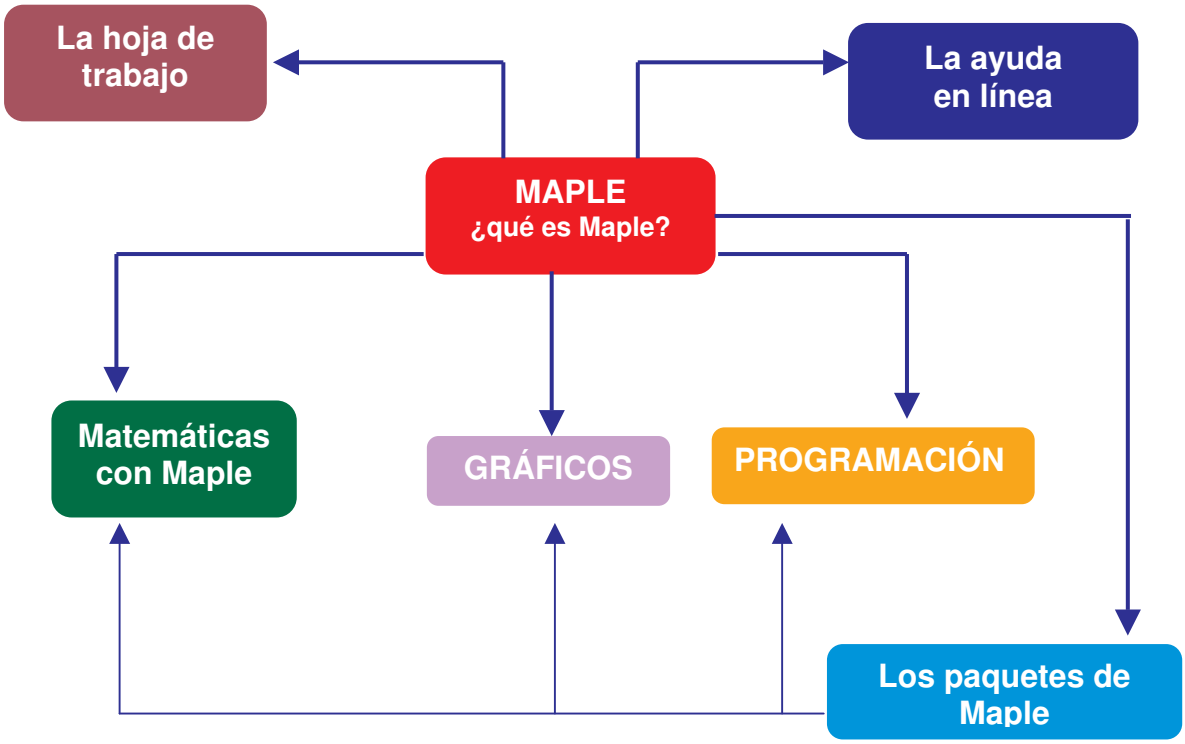
INTRODUCCIÓN A MAPLE

Autores: María Teresa Pérez Rodríguez (terper@wmatem.eis.uva.es), Oscar Arratia García (oscarr@wmatem.eis.uva.es).

MAPA CONCEPTUAL



INTRODUCCIÓN

En los últimos años los ordenadores han incrementado de forma drástica su capacidad para resolver grandes problemas procedentes de los más diversos campos de la Ciencia debido, de un lado al portentoso avance que ha sufrido el hardware (ordenadores más potentes y rápidos) y de otro al reciente desarrollo de software con un elevado nivel de sofisticación. Como parte de este software están los sistemas de Cálculo Científico que permiten llevar a cabo no sólo cálculos numéricos complicados sino manipulaciones analíticas y tratamientos gráficos de los problemas.

Son múltiples los sistemas de este tipo, mencionaremos algunos como DERIVE, REDUCE, MACSIMA, Mathematica, Maple. MuPAD o AXIOM, que están entre los de propósito general. Citamos también otros, más dirigidos al cálculo numérico, como Mathcad o Matlab que han incorporado el núcleo algebraico de Maple para manipulaciones analíticas.

Debido a la gran utilidad y aplicabilidad de estos programas es una ventaja el contar con conocimientos sobre el manejo de alguno o varios de ellos. Por esto, en este bloque pretendemos dar las nociones básicas que permitan comenzar a manejar el manipulador simbólico Maple y que

dejen al lector en situación de explorar por si mismo otras opciones diferentes de las que aquí se presentan.

Sería imposible una descripción detallada del sistema, por lo que nos restringimos a mostrar la amplia gama de posibilidades que ofrece realizando una pequeña introducción para aquellas que juzgamos más relevantes. La aplicación del manipulador a los distintos campos de las Matemáticas se deja para los bloques específicos en los que se presentan problemas resueltos con Maple y se describen en detalle los comandos relacionados.

El sistema Maple es esencialmente un sistema interactivo. Por ello es muy interesante que el lector tenga acceso al propio programa de modo que pueda experimentar inmediatamente todo lo que se comente en las secciones siguientes. Desde la dirección http://www.maplesoft.com/trial.shtml se puede descargar una copia gratuita con la que explorar las posibilidades del manipulador. En lo que sigue, la versión 8 de Maple será la base sobre la que se explique el comportamiento del sistema.

OBJETIVOS

- Entender lo que es el sistema Maple
- Adquirir las nociones básicas del trabajo con Maple.
- Manejar la ayuda y la interfaz del programa.
- Formarse una idea global de las múltiples capacidades de este manipulador.

CONOCIMIENTOS PREVIOS

Es recomendable estar familiarizado con entornos gráficos de ordenadores. También es necesario el conocimiento de las Matemáticas a nivel elemental y en particular es aconsejable conocer cómo se representan los números reales en coma flotante.

CONCEPTOS FUNDAMENTALES

¿Qué es Maple?

Maple es *un sistema de cálculo simbólico o algebraico*. Ambas expresiones hacen referencia a la habilidad que posee Maple para trabajar con la información de la misma manera que lo haríamos nosotros cuando llevamos a cabo cálculos matemáticos analíticos. Mientras que los programas matemáticos tradicionales requieren valores numéricos para todas las variables, Maple mantiene y manipula los símbolos y las expresiones. Estas capacidades simbólicas permiten obtener soluciones analíticas exactas de los problemas matemáticos: por ejemplo se pueden calcular límites, derivadas e integrales de funciones, resolver sistemas de ecuaciones de forma exacta, encontrar soluciones de ecuaciones diferenciales, etc. Como complemento a las operaciones simbólicas existe un amplio conjunto de rutinas gráficas que permiten visualizar información matemática compleja, algoritmos numéricos que dan soluciones en precisión arbitraria de problemas cuya solución exacta no es calculable y un lenguaje de programación completo y comprensible que permite al usuario crear sus propias funciones y aplicaciones.

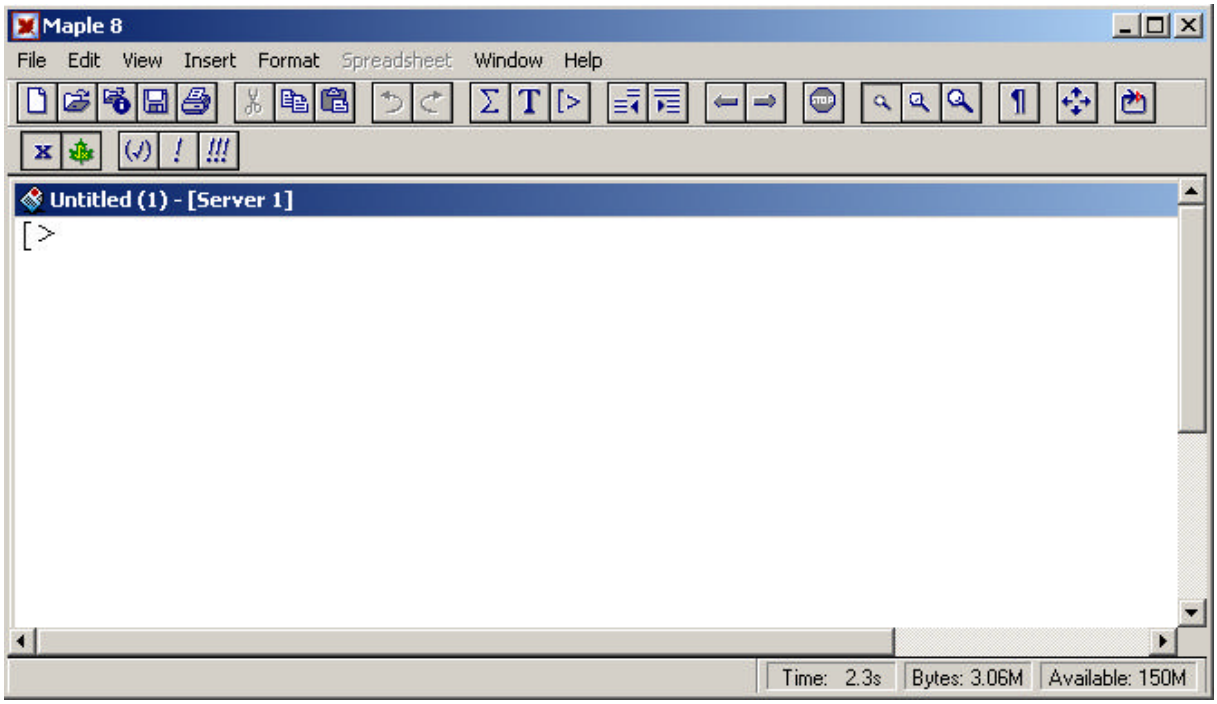
Internamente Maple se estructura en tres partes. En primer lugar está el *núcleo*, formado por rutinas escritas y compiladas en lenguaje C, donde se realizan la mayor parte de los cálculos básicos hechos por el sistema. La segunda parte es un conjunto de *librerías*, donde se encuentra la mayoría de los comandos de Maple, y que están escritas en su propio lenguaje de programación (interpretado no compilado), lenguaje que permite al usuario crear sus propios comandos y añadirlos a la librería estándar (es por tanto un sistema extensible). Y finalmente la interfaz del programa a través de la cual es posible comunicarse con el sistema.

Esta interfaz de Maple tiene un aspecto muy similar a la de otros programas usados en sistemas operativos con entorno gráfico y permite el acceso a todas las funciones y capacidades del manipulador. Básicamente lo que aparece al invocar el programa Maple (haciendo doble clic en su icono, por ejemplo) es una ventana más o menos convencional en la que se encuentra integrado lo que en inglés se denomina "worksheet" y que nosotros traduciremos como "hoja de trabajo". La flexibilidad de la hoja de trabajo permite tanto la investigación en ideas matemáticas como la creación de artículos técnicos sofisticados. De esta manera Maple presenta grandes posibilidades de aplicación y uso tanto en la investigación como en el trabajo profesional y por supuesto en la enseñanza de las Matemáticas.

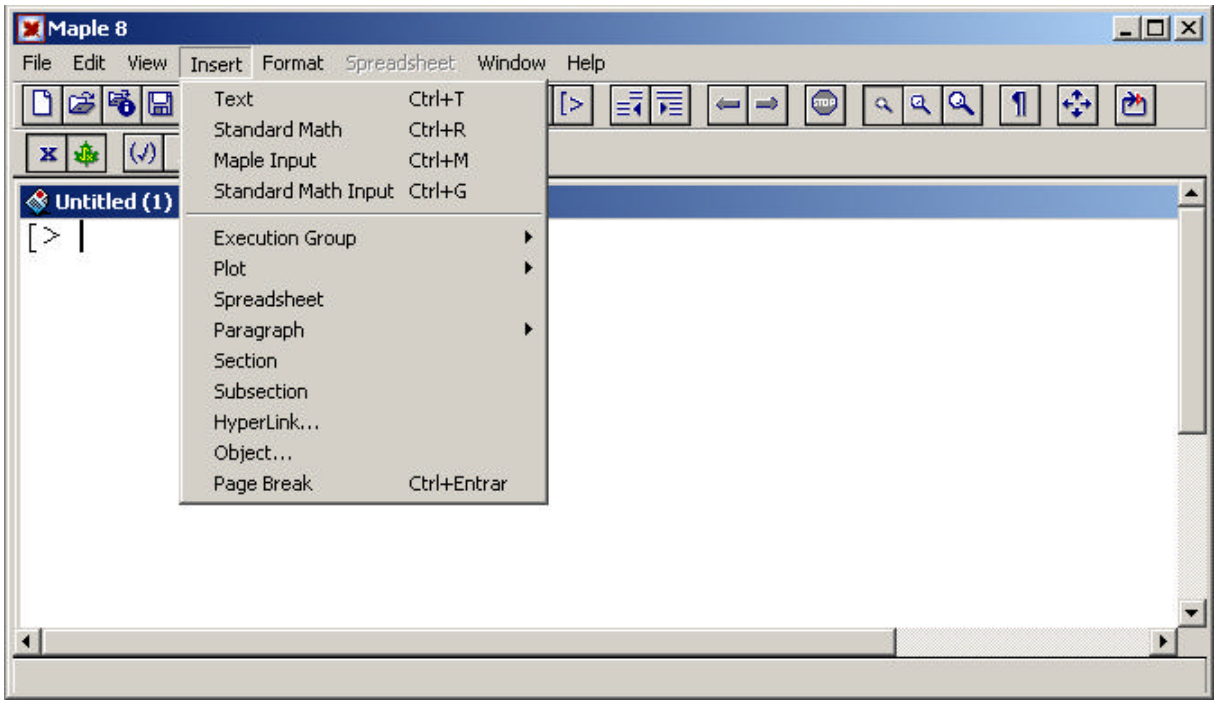
En la sección siguiente describimos con más detalle la interfaz y la hoja de trabajo de Maple.

La hoja de trabajo de Maple

La interfaz gráfica de Maple permite realizar todas las operaciones de edición que cabría esperar de cualquier software moderno. Así, una vez que se invoca el programa, aparece la ventana siguiente.



En su parte superior está la barra de Menú, con menús tales como **File** (Archivo) o **Edit** (Edición), muy parecidos a los de cualquier otra aplicación con entorno gráfico (en la figura siguiente vemos desplegado el menú **Insert** (Insertar)). Inmediatamente debajo tenemos la barra de herramientas, que contiene botones para tareas comunes de edición y otras específicas de Maple algunas de las cuales comentaremos más adelante. Finalmente, debajo de la barra de herramientas, aparece la llamada barra de contexto que contiene controles específicos de la tarea que se está realizando en cada momento.

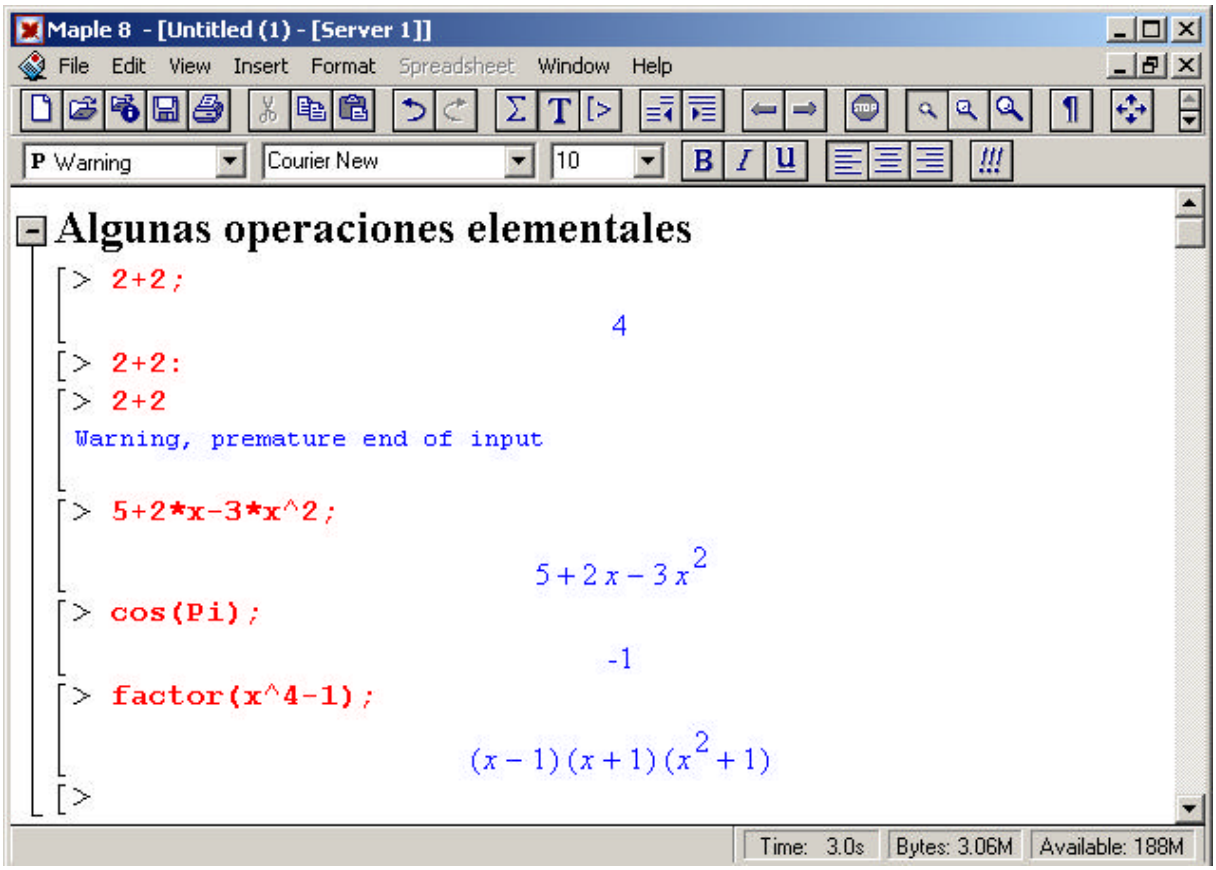


Debajo de estas tres barras hay un área en blanco en la que se desplegará la hoja de trabajo: es la región donde el usuario va a introducir comandos de Maple, texto, etc . Por último, en la parte inferior de la pantalla se encuentra la barra de estado.

La hoja de trabajo, componente especial de la interfaz de Maple, es un entorno integrado en el que, interactivamente, se resuelven problemas y se documenta el trabajo. Contiene no solamente texto sino también comandos matemáticos vivos que generan resultados automáticamente. La resolución de problemas interactivamente se reduce a ejecutar los comandos adecuados de Maple y recibir sus respuestas. En la hoja de trabajo, el cambio de la secuencia de comandos y su re-ejecución es muy sencilla. También permite controlar la forma en que se dan los comandos y sus salidas. Finalmente el contenido de la hoja de trabajo se puede guardar en un archivo con extensión mws o exportar en distintos formatos. Las opciones para llevar a cabo estas acciones se encuentran en el menú **File**.

Este es el icono que se asigna a un fichero generado con Maple a partir de una hoja de trabajo.

Como se observa en la imagen, en la parte superior de la hoja de trabajo en blanco aparece un símbolo con el siguiente aspecto [> . Este símbolo es el *prompt* de comandos e indica que lo que espera el editor es una instrucción del sistema Maple: cualquier cosa que se escriba a continuación aparecerá en rojo, color reservado a los comandos, mientras que el texto utiliza el color negro. Las instrucciones de Maple han de finalizar con **;** (característica esta común con el lenguaje de programación C) o con **:**. La diferencia entre ambas opciones es que la primera genera una salida en la pantalla (en azul) mientras que la segunda evita que ésta aparezca aunque, por supuesto, en ambos casos el comando se ejecuta cuando se pulsa la tecla de retorno de carro.



La hoja de trabajo de la imagen superior muestra una sección titulada **Algunas operaciones elementales** (creada con la opción **Section** del menú **Insert**). La sección se puede plegar pinchando en el cuadrado a la izquierda del título, desplegándose después de la misma forma. Los tres primeros comandos pretenden conseguir la suma de dos números enteros: 2+2. Observemos las salidas que generan: en el primer caso como el comando termina en ; aparece la suma 4 en azul, el segundo no tiene salida pues finaliza con : , en el tercer comando la salida es un aviso indicando que falta el símbolo de final de instrucción. En la cuarta línea simplemente se escribe un polinomio. Nótese hasta aquí que suma, resta, producto y exponenciación se designan por **+**, **-**,* y **^** ; para completar los operadores aritméticos diremos que la división se designa por / y que, alternativamente, la exponenciación también admite la representación **** . En la quinta línea tenemos la función coseno evaluada en π cuyo resultado, –1, aparece al ejecutar el comando. Finalmente el comando **factor** factoriza la expresión que aparece entre paréntesis.

Existe la posibilidad, como ya hemos indicado antes, de escribir texto en la hoja de trabajo. El texto aparece en negro y para cambiar de modo comando a modo texto y viceversa se pueden utilizar los botones  de la barra de herramientas. El segundo botón, con una T, cambia de modo comando a modo texto mientras que el tercero botón, con [>, hace aparecer un *prompt* en el momento que se pincha. Por último, el primer botón, con una Σ, permite introducir fórmulas matemáticas dentro de texto con un formato similar al que tienen en las salidas de los comandos. En el menú **Insert** se encuentran estas mismas acciones junto con otras posibilidades de edición que permiten estructurar la hoja de trabajo mediante secciones y subsecciones o crear hipervínculos a otra hoja de trabajo o a una página de ayuda.

Si el usuario está familiarizado con programas cuya interfaz esté desarrollada en un entorno gráfico no tendrá ningún problema en lograr un ágil manejo de la hoja de trabajo de Maple, puesto que la mayoría de las acciones de edición son estándar y aquellas especificas del manipulador son bastante intuitivas.

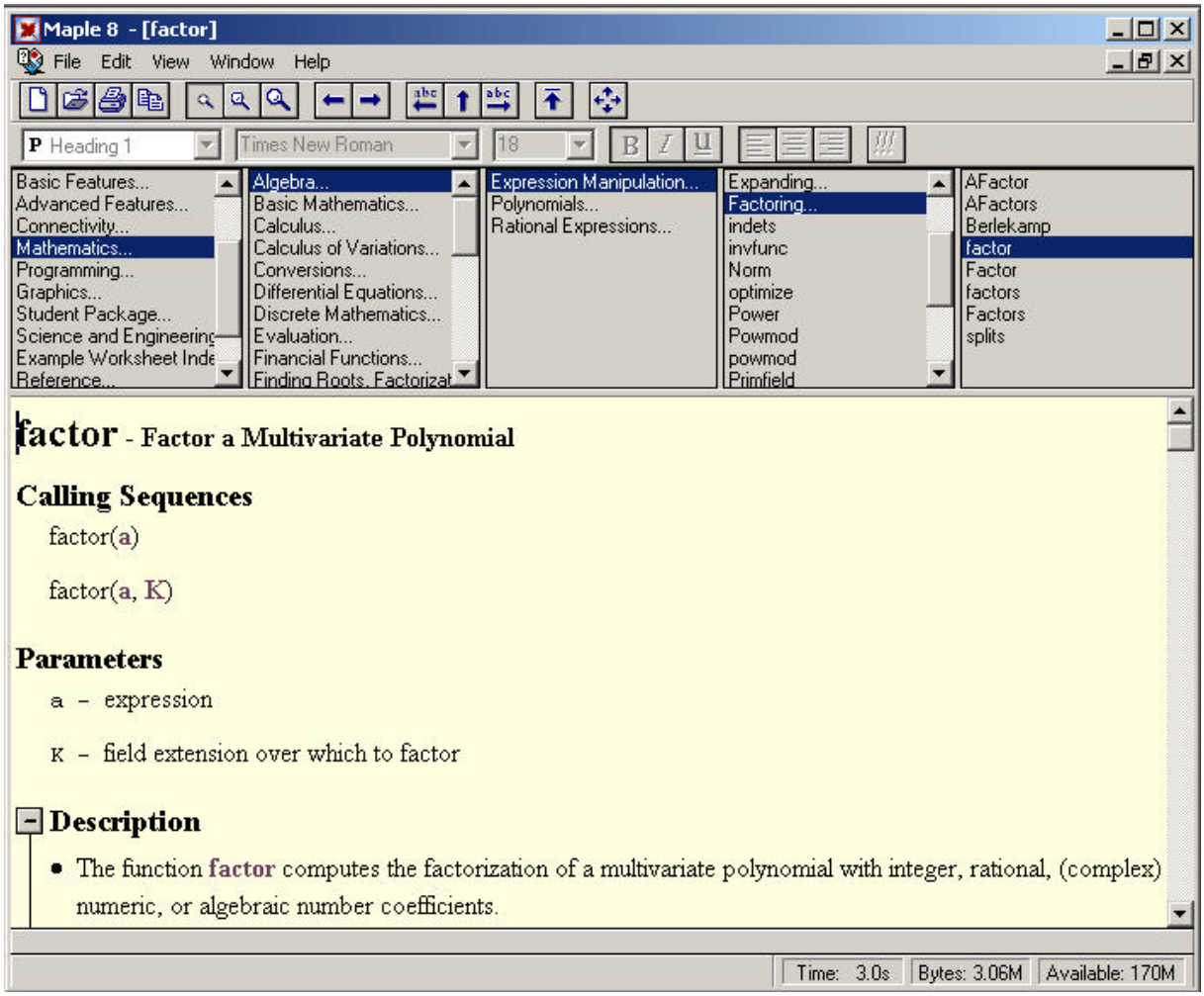
La ayuda de Maple

Maple posee un completo manual de referencia que se puede consultar "on-line". El sistema de ayuda permite explorar los comandos de Maple, así como las características del sistema, por nombre o materia. Además puede localizar páginas de ayuda que contengan una palabra o frase determinada. Las páginas de ayuda relacionadas están unidas mediante hipervínculos, lo que permite investigar cualquier tópico de forma sencilla. A continuación damos una breve introducción al uso de la ayuda de Maple.

Si se conoce el nombre de un comando determinado, es posible pedir ayuda sobre el mismo desde la línea de comandos utilizando el símbolo **?** seguido del nombre. Así la ejecución de la instrucción

```
[> ?factor
```

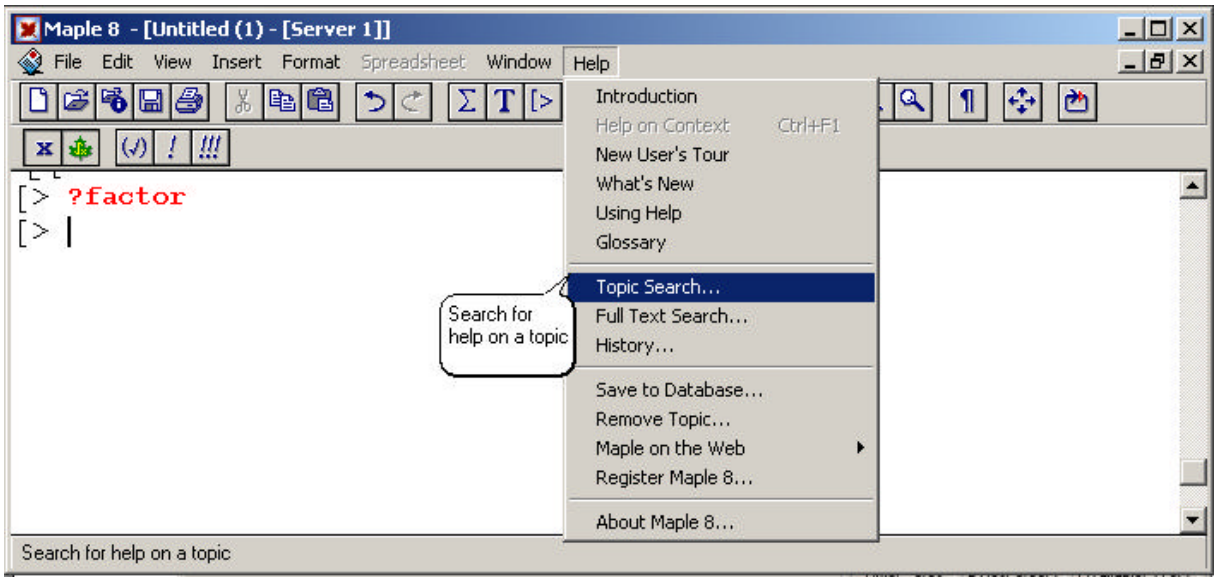
despliega la hoja de ayuda que mostramos en la figura



Como se observa en la imagen es posible navegar por el manual de ayuda de forma sencilla seleccionando el tópico que se desee en la zona que aparece debajo de la barra de estado en la que se selecciona pinchando, primero el capítulo (zona de más a la izquierda), después la sección del capítulo, luego las subsecciones y finalmente el comando deseado.

La consulta del manual de referencia se puede realizar también desde el menú desplegable de **Help** en el que aparecen varias opciones, algunas de las cuales comentamos a continuación.

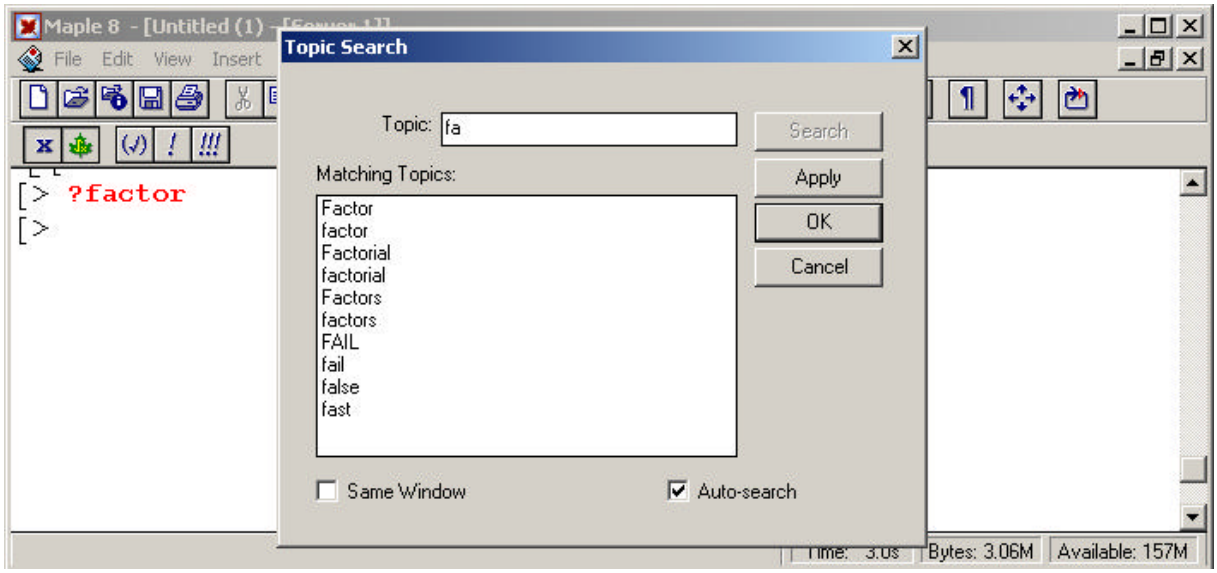
Si se desea ayuda sobre una palabra escrita en la hoja de trabajo, basta situar el cursor sobre dicha palabra y seleccionar en el menú **Help** la opción **Help on *word*** (donde ***word*** es la palabra sobre la que se encuentra el cursor).



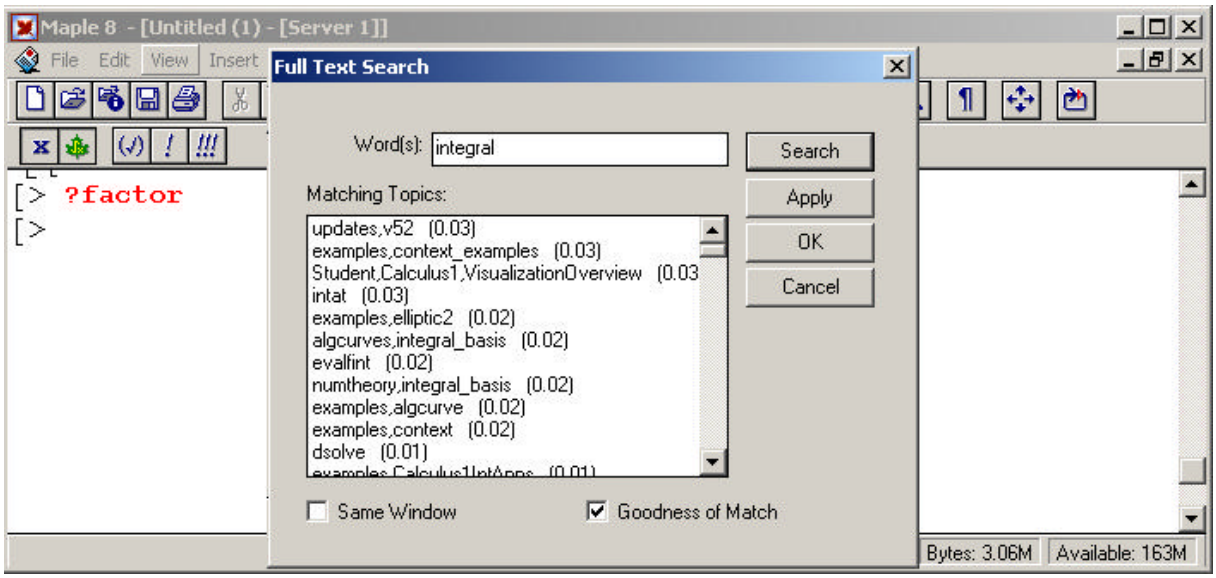
La opción **New User's Tour** accede a un conjunto de páginas de ayuda a través de las cuales se presentan los comandos fundamentales que todo usuario debe conocer así como una breve introducción a la hoja de trabajo y a la ayuda en línea.

Using Help remite a una página en la que aparece información sobre el uso de la ayuda.

Topic Search permite encontrar los tópicos que comiencen de la manera especificada (véase la figura siguiente). De esta forma se puede investigar si existe un comando que lleve a cabo ciertas acciones y cuyo nombre esperamos que esté relacionado con su acción.



Full Topic Search permite encontrar las páginas de ayuda en las que aparecen la palabra o palabras especificadas.



Finalmente comentaremos que la opción **History** permite revisitar cualquiera de las páginas de ayuda que se hayan invocado durante la sesión de Maple.

❑ Matemáticas con Maple

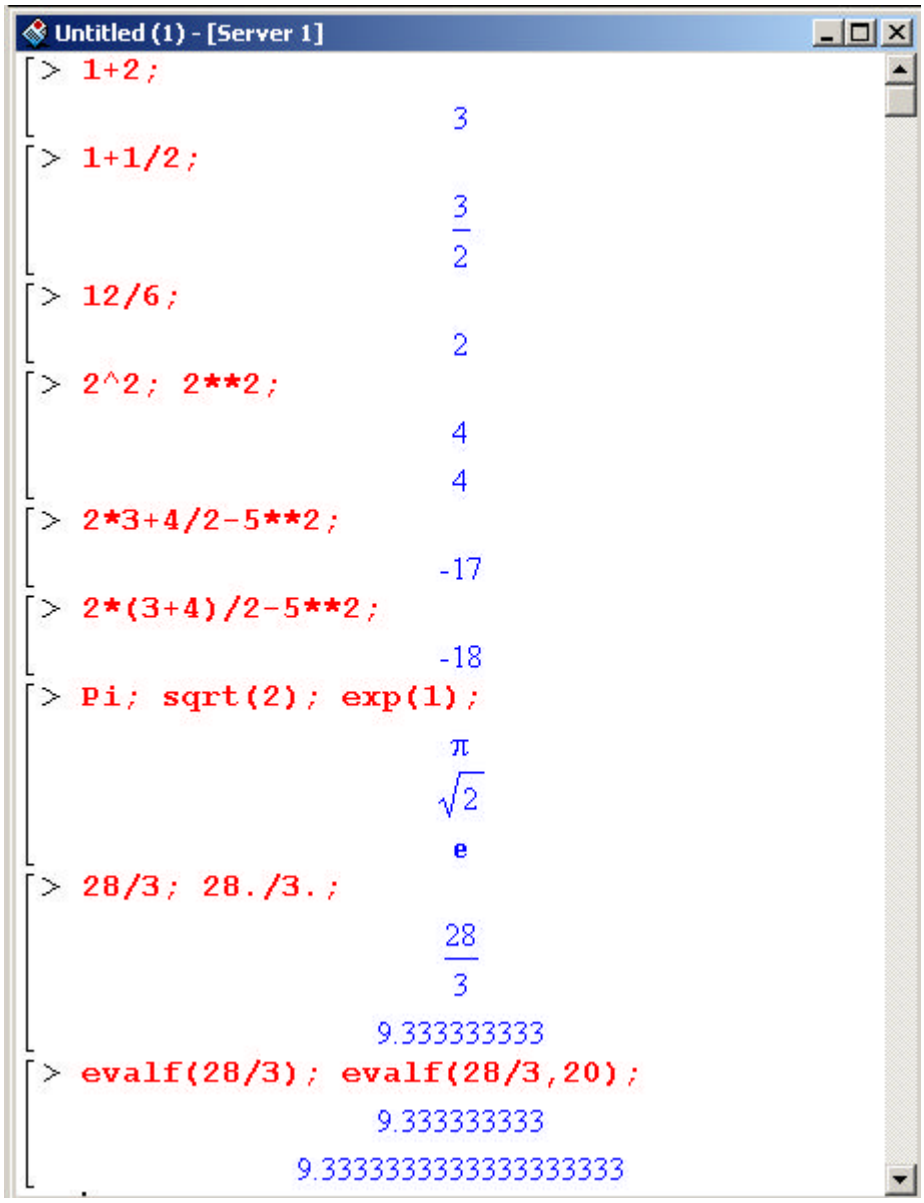
Los cálculos más básicos que se pueden realizar con Maple son numéricos. Maple opera como una calculadora convencional con enteros y números en coma flotante. Además es capaz de realizar cálculos exactos con números racionales: el resultado de la operación 2+1/2 es 5/2 que para Maple es un objeto totalmente diferente del número en coma flotante 2.5.

Sin embargo, Maple no sólo trabaja con números racionales sino también con expresiones, variables, conjuntos, listas, sucesiones, polinomios, matrices y muchos otros objetos matemáticos. Además es un lenguaje de programación completo que contiene procedimientos, tablas y otras estructuras.

Los cálculos se llevan a cabo utilizando los llamados operadores aritméticos ya mencionados con anterioridad, que son **+**, **-**, *, / ,**^** (******). Su orden de prioridad es justamente inverso al que hemos usado para enumerarlos. De esta forma una exponenciación será siempre la primera operación que se realice seguida de los productos y las divisiones (ambas con la misma prioridad) y finalmente las sumas y restas indistintamente. La prioridad se cambia por medio de paréntesis de igual forma que en los cálculos a mano.

En la siguiente imagen vemos algunos ejemplos. En la cuarta línea de comandos ilustramos las dos posibles representaciones del operador exponencial. En las líneas quinta y sexta se constata cómo los paréntesis alteran la prioridad de los operadores aritméticos, lo que conduce a resultados diferentes. Nótese que el resultado que obtiene Maple es, en ambos casos, el que obtendríamos nosotros si realizásemos estos mismos cálculos con lápiz y papel.

Los números 1, 2, 1/ 2 son, para Maple, números en aritmética exacta mientras que si los escribimos como 1., 2., 1./ 2. pasan a ser números en coma flotante. Obsérvese la diferencia que existe entre la salida que proporciona el comando 28/3 y la que proporciona el comando 28./3. en la penúltima instrucción: en el primer caso Maple trabaja en aritmética exacta y como 28 no es divisible entre 3 se queda con el número racional 28/3, mientras que en el segundo trabaja con aritmética de punto flotante (con 10 dígitos significativos y redondeo) dando una aproximación al resultado de la división, como haría una calculadora.



En la séptima línea de comandos vemos cómo se representan de forma simbólica algunos números irracionales: **Pi** es el número π, **sqrt(2)** es la raíz cuadrada de dos y **exp(1)** es el número **e** base del logaritmo neperiano. Esta representación simbólica permite cálculos exactos con estos números.

La última línea muestra cómo actúa el comando **evalf** : da la expresión en coma flotante de su primer argumento, el segundo argumento indica el número de dígitos significativos que se usa en la representación. Nótese que el segundo argumento es opcional, si no aparece se usan 10 cifras .

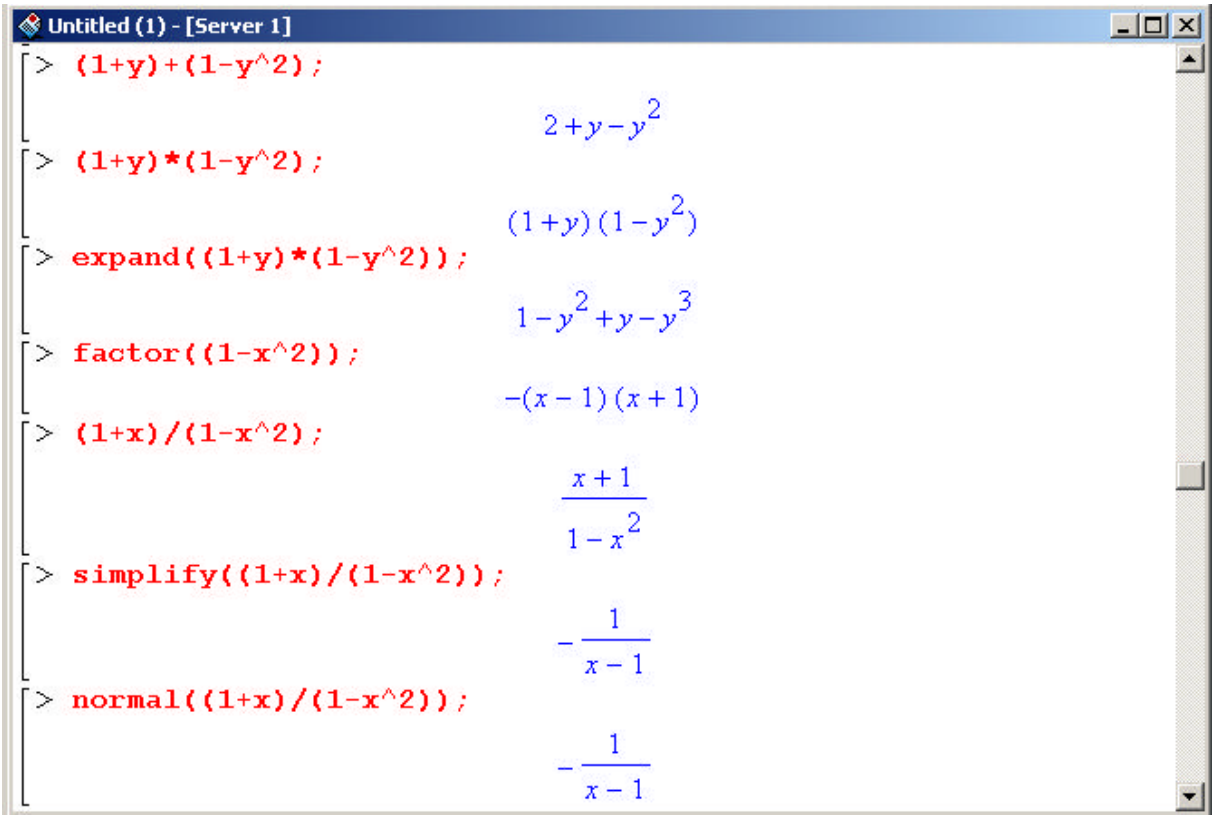
FUNCIONES MATEMÁTICAS

Maple conoce todas las funciones matemáticas estándar. Damos una pequeña lista de las más básicas:

- Funciones trigonométricas: **sin**, **cos**, **tan**,...
- Funciones hiperbólicas: **sinh**, **cosh**, **tanh**, ...
- Función exponencial: **exp** y logaritmos: **ln** (neperiano), **log[10]** (base 10).
- Función raíz cuadrada: **sqrt** .
- Redondeo al entero más próximo: **round**, truncación a la parte entera: **trunc**, parte fraccionaria: **frac**.

MANIPULACIÓN DE EXPRESIONES. VARIABLES Y SU ASIGNACIÓN.

En la imagen siguiente se pueden ver algunos cálculos simbólicos con expresiones y varios comandos que permiten su manipulación.

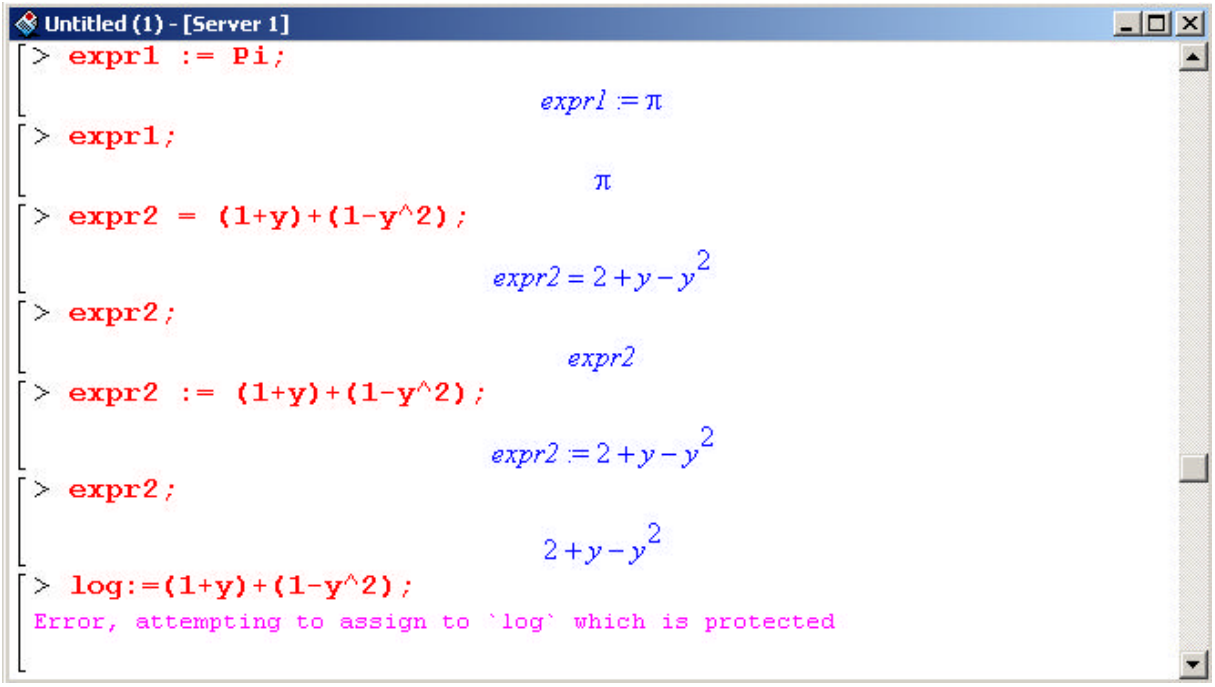


La instrucción **expand** desarrolla la expresión que va entre paréntesis, **factor** la factoriza, y **simplify** la simplifica. El comando **normal** pone el mismo denominador a las fracciones que aparezcan en la expresión entre paréntesis y elimina factores comunes del numerador y denominador. Como se observa en el ejemplo sirve también para simplificar la expresión racional obteniendo el mismo resultado que **simplify**.

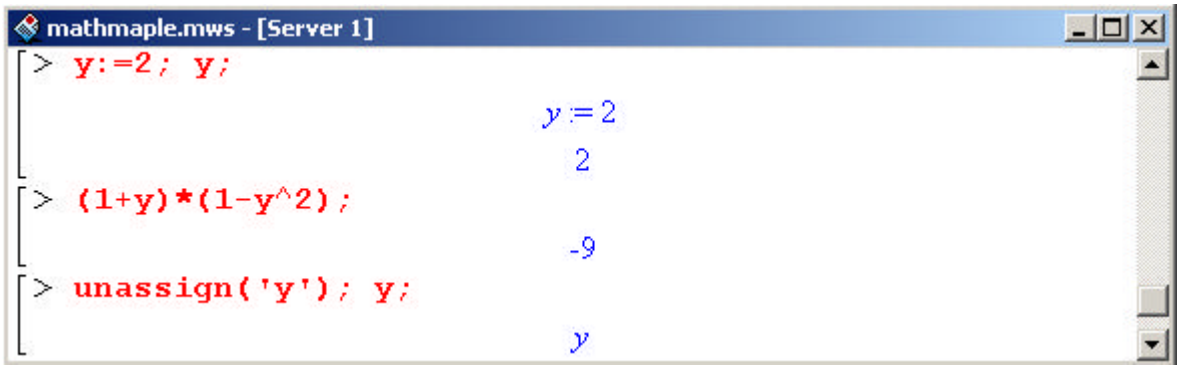
Maple puede trabajar con variables. Los nombres de variables se forman con letras, números y el signo *underscore* y han de ser distintos de las palabras reservadas del sistema. A las variables se les puede asignar valores. Las asignaciones en Maple se hacen con el símbolo **:=** (mientras que = es el operador relacional de igualdad).

En la imagen siguiente:

1. Asignamos el valor π a la variable que hemos llamado expr1; comprobamos que esta asignación se ha realizado invocando el nombre de la variable: la salida es la deseada (instrucciones 1 y 2).
2. Sin embargo en la tercera línea hemos usado el operador = para llevar a cabo la asignación de la expresión $(1+y)(1-y^2)$ a la variable expr2. La asignación, lógicamente, ha fallado lo que constatamos invocando la variable en la siguiente línea. A continuación realizamos, ahora sí, la asignación.
3. Finalmente intentamos utilizar como nombre de variable la palabra log que es una palabra clave del sistema: se usa para nombrar la función logarítmica. Como vemos Maple devuelve una línea de error en la que se nos advierte de tal eventualidad evitando redefinir el significado de la expresión **log**. Este comportamiento es de agradecer puesto que es prácticamente imposible conocer todas las palabras reservadas.



Nótese que para formar expresiones se utilizan variables: en las expresiones de los ejemplos **y** es una variable no asignada, ya que si previamente a su aparición en la expresión se le hubiese asignado un valor Maple sustituiría **y** por ese valor en todos los lugares en los que apareciera. De esta forma si ese valor fuese numérico, al ejecutar la expresión, obtendríamos el resultado de las operaciones que aparecen en ella. Este es el caso en la hoja de trabajo siguiente.



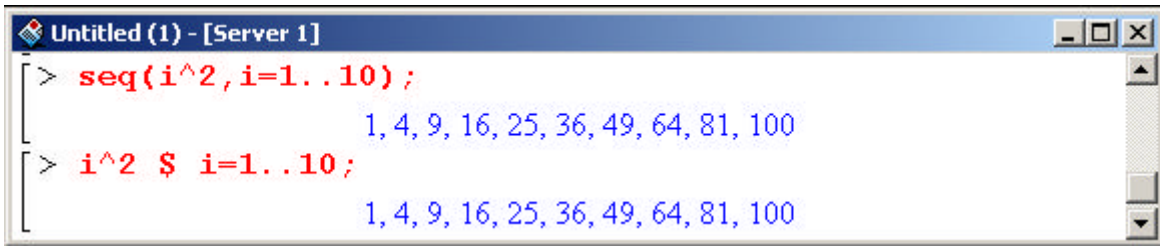
Para desasignar variables se utiliza el comando **unassign('var')** donde **var** es la variable que se quiere vaciar. Durante una sesión de trabajo Maple guarda en memoria todas las asignaciones realizadas hasta que ésta se cierra. Por ello en algunos momentos puede resultar conveniente vaciar la memoria: el comando **restart** realiza esta acción. Es aconsejable iniciar la hoja de trabajo con este comando.

OTROS OBJETOS MANIPULABLES

Maple, además de números, variables y expresiones, puede manipular estructuras más complejas. Entre ellas tenemos las *sucesiones de expresiones* que se crean usando la coma: por ejemplo el comando

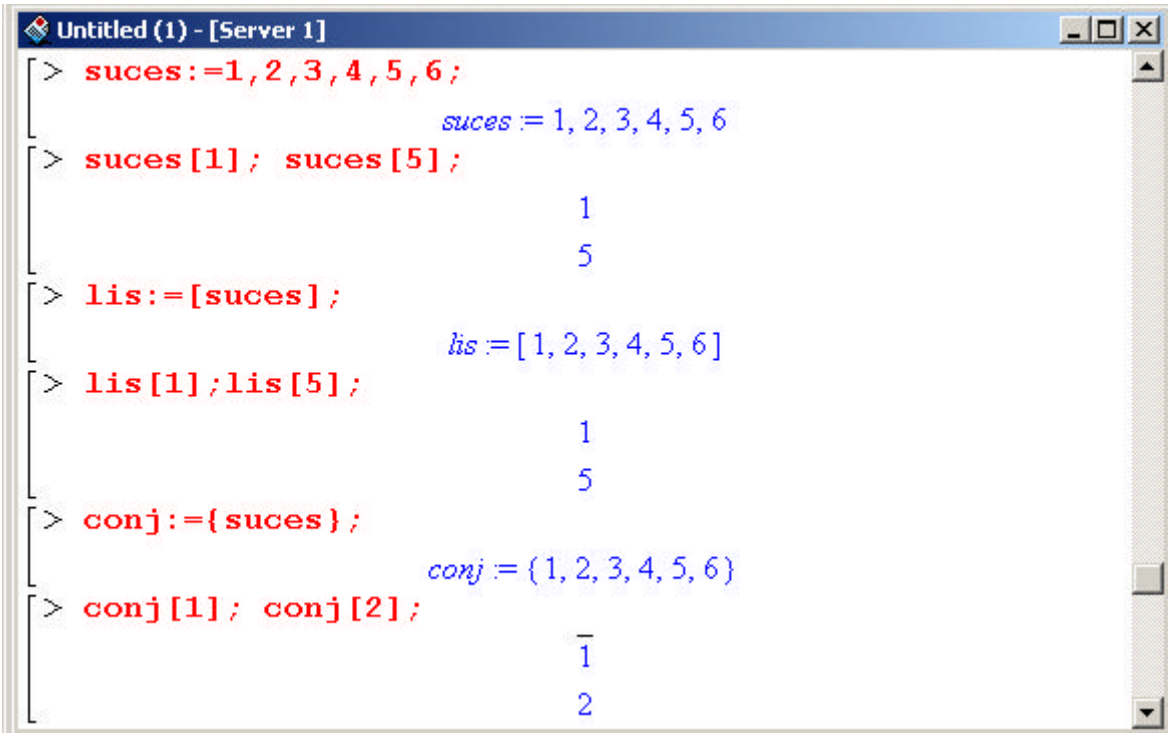
[> **1, 2+x, 3*x^2, 5;**

crea una sucesión con cuatro elementos que son las expresiones $1, 2+x, 3x^2, 5$. También se pueden crear sucesiones utilizando el operador de repetición **$** (**x$3** genera la sucesión **x,x,x**) o llamando al comando **seq**: por ejemplo **seq(f(i), i=1..3)** generará la sucesión **f(1), f(2), f(3),** donde f(i) es una expresión donde aparece la variable i (que no ha sido asignada previamente).



En la imagen vemos dos comandos que generan la sucesión de los cuadrados de los 10 primeros números naturales.

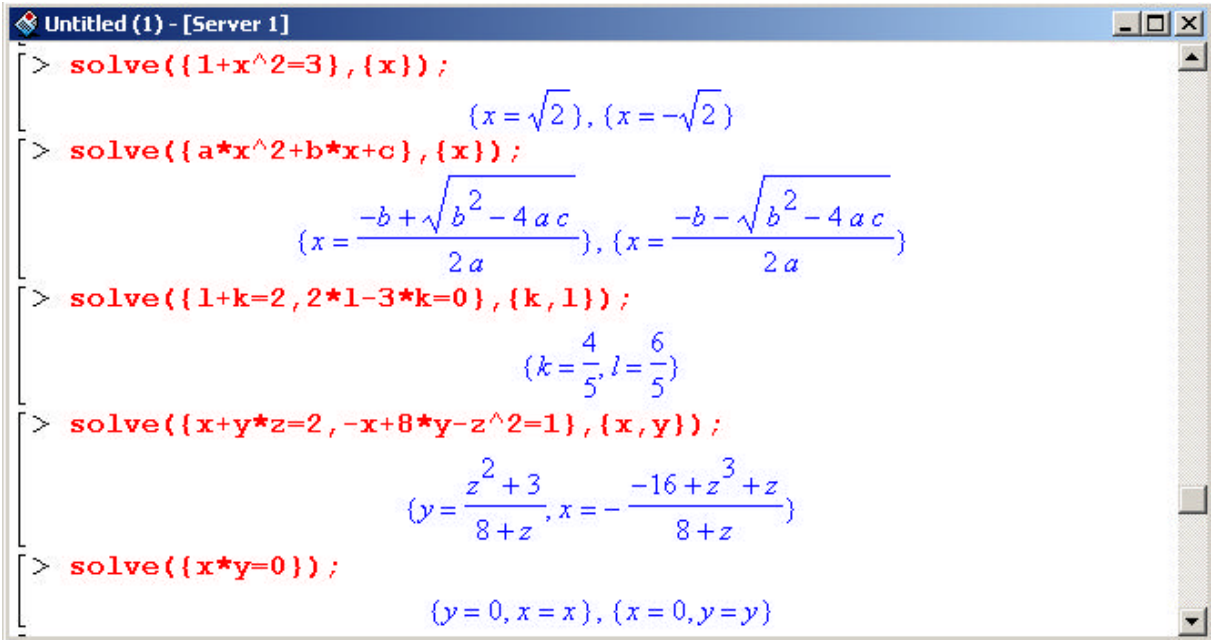
Otra estructura compleja es la *lista*. Básicamente una lista es una sucesión de expresiones encerrada entre corchetes. Similar a la lista es la estructura de *conjunto*: sucesión encerrada entre llaves. La diferencia fundamental entre ellas es que en la lista importa el orden: así las listas [1,1,,2,2] y [1,2,1,2] son distintas mientras que los conjuntos {1,1,2,2} y {1,2,1,2} son iguales al conjunto {1,2}. Es posible acceder a los elementos de las sucesiones, las listas o los conjuntos de forma sencilla como vemos en la pantalla siguiente.



Como extensión de la estructura de lista encontramos el *array*, que básicamente es una lista a cuyos elementos se les han asignado índices. Precisamente los *arrays* son las estructuras que se usan para definir vectores y matrices.

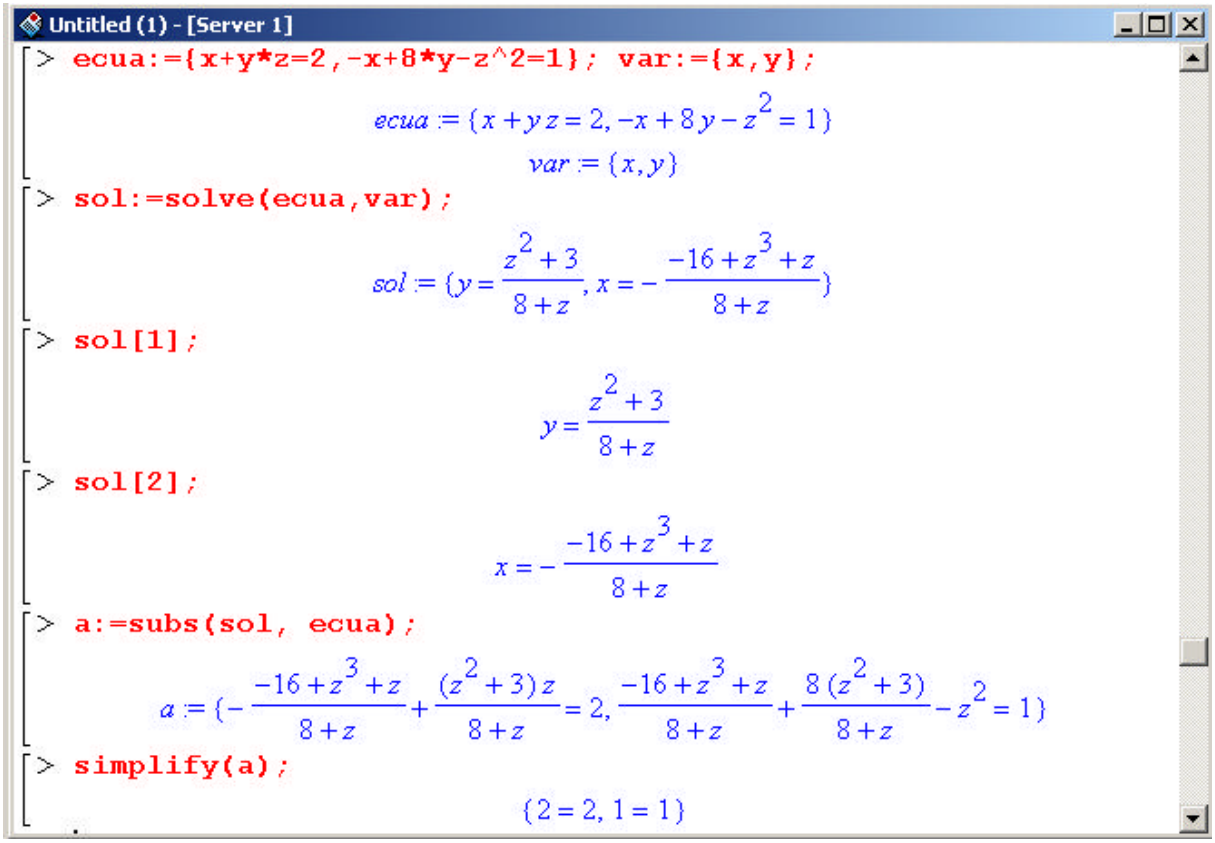
CÁLCULO DE SOLUCIONES

El comando **solve** tiene como propósito resolver de forma exacta ecuaciones o sistemas de ecuaciones. Tiene dos argumentos: en el primero se escriben, entre llaves, las ecuaciones a resolver separadas por comas y en el segundo, también entre llaves, las incógnitas.



Como se observa en las salidas que aparecen en la imagen, Maple da cada solución de la ecuación o del sistema de ecuaciones como un conjunto, es decir, entre llaves. Además, si no se especifican las incógnitas respecto a las que se quiere resolver Maple resuelve para todas; éste es el caso en la última instrucción.

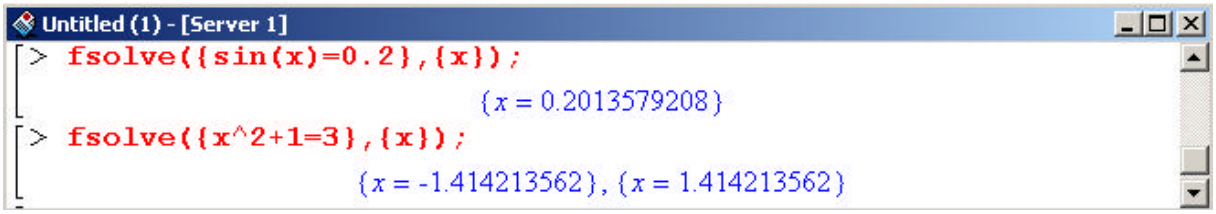
El comando **subs** sustituye una variable ***var*** en una expresión ***expr*** por un valor determinado ***val***. Como primer argumento se pasa la igualdad ***var=val*** ; el segundo argumento será la expresión en la que se desea hacer la sustitución. La instrucción queda **subs(** ***var=val*, *expr*** **)**. Véase su uso en la pantalla siguiente en la que se resuelve un sistema de ecuaciones y se verifica la solución obtenida.



En la imagen:

- A las variables ecua y var se les asignan sendos conjuntos: el de ecuaciones a resolver y el de incógnitas.
- En la variable sol guardamos las soluciones del sistema.
- Para comprobar de forma efectiva que las soluciones calculadas por Maple realmente lo son, sustituimos los valores obtenidos para x e y en las ecuaciones.
- Como después de realizar la sustitución los cálculos quedan indicados utilizamos el comando **simplify** para que se lleven a cabo las operaciones.

El comando **fsolve** es el equivalente en aritmética de punto flotante a **solve**. Así, este comando obtiene una aproximación numérica a la solución de una ecuación o un sistema de ecuaciones (mediante un método numérico). En general calcula sólo una solución; sin embargo para ecuaciones polinómicas busca todas las raíces reales.



Gráficos con Maple

Maple incluye potentes capacidades gráficas que permiten realizar representaciones bidimensionales, tridimensionales e incluso animaciones. El programa es muy flexible en lo que a la entrada de datos se refiere de tal forma que es posible representar funciones dadas en forma explícita, curvas y superficies especificadas a través de expresiones paramétricas e incluso se pueden manejar lugares geométricos definidos en forma implícita. Por otra parte, el sistema otorga al usuario control total sobre el resultado de modo que, por ejemplo, es posible cambiar desde los colores de los distintos objetos hasta las fuentes utilizadas en los títulos o las etiquetas de los ejes.

GRAFICOS 2D

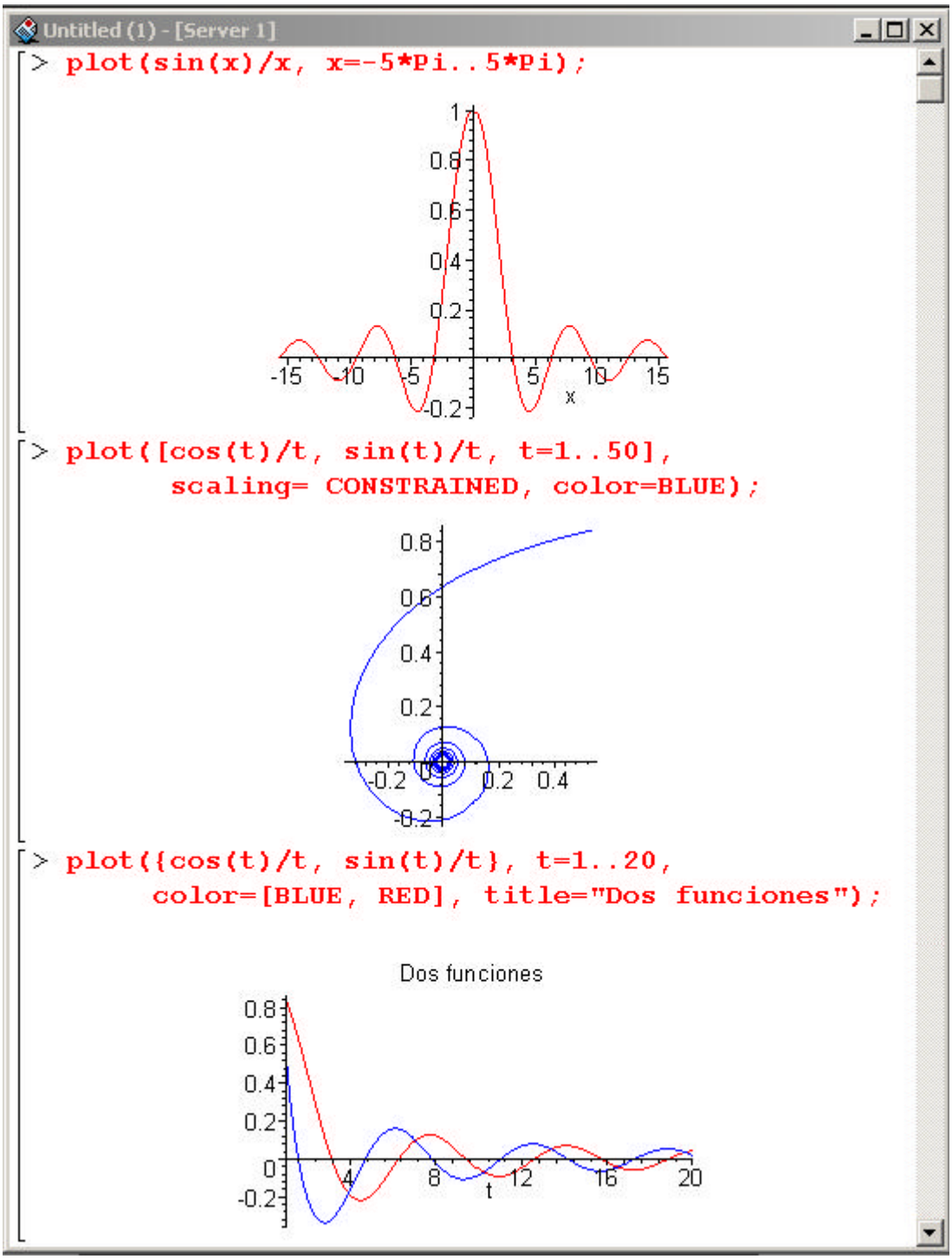
El comando básico para la representación de funciones en el plano es **plot**. En la siguiente figura se ilustra el empleo de dicho comando a través de tres ejemplos. El primero de ellos muestra la sintaxis básica de la instrucción: su primer argumento es la función que deseamos representar, en este caso se trata de $f(x) = \dfrac{sen(x)}{x}$. El segundo argumento especifica la variable independiente y su rango de variación.

La segunda llamada a **plot** ilustra cómo representar curvas dadas en forma paramétrica. El primer argumento es ahora una lista con tres elementos: los dos primeros constituyen la expresión paramétrica de la curva espiral

$$\begin{cases} x(t) = \dfrac{\cos(t)}{t} \\ y(t) = \dfrac{sen(t)}{t} \end{cases}$$

y el tercero especifica el parámetro y su rango de variación. El resto de los argumentos que aparecen en la expresión son optativos y simplemente especifican opciones que modifican el aspecto de la gráfica. Así, la opción **scaling** con el valor **CONSTRAINED** especifica que deben usarse las mismas unidades en los dos ejes. La opción **color** indica el color que debe usarse para la gráfica de la función. Obsérvese que la instrucción completa para la realización de esta gráfica ocupa dos líneas. En general, si se desea escribir varias líneas de entrada antes de que el kernel de Maple las

interprete, debemos finalizar cada una de ellas pulsando la combinación de teclas [Shift]+[Return], excepto al final de la última donde pulsaremos simplemente [Return] para indicar al sistema que procese la instrucción o conjunto de instrucciones introducidas.

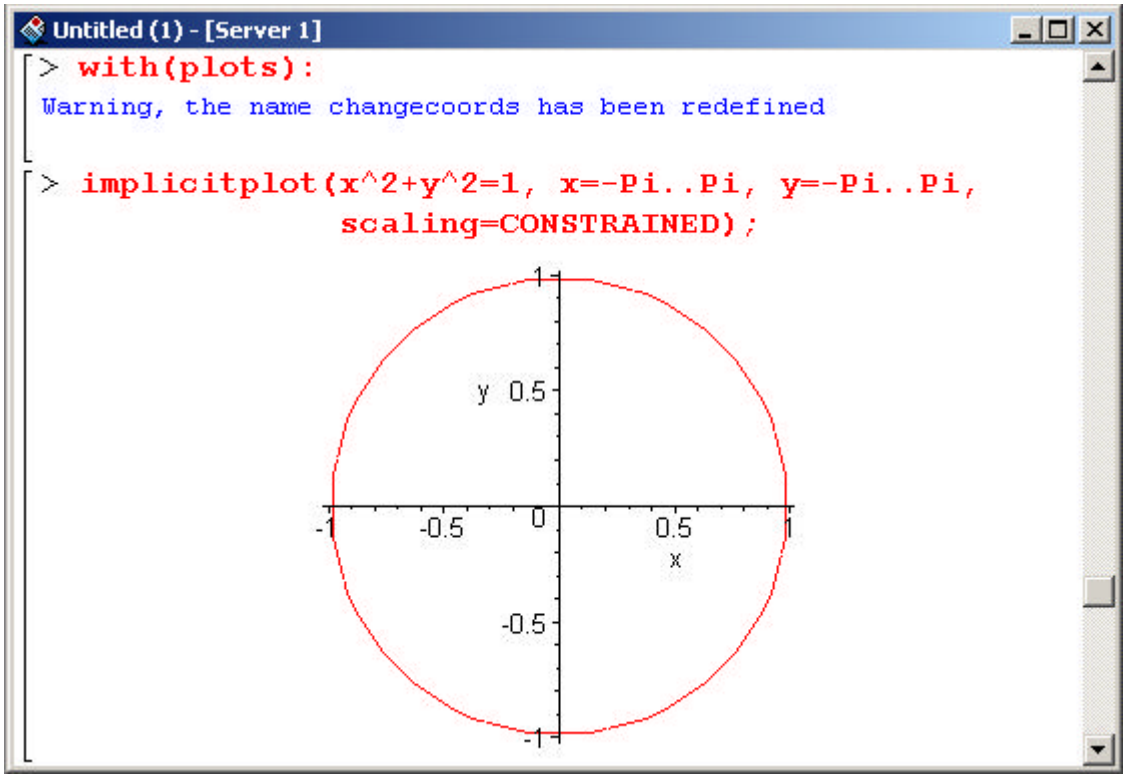


En el tercer ejemplo se utiliza **plot** para representar más de una función simultáneamente. En el caso que nos ocupa se representan las funciones $f(t)=\frac{\cos(t)}{t}$ y $g(t)=\frac{sen(t)}{t}$ en el intervalo [1,20]. Intencionadamente hemos seleccionado una notación que recuerda la del caso anterior para remarcar las diferencias. Nótese que ahora el primer argumento es un conjunto (se utilizan llaves en lugar de corchetes) que contiene solamente las funciones que queremos representar. El segundo argumento contiene la variable independiente y su rango de variación. La opción **color** toma como

valor una lista que contiene los colores que se utilizarán en la representación de cada una de las funciones especificadas en el primer argumento (f(t)→azul y g(t)→rojo). Por último, la opción **title** añade un título explicativo a la gráfica.

Existen multitud de opciones para el comando **plot** aparte de las que han aparecido en los ejemplos anteriores, con ellas es posible controlar aspectos tan variados del dibujo como el tipo de ejes que deben aparecer, la separación de las marcas sobre los mismos o el tipo de trazo que se usará en la representación de la gráfica.

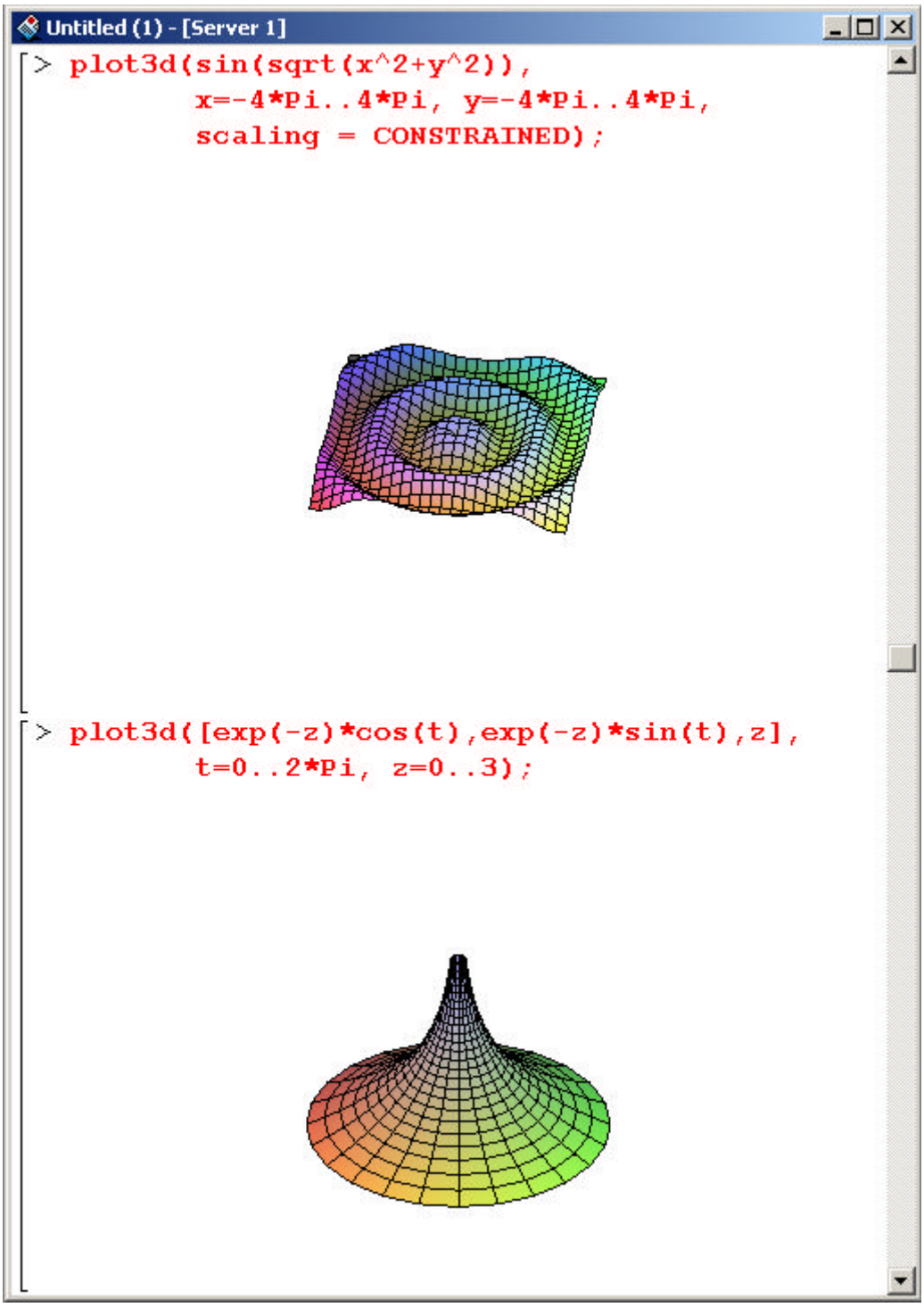
El paquete **plots** (Ver la sección **Los paquetes de Maple**, situada más adelante, para una descripción más detallada del concepto de *paquete*) incluye varios comandos avanzados para la realización de gráficos más específicos. De entre ellos destacamos dos: **animate** e **implicitplot**. Del primero nos ocuparemos al final de esta sección, en cuanto al segundo hay que decir que permite representar funciones dadas en forma implícita o, dicho de forma más rigurosa, es posible representar lugares geométricos definidos a través de una ecuación. En la siguiente figura se utiliza este comando para representar la circunferencia dada por $x^2 + y^2 = 1$. Si bien el rango de variación especificado para x e y es el intervalo [-π,π], Maple sólo representa la región de plano [-1,1]x[-1,1] en la que está incluida toda la circunferencia. El comando **with(plots)**, situado en la primera línea, sirve para cargar todas las funciones del paquete permitiendo de esta forma la utilización de **implicitplot**.



GRAFICOS 3D

La versión tridimensional del comando **plot** es la instrucción **plot3d** con una sintaxis muy similar a la de aquél. En la siguiente figura se ilustra el empleo de **plot3d** en la representación de funciones dadas en forma explícita y superficies expresadas en forma paramétrica.

En el primer caso se ha representado la función $f(x, y) = sen\sqrt{x^2 + y^2}$. Como puede observarse la función constituye el primer argumento en la llamada a **plot3d**. En el segundo y tercer argumento se especifican las variables independientes y sus rangos de variación. Por último hemos empleado la ya conocida opción **scaling** para mantener las mismas unidades a lo largo de los tres ejes.



En el segundo ejemplo el comando **plot3d** se utiliza para representar la superficie dada por

$$\begin{cases} x(t) = e^{-z}\cos(t) \\ y(t) = e^{-z}sen(t) \\ z(t) = z \end{cases}$$

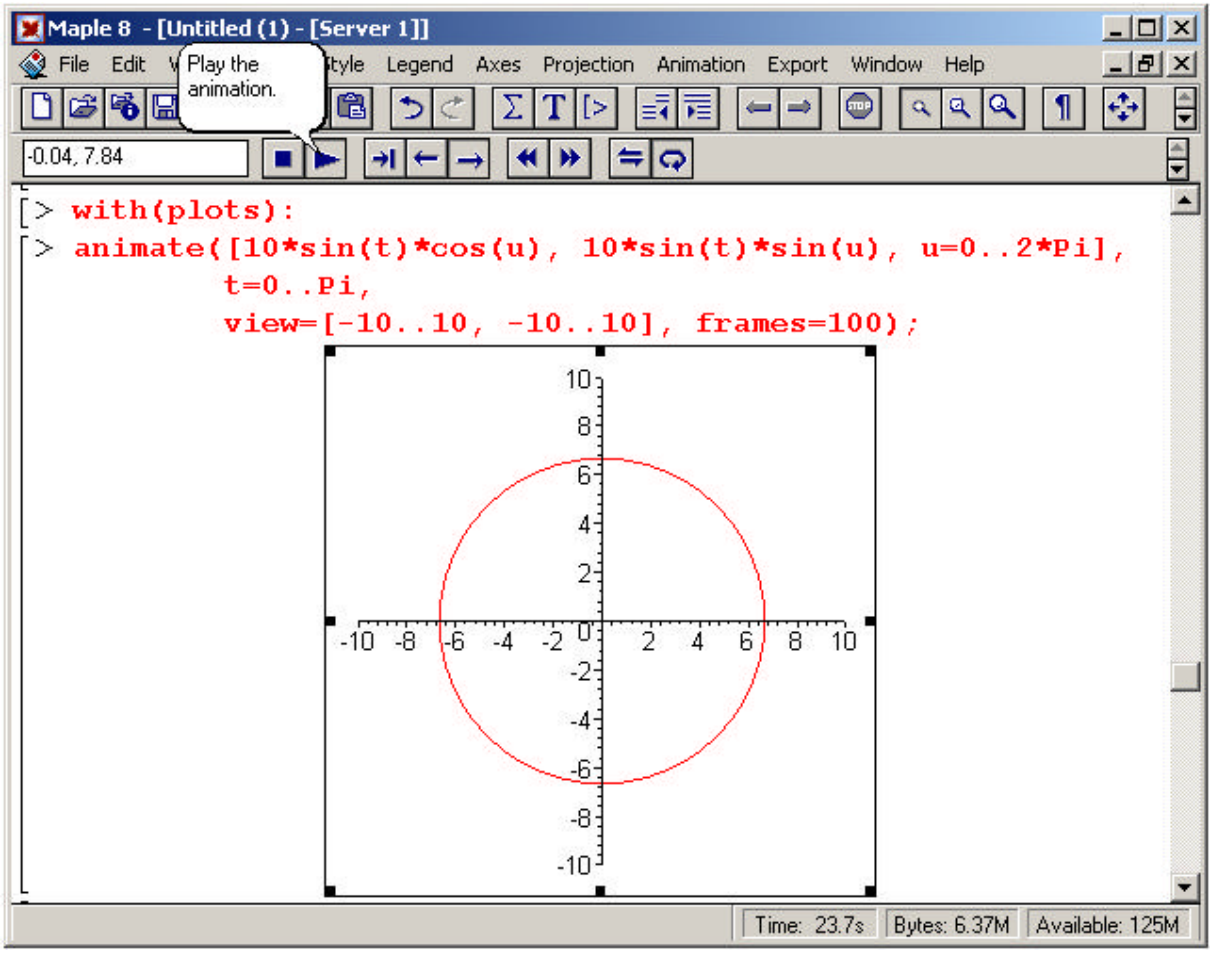
Nótese que el primer argumento lo constituye una lista que incluye la expresión paramétrica de cada una de las tres coordenadas. Los dos argumentos siguientes indican los parámetros y sus rangos de variación.

Las representaciones obtenidas mediante este procedimiento son auténticos modelos tridimensionales con los que es posible interaccionar. Basta arrastrar con el puntero del ratón sobre la figura para conseguir que ésta gire en la pantalla. De esta forma se puede observar la superficie desde cualquier ángulo lo que permite una completa comprensión espacial de la misma.

La mayoría de las opciones del comando **plot3d** están presentes también en **plot**, sin embargo hay algunas que son específicas de la versión tridimensional permitiendo, entre otras cosas, seleccionar la iluminación y el tipo de sombreado con el que se representan las superficies.

ANIMACIONES

Como comentamos anteriormente el paquete **plots** incluye utilidades para la generación de animaciones. En la siguiente figura se presenta un ejemplo basado en el uso del comando **animate**.



El primer argumento de **animate** lo constituye una lista que contiene la información necesaria para construir la gráfica de una circunferencia centrada en el origen de radio $10\,|sen(t)|$ dada en forma paramétrica (*u* es el parámetro de la circunferencia). Nótese que el parámetro *t*, que es el que juega el papel de tiempo en la animación, aparece definido en el segundo argumento de la llamada a **animate**. Puesto que el rango de variación de t es [0, π], la animación comienza con una circunferencia de radio nulo (un punto) que va creciendo hasta alcanzar un radio igual a 10 (t=π/2) y a continuación decrece hasta convertirse de nuevo en un punto.

La opción **view** fija los valores mínimos y máximos de las coordenadas x e y que son representados en la pantalla. Por último la opción **frames** permite especificar el número de "fotogramas" que constituirán la película. Un valor bajo para esta opción hace que se note el salto de un fotograma a otro y da como resultado animaciones poco fluidas. Por otro lado valores excesivamente altos de **frames** pueden agotar la memoria del ordenador.

Señalemos que para poder visualizar la animación es necesario seleccionar el dibujo creado por **animate** pinchando sobre el mismo. Es en ese momento cuando aparece en la barra contexto los botones con los que se controla la animación.

Finalizamos la sección indicando que es posible realizar animaciones de objetos 3D usando el comando **animate3d**, también incluido en el paquete **plots**.

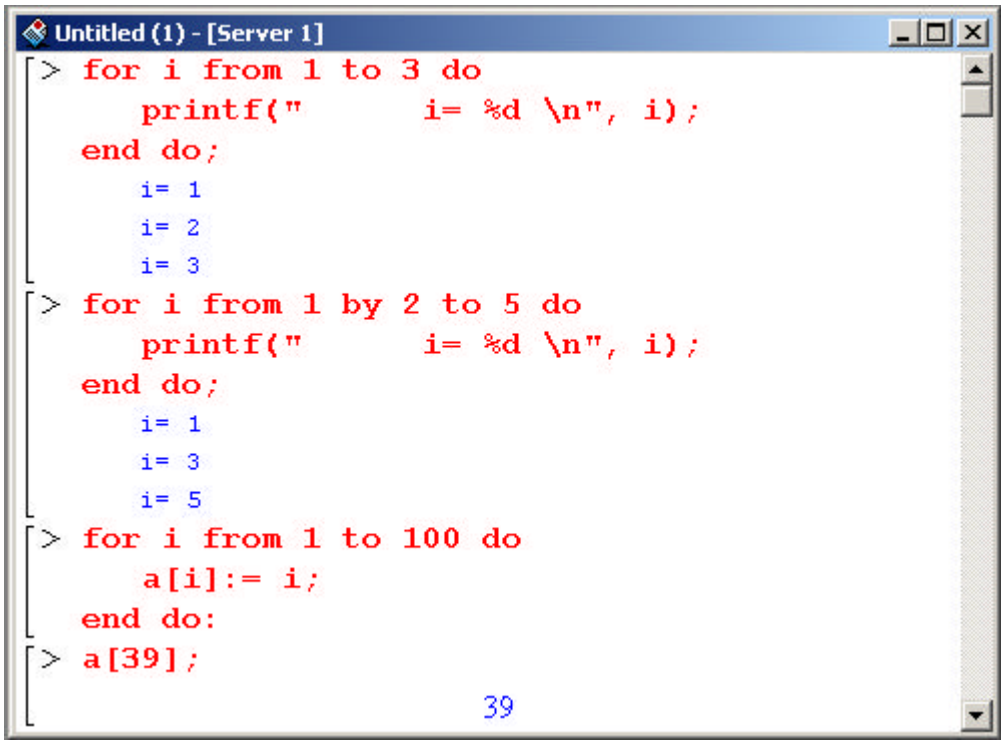
❑ Programación en Maple

Maple no es un programa diseñado sólo para el uso interactivo. Los comandos e instrucciones que se utilizan de manera individual desde la línea de comandos pueden agruparse formando programas que facilitan la realización de tareas repetitivas y nos proveen a su vez de nuevos comandos. A continuación se comentan, a modo de introducción al tema de la programación en Maple, las principales construcciones usadas en el desarrollo de programas. El lector interesado en una información más amplia puede consultar [3] como punto de partida.

CONSTRUCCIONES BÁSICAS

1.-Bucle for

Los bucles **for** se emplean para realizar tareas repetitivas un cierto número de veces. En la figura siguiente se pueden observar tres ejemplos concretos.



El primer caso presenta un bucle **for** que realiza tres iteraciones sobre la variable i. Dicha variable comienza tomando el valor 1 (**from 1**) y en cada ciclo incrementa su valor en una unidad hasta alcanzar el valor 3 (**to 3**). Obsérvese la palabra reservada **do** que aparece al final de la primera línea y que es la que marca el inicio de las instrucciones sobre las que debe actuar el bucle.

El cuerpo del bucle lo constituye una llamada a la función **printf** la cual será ejecutada en cada una de las tres iteraciones.

La función **printf** escribe expresiones en la salida de acuerdo con una cadena de formato. Su sintaxis de llamada es la siguiente

printf(formato, x1, ..., xn)

donde **formato** es una expresión encerrada entre comillas que contiene los caracteres que van a ser impresos junto con especificaciones de formato (que comienzan con el símbolo **%**) y otras secuencias de control de la salida. Las especificaciones de formato indican la forma en la que deben ser impresas las variables **x1, ..., xn**. En los ejemplos aparece la especificación de formato **%d** que indica que la variable **i** debe ser escrita como un número entero. La secuencia de control **\n** introduce un retorno de carro al final de cada impresión. Para más información acerca de **printf** puede consultarse la ayuda de Maple o cualquier manual que contenga información sobre el comando homónimo del lenguaje C.

Finalmente, la expresión **end do** señala el alcance del bucle **for**, es decir marca cuáles son las instrucciones a las que debe afectar la secuencia de iteraciones. El resultado final del bucle es la impresión de los tres valores que toma la variable i.

El segundo ejemplo es similar pero ahora el valor de la variable de iteración i se incrementa dos unidades (**by 2**) en cada ciclo, lo que hace que sólo se impriman los números impares de 1 a 5.

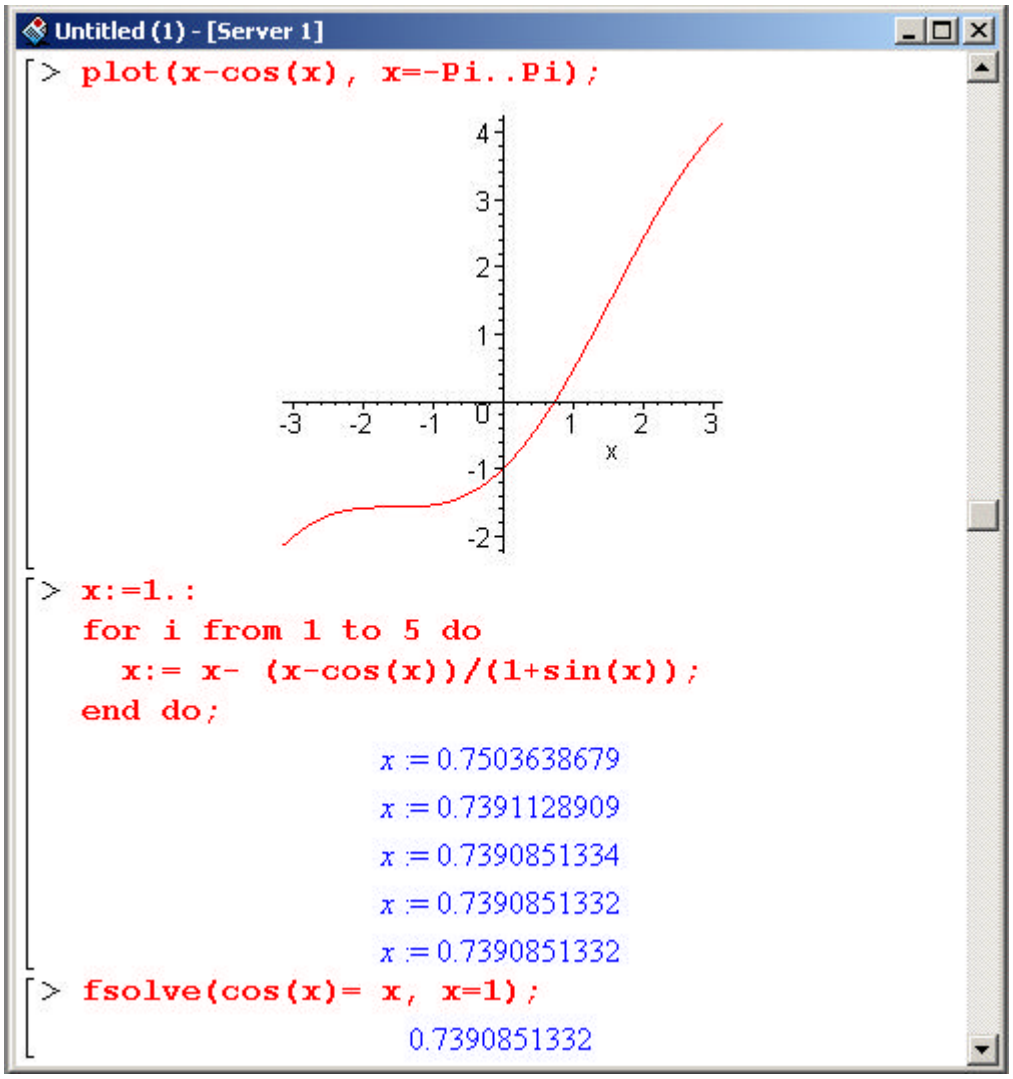
En el tercer caso el bucle **for** se utiliza para guardar los cien primeros números naturales dentro de la variable **a**. Nótese que al finalizar con dos puntos después de **end do** se suprimen las cien líneas de salida correspondientes a otras tantas asignaciones. La última instrucción comprueba que el elemento indexado con el número 39 dentro de la variable **a** contiene efectivamente el valor 39.

A continuación se presenta una aplicación del bucle **for** en la resolución de un problema matemático. Supongamos que estamos interesados en calcular las soluciones de la ecuación cos(x)=x. Evidentemente las soluciones de esta ecuación coinciden con los ceros de la función f(x)= x-cos(x). Para obtener estos últimos emplearemos un algoritmo conocido como método de Newton-Raphson que genera, a partir de una aproximación inicial x_0, una sucesión definida en forma recursiva como sigue

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Bajo ciertas condiciones la sucesión obtenida converge hacia un cero de la función f.

La siguiente figura muestra una pantalla de Maple con la resolución del problema. En primer lugar se ha efectuado la representación gráfica de la función f(x)= x-cos(x) usando el comando **plot**. A la vista de la gráfica se hace evidente la existencia de una solución cercana a x_0=1. Es fácil comprobar que, de hecho, la ecuación en la que estamos interesados sólo admite una solución: para ello basta notar que f(x) es monótona creciente y por lo tanto sólo puede cortar al eje de abscisas una vez. Tras la gráfica hemos utilizado un bucle **for** para implementar cinco iteraciones del método Newton-Raphson. Obsérvese que las dos últimas iteraciones arrojan el mismo resultado, lo que indica que el método ha sido convergente y proporciona la solución x=0.7390851332. Por último hemos empleado el comando **fsolve** para resolver directamente el problema y comprobar que obtenemos exactamente la misma solución.

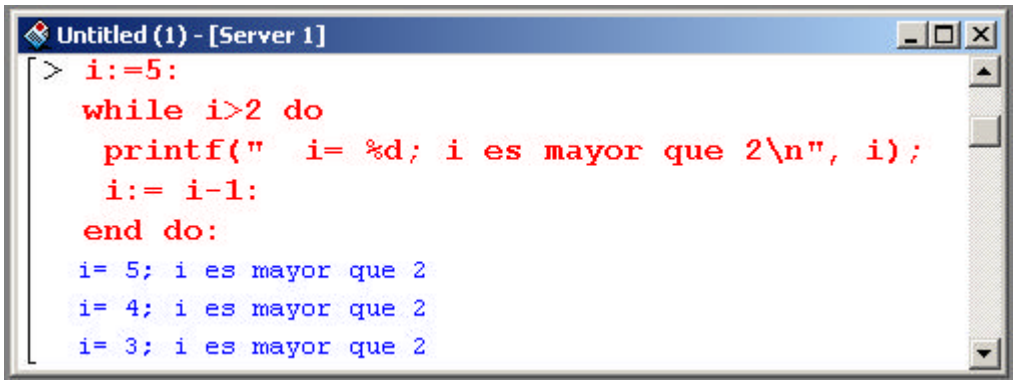


2.-Bucle while

Cuando deseamos realizar un bucle hasta que una cierta condición deja de satisfacerse se emplea la construcción **while**.

Tanto esta instrucción de control de flujo como la que veremos a continuación (**if**) puede necesitar de expresiones lógicas para formular la condición que gobierna el bucle. Una expresión lógica es aquella cuya evaluación da un resultado lógico, es decir, verdadero o falso. Las expresiones lógicas se forman utilizando operadores relaciones (**>** mayor, **<** menor, **<=** menor o igual, **>=** mayor o igual, **=** igual, **<>** distinto) y/u operadores lógicos (**and** y, **or** o, **not** no). Por ejemplo, la expresión **2>1 or 1>3** es lógica y su evaluación produce el resultado *true*.

En el ejemplo comenzamos inicializando la variable i con el valor 5. A continuación comienza un bucle **while** que imprime el valor de la variable i junto con el mensaje “i es mayor que 2” y seguidamente se disminuye el valor de i en una unidad. Después de la tercera iteración la variable i pasa a tener el valor 2 dejando de satisfacerse la condición que gobierna el bucle, con lo cual concluye su ejecución.



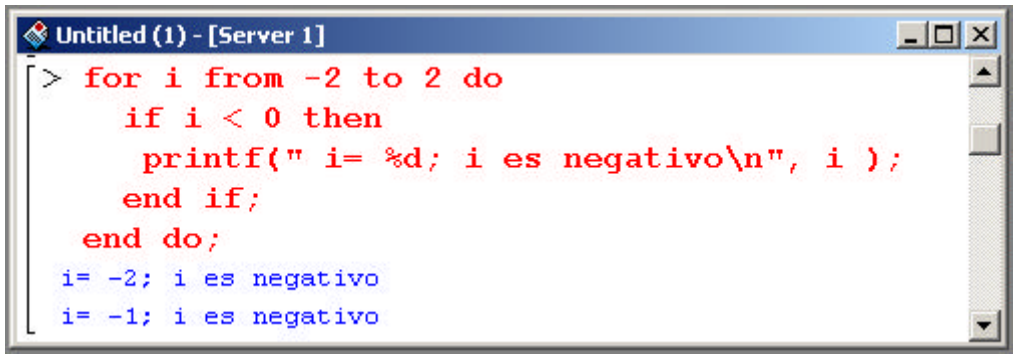
La instrucción **while** puede emplearse también en combinación con el bucle **for** de forma que la ejecución de éste se cancele en el momento en el que deja de verificarse una condición. De hecho ambas instrucciones forman parte de una expresión general de control de flujo cuya sintaxis es la siguiente

for <nombre> from <expr> by <expr> to <expr> while <expr>
do <sucesión de sentencias> end do;

2.-Sentencia if

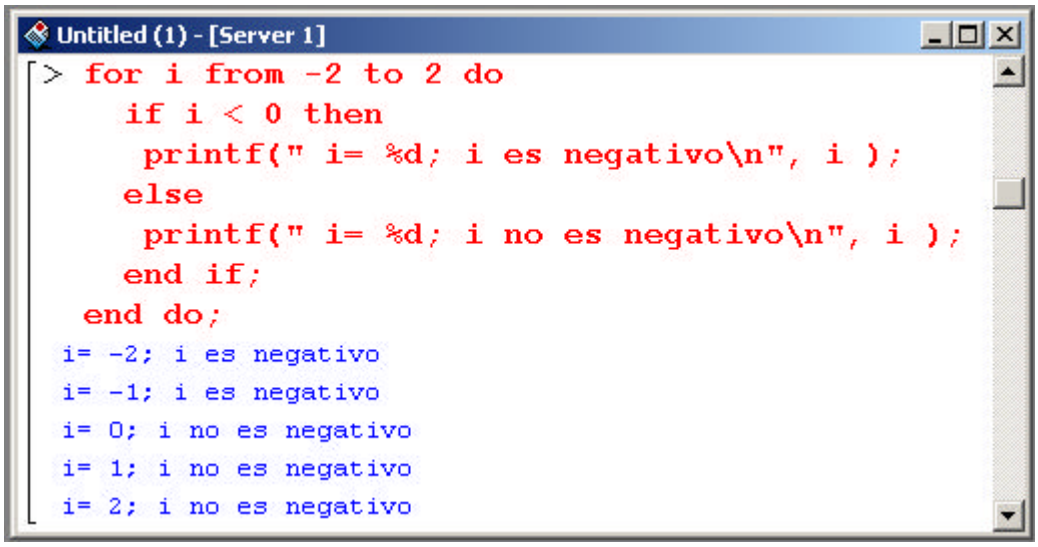
A menudo interesa ejecutar una instrucción o un grupo de instrucciones sólo si se verifica cierta condición. La sentencia **if** da respuesta a esta necesidad. Ilustraremos su empleo por medio de tres ejemplos en los que analizaremos su comportamiento dentro de un bucle **for**.

Comencemos describiendo el ejemplo recogido en la siguiente figura.

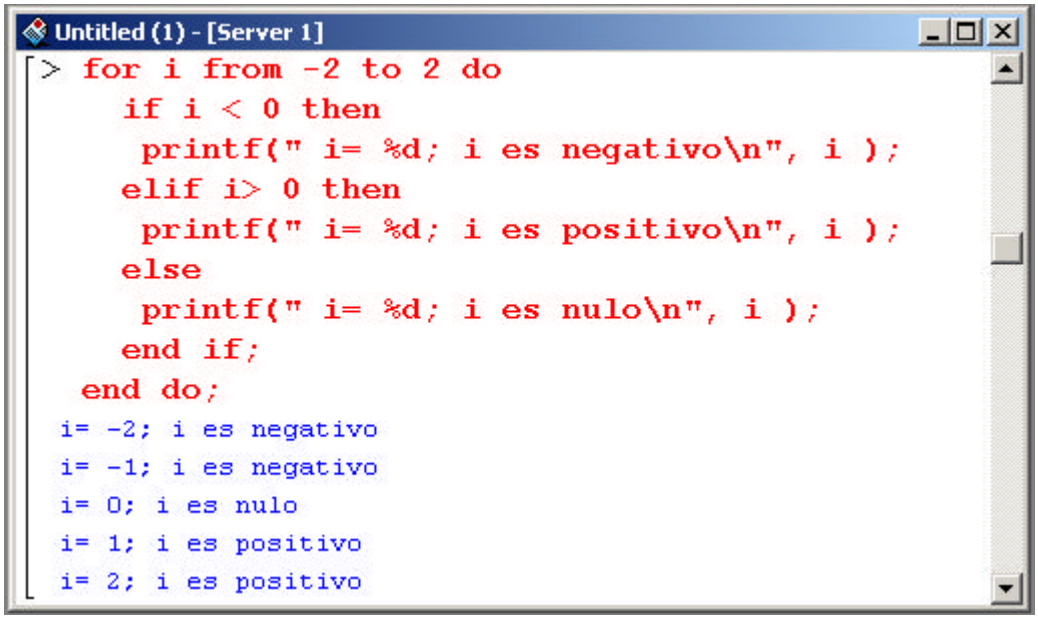


En primer lugar se define un bucle **for** que se ejecutará cinco veces. La sentencia **if** hace que la instrucción **printf** se ejecute únicamente para los valores negativos de i. Obsérvese la sintaxis de la sentencia **if** que incluye el término **then** detrás de la condición **i<0** y emplea la expresión **end if** para cerrar su alcance.

Hay ocasiones en las que, además de ejecutar ciertas instrucciones cuando se verifica una condición, deseamos que se ejecuten otras cuando no se verifica. Para ello basta con insertar la palabra reservada **else** seguida de las instrucciones correspondientes dentro del cuerpo de la sentencia **if**, tal y como se muestra en la siguiente figura.



Por último, si se quiere elegir entre la ejecución de un bloque de sentencias u otro dependiendo de la verificación de una u otra condición es posible insertar tantas líneas **elif** [condición] **then** como sean necesarias para dar cuenta de todas las posibilidades. En el tercer ejemplo, recogido en la siguiente figura, se puede apreciar esta construcción. **else** <**sucesión de sentencias**> es opcional y puede no aparecer, pero si aparece debe de estar al final, como en el ejemplo.



PROCEDIMIENTOS

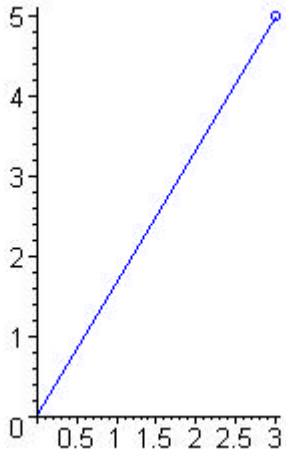
Al usar Maple frecuentemente en modo interactivo se descubre que hay secuencias de comandos que se repiten a menudo. El lenguaje de programación Maple permite agrupar todos esos comandos en unidades que se denominan *procedimientos*. La forma más sencilla de crear un procedimiento consiste en encapsular la secuencia de instrucciones que se habría introducido interactivamente entre las sentencias **proc()** y **end proc**.

En la siguiente figura se define un procedimiento sencillo que hemos denominado **suma**. El procedimiento toma dos argumentos y simplemente devuelve el valor de su suma.

```
Untitled (1) - [Server 1]
> suma:= proc(x,y)
    x+y;
  end proc;
                suma := proc(x, y) x + y end proc
> suma(2,7);
                              9
```

El verdadero poder de los procedimientos reside en la posibilidad de combinar cualquier tipo de comando de Maple. Por ejemplo, en la siguiente figura se muestra un procedimiento que admite como argumentos las coordenadas (x,y) de un punto del plano y devuelve un dibujo con el punto en cuestión así como su distancia hasta el origen de coordenadas.

```
Untitled (1) - [Server 1]
> f:= proc(x,y)
   # Calcula e imprime la distancia
   printf("Distancia al origen: %f\n",
          sqrt(x^2+y^2)):
   # Dibuja el punto sobre el plano
   plot([[x,y]], [[0,0],[x,y]] ],
      scaling=CONSTRAINED, color=blue,
      style=[point, line], symbol=circle);
  end proc:
> f(3,5);
Distancia al origen: 5.830952
```



El símbolo # que aparece en dos de las líneas del anterior procedimiento sirve para introducir comentarios que facilitan la comprensión de los programas. Maple interpreta como comentario todos los caracteres que siguen al símbolo # dentro de la misma línea. Observesé que al finalizar con **:** la expresión **end proc** evitamos que Maple saque por pantalla la definición del procedimiento completo como sucedía en el primer ejemplo.

Los procedimientos creados por el usuario constituyen nuevas instrucciones que, a su vez, pueden ser llamadas por otros procedimientos. De hecho muchos de los comandos de Maple son simples procedimientos cuyo código es accesible, de modo que pueden ser modificados para adaptarlos a las

necesidades específicas de cada situación. La mayoría de estos comandos se encuentran agrupados en los denominados *paquetes* que son el objeto de la siguiente sección.

Los paquetes de Maple

En el momento del arranque el programa Maple carga un número relativamente reducido de comandos en memoria. Existen sin embargo multitud de instrucciones adicionales diseñadas para resolver problemas más específicos o realizar tareas más avanzadas. Estos comandos se encuentran organizados en colecciones que en inglés reciben el nombre de "packages" y que nosotros traducimos como *paquetes*. Puede decirse que un paquete de Maple es un conjunto de comandos diseñados para facilitar la resolución de problemas en un área específica. En algunos casos el número de nuevas funciones es tan grande que se han terminado reorganizando en *subpaquetes*.

Al pasar el nombre de un paquete como argumento al comando **with** se consigue cargar en memoria todas las funciones del paquete en cuestión. Un ejemplo concreto, referente al paquete **plots** aparece en el apartado ANIMACIONES de la sección **Gráficos con Maple**. Hay que decir, sin embargo que también es posible cargar parcialmente un paquete o incluso se puede llamar a una función sin necesidad de cargar el paquete correspondiente. En este último caso la sintaxis de llamada es la siguiente

paquete[función](argumentos)

A continuación se comentan brevemente alguno de los paquetes más representativos.

Student Este paquete está diseñado para la enseñanza y el aprendizaje de los cursos de matemáticas a un nivel básico. Proporciona, además, una buena introducción para la comprensión del sistema Maple. Contiene el subpaquete **Calculus1** que cubre el material esencial para un curso sobre funciones de una variable.

plots Contiene funciones que facilitan la representación de curvas y superficies en dos y tres dimensiones. También contiene instrucciones para la realización de animaciones.

linalg En este paquete se encuentran recogidas funciones de utilidad en el álgebra lineal. Con ellas se pueden realizar fácilmente productos de matrices, cálculo de inversas o de exponenciales, obtención de polinomios característicos, valores propios, formas de Jordan ...

stats Este paquete proporciona funciones como medias o cuantiles para el análisis de datos y funciones para construir histogramas y realizar representaciones gráficas. Contiene varios subpaquetes como **anova** para el análisis de la varianza, **fit** para efectuar regresiones lineales, **statevalf** para la evaluación numérica de distintas distribuciones de probabilidad o **statplots** que proporciona funciones para crear diferentes tipos de gráficos estadísticos.

DEtools Está integrado por funciones que facilitan el trabajo con ecuaciones diferenciales. Con las utilidades contenidas en este paquete se puede, entre otras tareas, realizar representaciones gráficas de campos vectoriales, trabajar con secciones de Poincaré, manipular operadores diferenciales, simplificar sistemas y construir soluciones en forma cerrada.

Además de los anteriormente enumerados, existen decenas de paquetes que abarcan los más variados tópicos: **combinat** sobre combinatoria, **finance** para cálculos financieros, **group** para trabajar en teoría de grupos, **networks** útil para manejar grafos y redes, **simplex** diseñado para la optimización lineal, **Maplets** si se pretende crear interfaces gráficos de usuario, **Matlab** para usar funciones del sistema Matlab en una sesión Maple... Con tal variedad de paquetes el usuario puede estar seguro de que, sea cual sea el problema sobre el que trabaje, el sistema Maple siempre le ofrecerá ayuda especializada sobre la materia.

BIBLIOGRAFÍA

[1] Meal, K.M.; Hansen, M.L:; Rickard, K.M. (1996): “Maple learning guide”, Waterloo Maple.Springer Verlag.

[2] Redfern, D. (1996): “The Maple Handbook”, Springer Verlag.

[3] Monagan, M.B.; Geddes, K.O.; Labahn, G.; Vorkoetter, S. (1996): “Maple Programming guide”, Springer Verlag.

[4] Garvan, F.(2001): ”The Maple Book”, Chapman&Hall/CRC.

ENLACES

[1] http://www.maplesoft.com/main.shtml
Institución: Waterloo Maple, Inc
Título: Waterloo Maple – Advancing Mathematics
Descripción: Página principal de la compañía que desarrolla el sistema Maple. Entre otras cosas contiene información acerca de sus productos, congresos sobre Maple y enlaces a otras direcciones Web cuyos contenidos se relacionan con este software.

[2]..http://www.mapleapps.com/maplelinks/sh_resources.shtml
Institución: Waterloo Maple, Inc
Título: Maple Resources and Web Sites
Descripción: Página oficial de Maple sobre recursos y direcciones de interés internet. Contiene enlaces a páginas con aplicaciones de Maple, direcciones de soporte técnico y páginas con interesantes tutoriales de introducción a Maple.

[3] http://www.mcs.dundee.ac.uk:8080/~dfg/MapleExternal.html
Institución: Dundee University, UK
Título: Online tutorials on basics of Maple
Descripción: Contiene varios documentos en formato PostScript sobre asuntos básicos de Maple.

[4] http://www.math.uic.edu/maple/labs/index.html
Institución: University of Illinois, USA
Título: Introduction to Maple
Descripción: Contiene documentos que ilustran desde la sintaxis básica de Maple hasta el desarrollo de animaciones. Esta página usa "frames".

[5] http://www.indiana.edu/~statmath/math/maple/gettingstarted/index.html
Institución: Indiana University, USA
Título: Getting Started with Maple
Descripción: Documento disponible formatos HTML y PDF. Se trata de un completo tutorial en el que se describe detalladamente desde la entrada en el sistema Maple hasta la creación de funciones y el uso de paquetes.

[6] http://web.mit.edu/afs/athena/astaff/project/logos/olh/Math/Maple/Maple.html
Institución: MIT, USA
Título: Maple Introduction
Descripción: Tutorial que cubre aspectos tanto básicos como avanzados del lenguaje Maple.

[7] http://www.math.tamu.edu/~boas/courses/math696/Maple.html
Institución: Texas A&M Univesity, USA
Título: Maple pages by Harold Boas
Descripción: Curso básico sobre Maple. También contiene enlaces a otros documentos y una pequeña introducción al programa MATLAB

[8] http://wmatem.eis.uva.es/~matpag/
Institución: Departamento de Matemática Aplicada a la Ingeniería, Universidad de Valladolid
Título: Invitación a las Matemáticas
Descripción: Página de interés para estudiantes de carreras técnicas. En ella se recogen algunas cuestiones de uso muy común y que a menudo quedan fuera de los temarios oficiales. Entre otros tópicos se discuten la teoría elemental de conjuntos, los números complejos o la clasificación de las cónicas y las cuádricas.