)^2$$

```
> f(g(34));
```

$$12506832$$

El procedimiento que debe seguirse para funciones de varias variables es análogo. Para definir la función procedemos de la siguiente forma:

*Coordinación de Cómputo, Facultad de Ciencias, UNAM

nom := (var1, var2, var3, ...) -> regla;

Nuevamente, **nom** es el nombre de la función, **"(var1, var2, var3, ...)"** son las variables de las cuales depende (deben colocarse entre parentesis) y **regla** es la regla de correspondencia asignada. Por ejemplo, definamos la siguiente función:

```
> h := (x, y, z) -> x^2 + y^2 + z^2;
```

$$h := (x, y, z) \to x^2 + y^2 + z^2$$

Para poder evaluar ***h*** en $x = 3$, $y = 9$, $z = g(x + y)$, procedemos de la siguiente forma:

```
> h(3, 9, g(3 + 9));
```

$$208026$$

La función se evaluará tomando el primer dato como el valor de la primer variable definida, el segundo dato como el valor de la segunda variable y así sucesivamente. De esta manera es posible definir funciones que dependan de una o varias variables. Este tipo de funciones también pueden ser utilizadas en instrucciones que reciben funciones como argumento, solamente se debe indicar a Maple que se trata de una función operador y se deben escribir explicitamente las variables de las cuales depende. Por ejemplo, la siguiente instrucción nos dá un valor erroneo:

```
> diff(f, x);
```

$$0$$

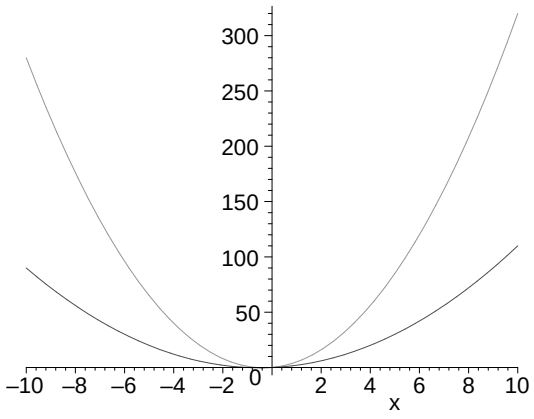
Comparese con la siguiente :

```
> diff(f(x), x);
```

$$1 + 2\,x$$

A continuación graficaremos **f** y **g** de la siguiente forma:

```
> plot({f(x), g(x)}, x=-10..10);
```



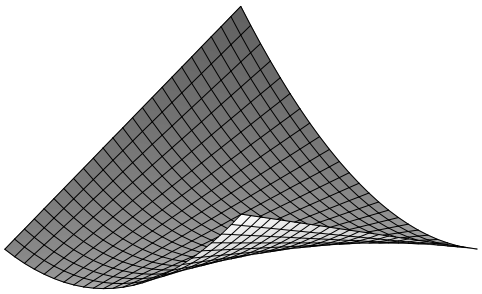
En este caso, al colocar **f** como argumento de **plot** debe expresarse como **f(x)**, de lo contrario Maple no podra hacer la evaluación correctamente. Lo mismo sucede con funciones de varias variables. Por ejemplo, si tenemos una función **d** que depende de ***a***, **b** y ***c,*** ésta debe ser siempre invocada como **d(a, b, c)**. Veamos el siguiente ejemplo. Primero definimos una función de dos variables:

```
> p := (e, h) -> e*h + h^2;
```

$$p := (e, h) \to e\,h + h^2$$

A continuación graficamos **p** para $-5 \leq e \leq 5$, $-4 \leq h \leq 4$

```
> plot3d(p(e, h), e=-5..5, h=-4..4);
```



Existe una forma de evitar el tener que escribir explicitamente dichas variables mediante el uso de ***"alias"***(esto se tratará más adelante). Por último, cabe señalar que Maple no nos permite hacer una definición recursiva de una función. Por ejemplo, no es válido hacer lo siguiente:

```
> F := x -> sin(x);
```

$$F := \sin$$

```
> F := x -> F(x) + sqrt(25);
```

$$F := x \rightarrow \mathrm{F}(x) + \sqrt{25}$$

Maple nos enviara un mensaje de error al tratar de evaluar esta función :

```
> F(5);
Error, (in F) too many levels of recursion
```

2. Composición de funciones

El definir funciones como operadores es bastante útil para poder hacer composición de funciones. Existen varias formas de hacer dicha composición. Una de ellas es simplemente anidar las funciones. Veamos un ejemplo.

Primero definimos las funciones **H** y **K**:

```
> H := a -> sin(a + 1);
```

$$H := a \rightarrow \sin(a + 1)$$

```
> K := a -> exp(1 - a);
```

$$K := a \rightarrow e^{(1-a)}$$

Ahora, calcularemos las composiciones **H(K(w))**, **K(H(w))** y **H(H(H(w)))**.

```
> H(K(w));
```

$$\sin(e^{(1-w)} + 1)$$

```
> K(H(w));
```

$$e^{(1-\sin(w+1))}$$

```
> H(H(H(w)));
```

$$\sin(\sin(\sin(w+1)+1)+1)$$

Otra forma de hacer este último cálculo es utilizando el operador de composición de funciones, de la siguiente forma:

(f1 @ f2 @ f3 @ ... @ fn)(x);

Donde **f1**, **f2**, **f3**, ... , **fn** son las funciones que se desea componer y ***x*** es el valor en el cual sera evaluada la composición. Esta expresión nos produce el mismo resultado que si emplearamos lo siguiente:

f1(f2(f3(...(fn(x)))));

Sin embargo, al usar la primera forma nos evitamos el tener que anidar todas las funciones con paréntesis. Utilizando esta notación, calculemos nuevamente las composiciones anteriores:

```
> (H@K)(w);
```

$$\sin(e^{(1-w)}+1)$$

```
> (K@H)(w);
```

$$e^{(1-\sin(w+1))}$$

```
> (H@H@H)(w);
```

$$\sin(\sin(\sin(w+1)+1)+1)$$

En este último caso, estamos haciendo una composición de la misma función tres veces. Este tipo de operaciones también las podemos realizar utilizando el operador de composición repetida de funciones, cuya notación es:

(f @@ n)(x);

Donde **f** es el nombre de la función, **n** es el número de veces que se desea componer consigo misma y ***x*** el valor en el cual se evaluará dicha composición. Por ejemplo, para hacer la composición de **H** consigo misma tres veces:

```
> (H@@3)(w);
```

$$\sin(\sin(\sin(w+1)+1)+1)$$

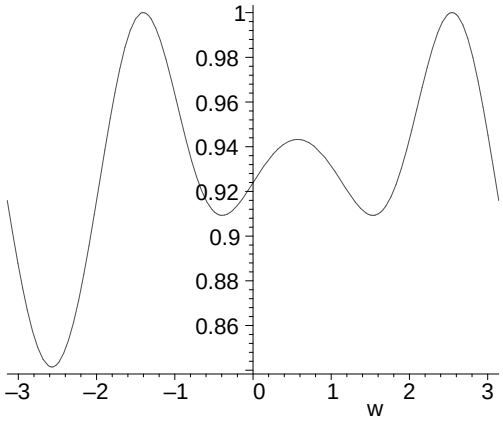
Este último resultado es el mismo que obtenemos al usar **(H@H@H)(w)** o bien **H(H(H(w)))**, solo que se trata de una forma más breve y sencilla. Utilizando esta notación, definiremos la siguiente función:

```
> Q := w -> (H@@3)(w);
```

$$Q := H^{(3)}$$

A continuación graficamos esta función para $-\pi \leq w \leq \pi$.

```
> plot(q, w=-Pi..Pi);
```



De hecho, todas estas formas de composición pueden ser utilizadas con las mismas funciones que proporciona Maple al usuario. Por ejemplo, calculemos el valor numérico de: $\sqrt{\cos(\cos(\cos(\pi^2)))}$.

Una forma de hacerlo es la siguiente:

```
> evalf(sqrt(cos(cos(cos(Pi^2)))));
```

$$0{,}9023117120$$

Otra manera es:

```
> (evalf@sqrt@cos@cos@cos)(Pi^2);
```

$$0{,}9023117120$$

Y existe una opción más:

```
> (evalf@sqrt@cos@@3)(Pi^2);
```

$$0{,}9023117120$$

Notese que en esta última instrucción utilizamos, además del operador de composición, el operador de composicición repetida.

En general, Maple nos permite hacer cualquier composición de funciones, siempre y cuando los datos de salida y entrada de éstas sean compatibles. Por ejemplo, no podemos hacer una composición con las funciones **plot** y **evalf**, pues los datos que manejan son diferentes. Téngase esto siempre en cuenta al hacer composiciones.

3. Transformación de expresiones en funciones operador

Considerese la siguiente asignación:

```
> g := x + sin(x);
```

$$g := x + \sin(x)$$

¿Cómo podemos, usando ***g***, obtener una función operador con la misma regla de correspondencia y de tal manera que también dependa de la variable ***x***?

Una posible opción es hacer lo siguiente:
Asignamos el argumento **x** a la expresión **g** (la regla de correspondencia) de la siguiente forma:

```
> G := x -> g;
```

$$G := x \to g$$

Intentemos evaluar para $\boldsymbol{x = 5}$:

```
> G(5);
```

$$x + \sin(x)$$

Esto nos genera un resultado erroneo, por lo cual se ve inmediatamente que este no es un método apropiado.

Maple nos proporciona un mecanismo a través del cual podemos transformar cualquier expresión en una función operador con la misma regla de correspondencia y que dependa de la misma variable (o variables, según sea el caso), por medio de la instrucción **unapply** (consúltese su página de ayuda). La notación es la siguiente:

unapply(E, x1, x2, x3,...xn);

Donde **E** es la expresión que se desea convertir y **x1, x2, x3, ... ,xn** son las variables de las cuales dependerá la función operador generada (deben ser las mismas variables que aparecen en la expresión **E**). Esta instrucción devuelve una función operador con las características antes descritas. Por ejemplo, podemos convertir $\boldsymbol{g}$ en operador de la siguiente forma:

```
> G := unapply(g, x);
```

$$G := x \to x + \sin(x)$$

Desde este momento, **G** es considerada una función operador con regla de correspondencia equivalente a la de **g**. Probemos evaluando en $\mathbf{x = Pi/2}$:

```
> G(Pi/2);
```

$$\frac{\pi}{2} + 1$$

Incluso, podemos hacer composiciones usando esta función:

```
> (evalf@G@@4)(Pi^3);
```

$$28{,}31175398$$

El caso de funciones de dos o más variables es equivalente. Veamos el siguiente ejemplo:

```
> d := 4*x + 5*y + x*y;
```

$$d := 4\,x + 5\,y + x\,y$$

Convertimos **d** a operador de la siguiente forma:

```
> funcion := unapply(d, x, y);
```

$$funcion := (x,\, y) \to 4\,x + 5\,y + x\,y$$

y a continuación la evaluamos para datos particulares:

```
> funcion(6, sqrt(24));
```

$$24 + 22\sqrt{6}$$

También podemos usar esta función para hacer composición con otras funciones:

```
> (evalf@funcion)(6, sqrt(24));
```

$$77{,}88877435$$

Otra de las ventajas de estas conversiones es que Maple nos permite definir operadores a partir de cualquier expresión, en particular a partir de resultados generados por otras instrucciones. Por ejemplo, calculemos la siguiente integral con respecto a $\boldsymbol{x}$:

```
> int(1/(a^3 + x^3), x);
```

$$-\frac{1}{6}\,\frac{\ln(a^2 - a\,x + x^2)}{a^2} + \frac{1}{3}\,\frac{\sqrt{3}\arctan(\frac{(-a + 2\,x)\,\sqrt{3}}{3\,a})}{a^2} + \frac{1}{3}\,\frac{\ln(a + x)}{a^2}$$

Este resultado lo asignaremos a la variable **res** pero en forma de operador y evaluaremos en: **a** = **.32**, **x** = **2.1**

```
> res := unapply(%, a, x);
```

$$res := (a,\, x) \rightarrow -\frac{1}{6}\,\frac{\ln(a^2 - a\,x + x^2)}{a^2} + \frac{1}{3}\,\frac{\sqrt{3}\arctan(\frac{1}{3}\,\frac{(-a + 2\,x)\,\sqrt{3}}{a})}{a^2} + \frac{1}{3}\,\frac{\ln(a + x)}{a^2}$$

```
> res(.32, 2.1);
```

$$0{,}686781499 + 3{,}255208333\,\sqrt{3}\arctan(4{,}041666667\,\sqrt{3})$$

Esta es una forma en la cual podemos retener expresiones generadas como resultado de alguna operación, para poder evaluarlas o manipularlas posteriormente.

4. Funciones predefinidas

Existen varias funciones de Maple que se encuentran disponibles automáticamente cada vez que iniciamos una sesión en este sistema. Todas ellas son conocidas como ***"funciones predefinidas"*** y no requieren de ninguna definición por parte del usuario para poder ser utilizadas, simplemente deben ser invocadas con sus respectivos argumentos. Entre ellas se encuentran las siguientes:

Funciones trigonométricas, trigonométricas inversas e hiperbólicas

A continuación se muestran las funciones trigonométricas soportadas por Maple, así como sus respectivas inversas, también se proporciona una liga a su página de ayuda.

A continuación, la tabla 1 muestra las funciones trigonométricas soportadas por Maple, así como sus respectivas inversas.

Nombre	**Función**	**Inversa**
Seno	sin(x)	arcsin(x)
Coseno	cos(x)	arccos(x)
Tangente	tan(x)	arctan(x)
Cotangente	cot(x)	arccot(x)
Secante	sec(x)	arcsec(x)
Cosecante	csc(x)	arccsc(x)

Cuadro 1: Funciones trigonométricas

La tabla 2 muestra sus respectivas funciones hiperbólicas y las inversas de éstas:

Nombre	Función	Inversa
Seno hiperbólico	sinh(x)	arcsinh(x)
Coseno hiperbólico	cosh(x)	arccosh(x)
Tangente hiperbólica	tanh(x)	arctanh(x)
Cotangente hiperbólica	coth(x)	arccoth(x)
Secante hiperbólica	sech(x)	arcsech(x)
Cosecante hiperbólica	csch(x)	arccsch(x)

Cuadro 2: Funciones trigonométricas hiperbólicas

Nota: la hoja de ayuda de estas funciones puede consultarse en la forma: **?función**

Todas estas funciones son calculadas en radianes. Algunas de ellas, particularmente el seno y el coseno, ya habian sido usadas con anterioridad. La forma de utilizar las demás es análoga. Por ejemplo:

```
> sin(Pi/8);
```

$$\sin(\frac{\pi}{8})$$

```
> coth(3.1 + 2.5*I);
```

$$1{,}001144421 + 0{,}003896610898\, I$$

```
> diff(arctanh(x), x);
```

$$\frac{1}{1-x^2}$$

```
> (sin@cos@arctan)(.2);
```

$$0{,}8308206799$$

Estas funciones también pueden ser utilizadas para hacer composiciones; incluso es valido usar expresiones como la siguiente para calcular inversas:

```
> sin@@(-1);
```

$$\arcsin$$

Por ejemplo, para evaluar **arctan** en 0.5, podemos utilizar :

```
> (tan@@(-1))(.5);
```

$$0{,}4636476090$$

Exponencial y logaritmo

La función exponencial está disponible en Maple como: **exp(x)**. Veamos algunos ejemplos de su uso:

```
> exp(5);
```

$$e^5$$

```
> evalf(%);
```

$$148{,}4131591$$

```
> evalf(cos(exp(44)));
```

$$-0{,}1618529225$$

Acerca del *logaritmo*, Maple proporciona tres funciones para calcularlo:

- $ln(x)$. Calcula el logaritmo natural de x.
- $log[b](x)$. Calcula el logaritmo en base "b", de "x".
- $log10(x)$. Calcula el logaritmo en base 10.

Estas funciones son evaluadas por Maple de la siguiente forma:

- El logaritmo natural, **ln**, es el logaritmo con base $exp(1) = 2{,}71828...$
  Para $x > 0$ tenemos que: $ln(x) = y \Longleftrightarrow x = exp(y)$.
- Dada una expresión compleja **x**, su logaritmo natural está dado por: $ln(x) = ln(abs(x)) + I * argument(x)$, donde $-\pi < argument(x) \leq \pi$. La función **abs** calcula el valor absoluto (módulo), de **x**, mientras que **argument** calcula el argumento. En Maple, este cálculo es considerado como la definición de la rama principal del logaritmo.
- La función **log** calcula el logaritmo en general, para cualquier base dada. Para $x > 0$ y $b > 0$, tenemos que $log[b](x) = y \Longleftrightarrow x = b^y$. Esta función es extendida para complejos b y x en general, por medio de la fórmula: $log[b](x) = ln(x)/ln(b)$.
- En el caso de la función **log** , el valor por default para la base **b** es **exp(1)**.
- La función **log10(x)**, es equivalente a **log[10](x)**.

Recuérdese que la función logaritmo solo acepta argumentos mayores que cero.

```
> exp(ln(9));
```

$$9$$

```
> ln(exp(-14));
```

$$-14$$

Veamos ahora algunos ejemplos con complejos:

```
> ln(4 + 9.5*I);
```

$$2{,}332897404 + 1{,}172273881\, I$$

```
> ln(3 + 4*I);
```

$$\ln(3 + 4\, I)$$

En este úlimo ejemplo, Maple no hace la evaluación ya que solo estamos incluyendo números enteros como argumento de **ln**. En este caso es necesario aplicar la función **evalc** para evaluación numérica de complejos y despues **evalf**:

```
> evalc(ln(3 + 4*I));
```

$$\ln(5) + \arctan(\frac{4}{3})\, I$$

```
> evalf(%);
```

$$1{,}609437912 + 0{,}9272952179\, I$$

Veamos otro ejemplo:

```
> log10(10000);
```

$$\frac{\ln(10000)}{\ln(10)}$$

Esta última instrucción nos dá el logaritmo en base 10 de 10000; es equivalente a: **evalf(log[10](10000))**.

Otras funciones básicas

La tabla 3 muestra otras funciones comunmente usadas, disponibles en Maple:

Nombre	**Función**
Raiz cuadrada	sqrt(x)
Valor absoluto	abs(x)
Límite de f en x	limit(f, x)
Derivada de f respecto a x	diff(f, x)
Factorial	fact(x) ó !x
Máximo entero menor o igual	floor(x)
Mínimo entero mayor o igual	ceil(x)
Redondeo	round(x)
Truncamiento	trunc(x)
Máximo y mínimo de una secuencia	max(sec), min(sec)
Parte fraccionaria	frac(x)
Módulo	a mod b
Conjugado de un complejo	conjugate(c)
Argumento de un complejo	argument(c)
Parte real e imaginaria de un complejo	Re(c), Im(c)
Forma polar de un complejo	polar(c)

Cuadro 3: Otras funciones disponibles en Maple

Una lista completa de las funciones matemáticas más comúnmente utilizadas, puede ser consultada en la página de ayuda de **inifcn**.

Existen otras funciones predefinidas disponibles en los diversos paquetes de Maple. Por ejemplo, el paquete **LinearAlgebra** nos proporciona varias de ellas, útiles para operaciones de Álgebra Lineal; mientras que el paquete **stats** nos proporciona una serie de funciones estadisticas. Una lista completa de los paquetes disponibles puede consultarse en la hoja de ayuda **index[package]**. Más adelante se tratarán algunas de estas funciones predefinidas.

5. Funciones sin regla de correspondencia

En Maple es posible definir una funcion sin dar su regla de correspondencia (de la misma forma que podemos usar variables simbólicas). Por ejemplo:

```
> B := x -> b(x);
```

$$B := b$$

Nota: Si la regla de correspondencia no ha sido especificada (como es el caso), no podemos usar el mismo nombre para la función y para la regla de correspondencia (por ejemplo $B := x->B(x)$).

Para evaluar la función $\boldsymbol{B}$ en $\boldsymbol{x = 3}$, seguimos la regla de evaluación de cualquier función operador:

```
> B(3);
```

$$\mathrm{b}(3)$$

Podemos combinar funciones de este tipo con funciones predefinidas y con aquellas definidas por el usuario, cuya regla de correspondencia haya sido especificada. Por ejemplo:

```
> F := x -> x^2;
```

$$F := x \to x^2$$

```
> C := x -> B(x) + sin(x) + sqrt(x) - F(x);
```

$$C := x \to \mathrm{B}(x) + \sin(x) + \sqrt{x} - \mathrm{F}(x)$$

```
> C(2);
```

$$\mathrm{b}(2) + \sin(2) + \sqrt{2} - 4$$

6. Forma inerte de una función

Entre las funciones predefinidas existen algunas que soportan dos modos de operación. Uno de estos modos es el que hemos estado usando hasta aquí, en el cual pasamos un argumento a la función y ésta nos devuelve el resultado de aplicar la regla de correspondencia que tiene asignada. En el otro modo, al pasar un argumento, la función nos devuelve una expresión matemática que representa la operación solicitada. Esta última es conocida como la *"**forma inerte**"* de la función. Por ejemplo, una de las funciones que soporta estos dos modos de operación es **limit**, que nos calcula el límite de una función alrededor de un punto. La forma de invocar esta función para calcular un límite es:

limit(función, punto);
Limit(función, punto);

La segunda instrucción corresponde a la forma inerte.

En general, en las funciones que soportan una forma inerte, ésta puede ser invocada colocando la primer letra del nombre en mayúscula. Para visualizar la diferencia entre estos dos modos conaiderense las siguientes instrucciones:

```
> limit(tan(x), x=Pi/4);
```

$$1$$

```
> Limit(tan(x), x=Pi/4);
```

$$\lim_{x \to (\frac{\pi}{4})} \tan(x)$$

Nótese el resultado de la segunda instrucción, ésta nos devuelve no el límite, sino una expresión no evaluada, en simbología matemática, que representa dicha operación. Tales expresiones pueden ser evaluadas, como se vio anteriormente, utilizando le instrucción **value**.

```
> value(%);
```

$$1$$

Algunas otras funciones que soportan una forma inerte son:

- $diff(f(x), x)$ o $Diff(f(x), x)$. Calcula la diferencial de f con respecto a x.
- $int(f(x), x)$ o $Int(f(x), x)$. Calcula la integral de f respecto a x.
- $sum(expr(x), x = a..b)$ o $Sum(expr(x), x = a..b)$. Calcula la sumatoria para la expresión $expr(x)$, haciendo variar x desde a hasta b, es decir: $expr(a) + expr(a + 1) + expr(a + 2) + ... + expr(b)$.
- $product(expr(x), x = a..b)$ o $Product(expr(x), x = a..b)$. Calcula el producto de la expresión dada, haciendo variar x desde a hasta b, es decir: $expr(a) * expr(a + 1) * expr(a + 2) * ... * expr(b)$.

Por ejemplo:

```
> diff(sin(x) + cos(sqrt(x4)), x);
```

$$\cos(x)$$

```
> Diff(sin(x) + cos(sqrt(x4)), x);
```

$$\frac{\partial}{\partial x}\left(\sin(x) + \cos(\sqrt{x4})\right)$$

```
> Diff(sin(x) + cos(sqrt(x4)), x) = diff(sin(x) + cos(sqrt(x4)), x);
```

$$\frac{\partial}{\partial x}\left(\sin(x) + \cos(\sqrt{x4})\right) = \cos(x)$$

```
> Int(x^2 + y^3 + x^4, x) = int(x^2 + y^3 + x^4, x);
```

$$\int x^2 + y^3 + x^4\, dx = \frac{1}{3}\, x^3 + y^3\, x + \frac{1}{5}\, x^5$$

```
> Sum(a*x^2 + x/(x^4 + 1), x=-10..10) = sum(a*x^2 + x/(x^4 + 1),
> x=-10..10);
```

$$\sum_{x=-10}^{10} \left(a\, x^2 + \frac{x}{x^4 + 1}\right) = 770\, a$$

```
> Product(x^(1/x) + x/4, x=2..7) = product(x^(1/x) + x/4, x=2..7);
```

$$\prod_{x=2}^{7} \left(x^{\left(\frac{1}{x}\right)} + \frac{x}{4}\right) = \left(\sqrt{2} + \frac{1}{2}\right)\left(3^{(1/3)} + \frac{3}{4}\right)\left(4^{(1/4)} + 1\right)\left(5^{(1/5)} + \frac{5}{4}\right)\left(6^{(1/6)} + \frac{3}{2}\right)\left(7^{(1/7)} + \frac{7}{4}\right)$$

```
> Product(x^(1/x) + x/4, x=2..7) = evalf(product(x^(1/x) + x/4, x=2..7));
```

$$\prod_{x=2}^{7} \left(x^{\left(\frac{1}{x}\right)} + \frac{x}{4}\right) = 232{,}9768924$$