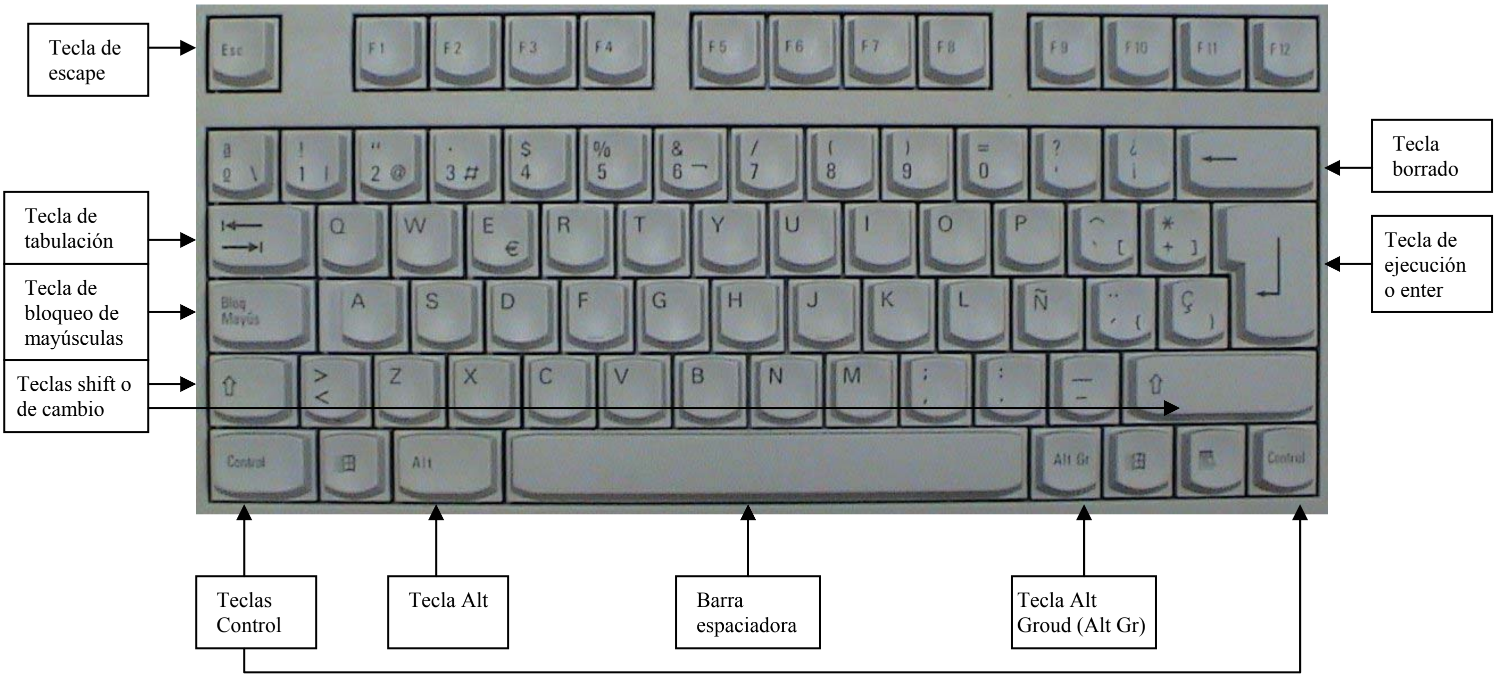
SESIÓN 0

1. MANEJO DEL TECLADO
2. SOBRE MATHEMATICA
3. METODO DE TRABAJO BASICO
4. OPERACIONES BASICAS Y MANEJO DE PALETAS
5. TIPOS DE DATOS
6. TIPOS DE DATOS Y PRECISION
7. FUNCIONES
8. VARIABLES, SÍMBOLOS Y ASIGNACIÓN DE VALORES
9. LISTAS
10. MATRICES Y VECTORES DE $\mathbb{R}^n$

- ELEMENTOS DEL PROGRAMA

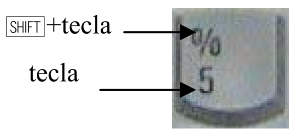
1. MANEJO DEL TECLADO

El teclado es la herramienta elemental para la introducción de datos en cualquier equipo informático. El teclado básico de cualquier PC está compuesto por las teclas correspondientes a las letras y dígitos numéricos junto con otras teclas dedicadas a funciones especiales. El siguiente gráfico muestra la disposición estándar en un teclado de un ordenador compatible destacándose además las teclas de uso especial:

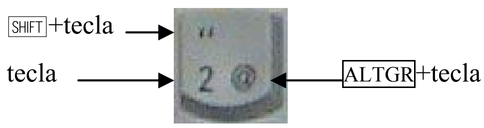


Si bien en la mayoría de las ocasiones es obvia la mecánica de uso del teclado, haremos a continuación una breve descripción de su modo de funcionamiento:

- Para escribir una letra del alfabeto o dígito numérico pulsaremos la tecla en que aparece.
- Para escribir una letra en mayúsculas, pulsaremos la tecla ‘shift’ y, sin soltar, la tecla correspondiente a la letra deseada.
- Cuando es preciso escribir un texto largo en mayúsculas, resulta incómodo mantener pulsada continuamente la tecla ‘shift’. En su lugar podemos pasar al modo mayúsculas pulsando la tecla de ‘bloqueo de mayúsculas’. En el modo mayúsculas todas las letras aparecen directamente en mayúsculas con solo pulsar su tecla. Para regresar al modo normal bastará pulsar nuevamente la tecla ‘bloqueo de mayúsculas’.
- Cuando en una tecla aparecen dos símbolos:
  - Para que aparezca el símbolo inferior bastará pulsar la tecla.
  - Para que aparezca el símbolo superior pulsaremos la tecla ‘shift’ y sin soltar pulsaremos la tecla en cuestión.



- Cuando en una tecla aparecen tres símbolos:
  - Para que aparezca el símbolo inferior izquierdo será suficiente con pulsar la tecla.
  - Para que aparezca el símbolo inferior derecho pulsaremos la tecla ‘Alt Gr’ y sin soltar pulsaremos la tecla en cuestión.



- Para que aparezca el símbolo superior pulsaremos la tecla ‘shift’ y sin soltar, pulsaremos la tecla en cuestión.

- Los símbolos de acentuación (acento agudo ´, acento grave `, acento circunflejo ^, diéresis ¨) pulsados antes de vocal, aparecerán acentuando a esa vocal. Para que aparezca el símbolo sin acentuar a ninguna vocal, pulsaremos la combinación de teclas que le corresponda y después pulsamos la barra espaciadora.

A lo largo de esta y las siguientes sesiones deberemos indicar cómo se teclean símbolos y notaciones en las que intervienen las teclas especiales. Con objeto de abreviar utilizaremos los siguientes símbolos para ellas:

Tecla	Símbolo
Tecla de escape	ESCAPE
Tecla de tabulación	TAB
Teclas de cambio o ‘shoft’	SHIFT
Teclas ‘control ‘	CONTROL
Tecla ‘alt’	ALT
Barra de espacio	SPC
Tecla ‘Alt Gr’	ALTGR
Tecla ‘enter’	ENTER

Además, seguiremos las siguiente nomenclatura para describir qué combinaciones de teclas se han de pulsar para insertar cierto símbolo o desencadenar determinada acción:

- Varios símbolos de teclas separados uno del siguiente por el símbolo de suma ‘+’ indican que se debe pulsar la primera tecla y, sin soltar, la segunda y, sin soltar ninguna de las anteriores, la tercera, etc.
- Varios símbolos de teclas uno a continuación de la otra indican que se debe pulsar la primera tecla, soltarla y pulsar la segunda, soltarla y pulsar la tercera, etc.

Así por ejemplo:

- ESCAPEpdESCAPE indica pulsar la tecla de escape, soltar y pulsar p, soltar y pulsar d, soltar y pulsar la tecla de escape.
- CONTROL+SHIFT+7 indica pulsar la tecla ‘control’, sin soltar pulsar la tecla ‘shift’ y sin soltar ninguna de las anteriores pulsar 7.

2. SOBRE MATHEMATICA

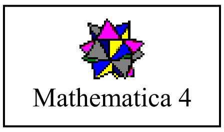
MATHEMATICA es un entorno de trabajo y lenguaje de programación orientado al desarrollo de cálculos matemáticos. Admite tanto la ejecución inmediata de cálculos tal y como lo haría una calculadora así como de programas.

MATHEMATICA implementa en el lenguaje básico y en paquetes de aplicaciones (Packages) la mayor parte de las funciones matemáticas conocidas, comandos para el cálculo diferencial, integral, matricial y numérico así como una amplia gama de primitivas gráficas y potentes herramientas para el cálculo simbólico.

Las funciones básicas y los elementos del lenguaje de programación están directamente incluidas en el núcleo del programa. Las funciones más específicas se pueden encontrar en los packages disponibles que han de ser cargados en cada sesión de trabajo en la que se vayan a utilizar.

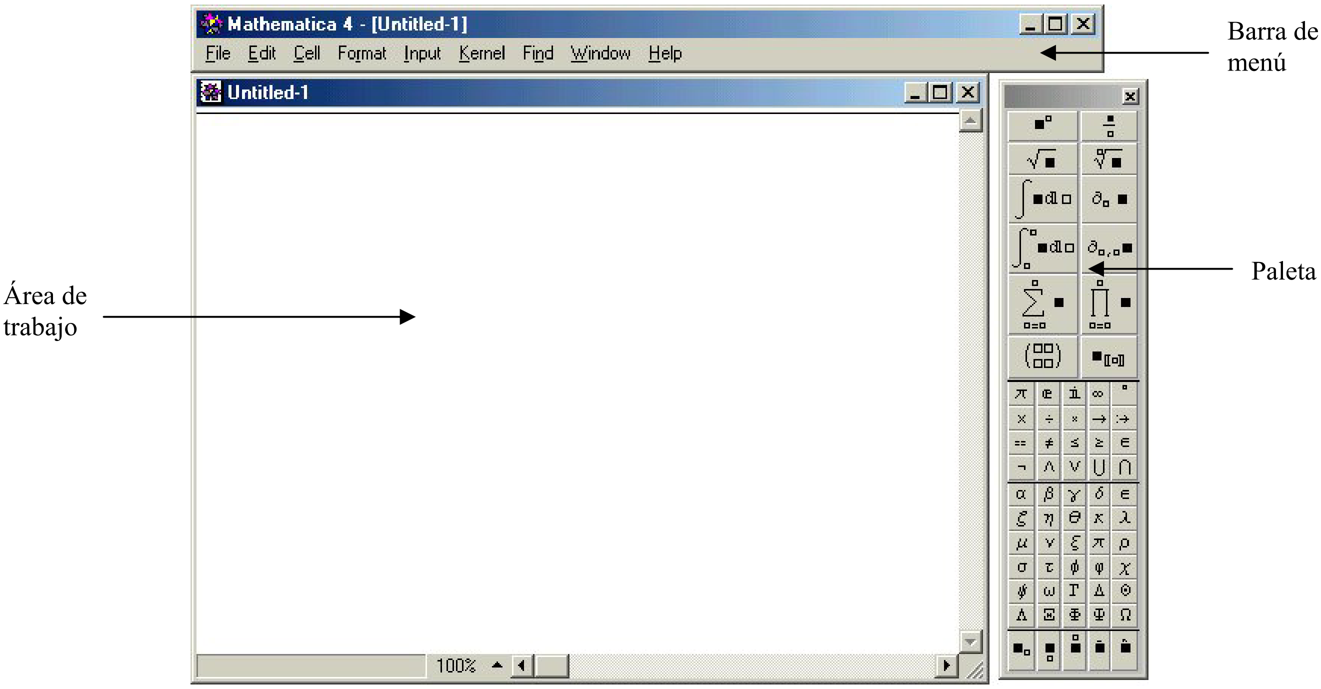
Hay varias versiones del programa de las cuales nosotros utilizaremos la quinta. Esta versión, además de todas las posibilidades de cálculo de las anteriores, permite (al igual que la versión 3 y 4) utilizar una notación muy similar a la notación matemática habitual.

3. METODO DE TRABAJO BÁSICO



Para iniciar una sesión de trabajo con MATHEMATICA basta con pulsar dos veces sobre su icono en el escritorio de windows:

Una vez activado MATHEMATICA aparecerá en pantalla el entorno de trabajo formado por el área de trabajo, la barra de menú y la paleta de símbolos básicos:

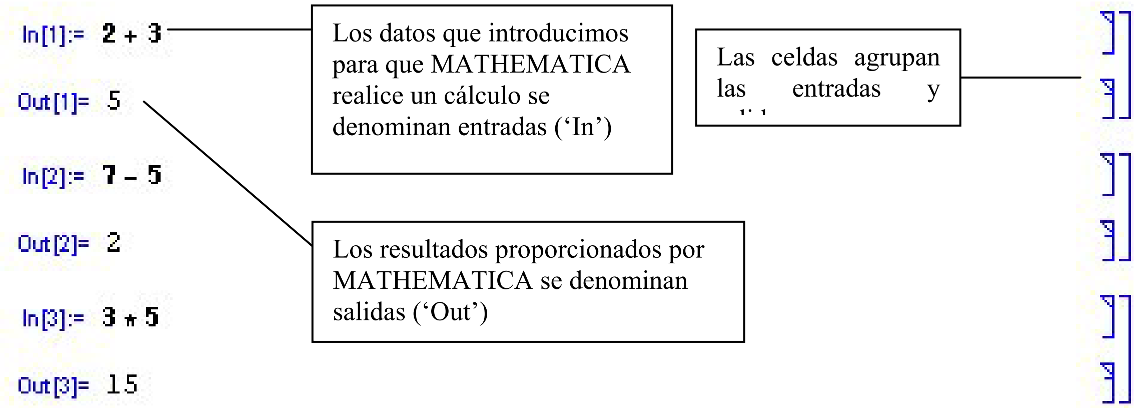


El área de trabajo es la zona principal del programa. En ella se introducen las instrucciones que deseamos que MATHEMATICA ejecute y en ella nos proporciona la respuesta el programa. La paleta nos permite introducir símbolos y notaciones matemáticas especiales. Finalmente, desde la barra de menú podemos grabar el trabajo realizado en disco, cambiar tamaños y tipos de letra y ajustar diversos aspectos del entorno de trabajo de MATHEMATICA. A lo largo de esta sesión y de las siguientes explicaremos diversos detalles de estos tres elementos.

MATHEMATICA espera a que introduzcamos una orden o expresión en el área de trabajo a la que se denomina entrada o 'In'. La manera de evaluar o ejecutar la expresión u orden es pulsar la combinación de teclas: SHIFT+ENTER. Una vez realizado esto, sucede lo siguiente:

a) El programa muestra el resultado de la expresión evaluada. A los datos mostrados se les llama salida o "Out".
b) El programa asignará un mismo número de orden a la entrada y a su salida correspondiente.
c) MATHEMATICA agrupa la información en celdas.

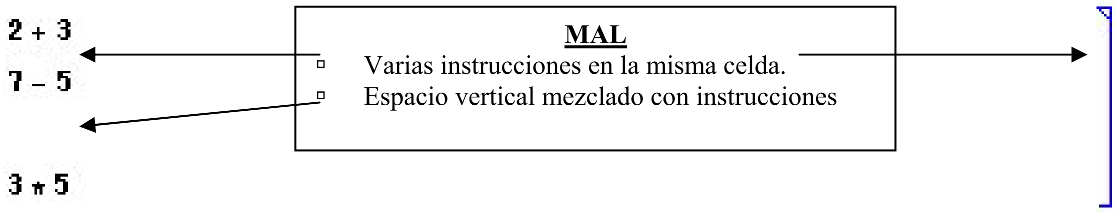
Ejemplo 1:



A medida que ejecutamos cálculos y recibimos resultados, MATHEMATICA va organizando la información de entrada y de salida en diferentes celdas. Para el manejo de celdas deben tenerse en cuenta los siguientes puntos:

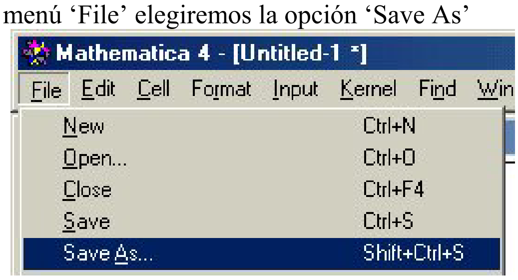
1. Si escribimos una instrucción en el área de trabajo y no la ejecutamos pulsando SHIFT+ENTER, dicha instrucción será ignorada. El ordenador se comportará como si no hubiéramos escrito nada. Sabremos que una instrucción ha sido evaluada cuando aparezca junto a ella el rótulo de entrada correspondiente ('In[...]:=') y la instrucción quede englobada en una celda junto a la respuesta dada por MATHEMATICA.
2. En principio, debe introducirse una sola instrucción en cada celda. Para introducir varias instrucciones véase lo indicado al final de la práctica.
3. La tecla ENTER por si sola tiene como único efecto introducir espacio vertical que es ignorado por MATHEMATICA al realizar los cálculos. Nunca debe utilizarse para separar varias instrucciones en una misma celda.
4. Si para hacer más claro lo escrito es necesario introducir espacio vertical esto debe hacerse en una celda separada en la que no aparezca ninguna instrucción.

Ejemplo 2:

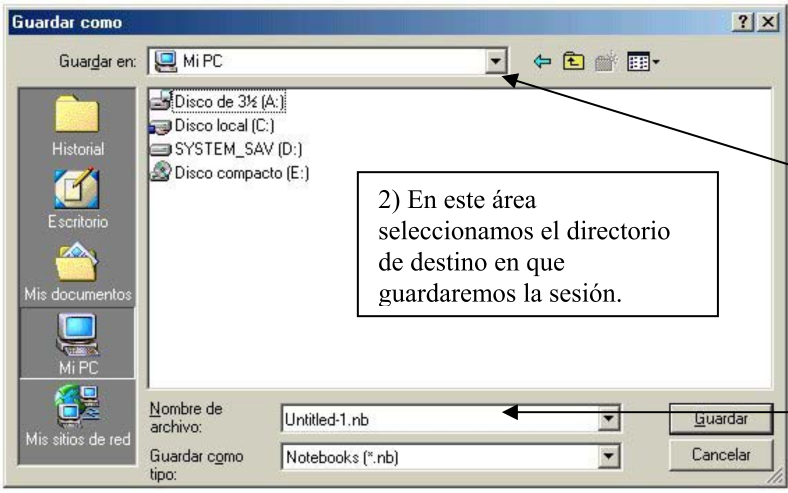


5. Para desplazarse de una celda a otra podemos utilizar el ratón o las teclas con flechas para el desplazamiento del cursor que aparecen en el teclado.
6. Podemos situar el cursor en una celda que ya ha sido evaluada, corregirla y volver a evaluarla. Esto es útil cuando necesitamos repetir el mismo cálculo para datos diferentes.

MATHEMATICA permite guardar en disco los cálculos realizados en una sesión de trabajo. Para ello, en la barra de menú, dentro del menú 'File' elegiremos la opción 'Save As'

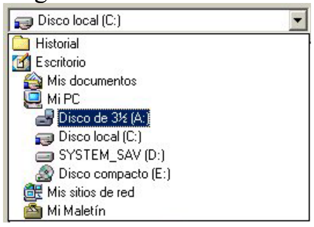


Entonces aparecerá el cuadro de dialogo 'Guardar como' en el que debemos indicar en qué lugar y con qué nombre hemos de guardar la sesión:



2) En este área seleccionamos el directorio de destino en que guardaremos la sesión.

1) Pulsaremos en la flecha al final del recuadro 'Guardar en' para elegir la unidad de destino:



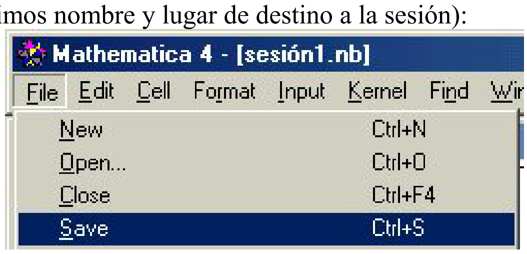
3) En el recuadro 'Nombre de archivo' escribiremos el nombre con el que queremos guardar la

Al guardar por primera vez la sesión le asignaremos un nombre y una ubicación en la que se grabará. Ello quedará reflejado en la barra de menú en la que ahora aparecerá el nombre que hemos elegido para la sesión. Por ejemplo, si asignamos el nombre 'sesión1' tendremos:

Antes de guardar por primera vez

Después de asignar el nombre 'sesión1' mediante 'Save as'

Al guardar la sesión de trabajo como acabamos de explicar, MATHEMATICA almacenará los cálculos realizados hasta el momento en la ubicación que hayamos indicado dentro de un fichero con el mismo nombre que hemos dado a la sesión y con extensión '.nb' (extensión usual de los ficheros de MATHEMATICA). En el caso de nuestro ejemplo el archivo sería 'sesión1.nb'. Después de este punto podemos seguir realizando cálculos y añadiendo nuevos datos e instrucciones a nuestra sesión de trabajo. Los nuevos cálculos y datos no son guardados automáticamente. Para ello en el menú 'File' de la barra de menú elegiremos la opción 'Save'(ya no es preciso utilizar la opción 'Save as' ya que en la primera ocasión que lo hicimos ya dimos nombre y lugar de destino a la sesión):



4. OPERACIONES BASICAS Y MANEJO DE PALETAS

Las operaciones básicas que podemos emplear en MATHEMATICA son las usuales que encontraríamos en cualquier calculadora, o que aparecen generalmente a la hora de efectuar cálculos. Estas son:

suma: El símbolo correspondiente es "+" .
diferencia: El símbolo es "-".
producto: El símbolo es un asterisco "*" o bien un espacio entre los factores.
división: El símbolo es "/".
potenciación: El símbolo es "^".

Todas ellas se emplean de la manera usual

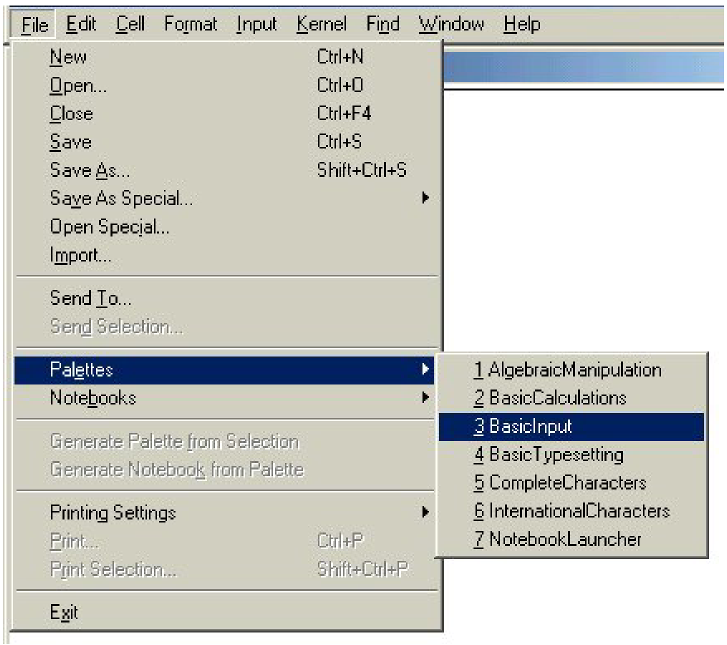
Ejemplo 3:

```
In[1]:= 3+2
Out[1]= 5
In[2]:= 5-3
Out[2]= 2
In[3]:= 5*3
Out[3]= 15
In[4]:= 5 3
Out[4]= 15
In[5]:= 9/3
Out[5]= 3
In[6]:= 5^2
Out[6]= 25
```

Podemos escribir el producto de 5 por 3 tanto utilizando el símbolo de asterisco (*) como dejando un espacio entre el 5 y el 3.

Si bien esta notación para las operaciones básicas (sobre todo para la división y potenciación) es habitual en todos los lenguajes de programación, no se asemeja a la que habitualmente empleamos cuando anotamos estas operaciones con lápiz y papel. Así por ejemplo, lo más usual sería escribir 5^2 en lugar de `5^2` o $\frac{9}{3}$ en lugar de `9/3`. Por este motivo MATHEMATICA incorpora métodos de escritura que permiten imitar la notación matemática clásica. Disponemos en realidad de tres formas diferentes de escribir la gran mayoría de expresiones matemáticas:

- **Método 1:** Utilizando el lenguaje interno de MATHEMATICA. Todas las expresiones se puede escribir utilizando los comandos o instrucciones del programa.
- **Método 2:** Utilizando la paleta de símbolos. En la parte derecha de la pantalla aparece la paleta de símbolos y notaciones especiales que permite introducir, con una notación similar a la clásica, potencias, quebrados, raíces, integrales, derivadas, letras del alfabeto griego y otros muchos símbolos especiales. En ocasiones, la paleta de símbolos no se halla presente. Para conseguir que aparezca de nuevo es suficiente con seleccionar en la barra de menú la opción 'File' y dentro de ella 'Palettes\Basic Input'.
- **Método 3:** Utilizando métodos abreviados de teclado para introducir las notaciones y símbolos especiales de la paleta.



En muchos casos indicaremos cómo realizar la escritura siguiendo cada uno de estos tres métodos. Comenzamos haciéndolo para las operaciones de división y potenciación (para suma, resta y producto no hay diferencia entre los tres métodos):

Operación	Lenguaje interno	Paleta	Paleta-Método de teclado
División	/		CTRL+SHIFT+7
Potenciación	^		▪ escribir la base ▪ CTRL+6 ▪ escribir el exponente

Ejemplo 4:

Por ejemplo, para escribir el quebrado $\frac{10}{3}$ procedemos como sigue:

- Pulsamos el botón correspondiente de la paleta ($\frac{\blacksquare}{\square}$) o bien tecleamos CTRL+SHIFT+7. Tras hacerlo aparecerá en pantalla el símbolo de quebrado

$$\frac{\boxdot}{\square}$$

- Hemos de completar los recuadros del símbolo de quebrado que ha aparecido situando un 10 en el recuadro superior y un 3 en el inferior. El cuadradito negro que aparece dentro del recuadro superior nos indica donde aparecerá lo próximo que escribamos.
- Tecleamos 10 en el recuadro superior para obtener

$$\frac{10}{\square}$$

- Para pasar al recuadro inferior podemos pulsar la tecla de tabulación TAB o bién señalar con el ratón el recuadro inferior y pulsar luego el botón izquierdo. El recuadrito negro aparecerá ahora en el recuadro inferior:

$$\frac{10}{\boxdot}$$

Ello indica que ahora podemos teclear 3 para obtener finalmente el quebrado deseado.

4.1. Prioridad de operaciones

Pueden darse casos en los que la notación sea confusa y no sepamos en qué orden se efectuarán las operaciones. Así por ejemplo la expresión:

$$3+1\cdot 5$$

podría entenderse como $3+(1\cdot 5)$= 8 o como $(3+1)\cdot 5$= 20. Para dar solución a este tipo de conflictos existe lo que se llama prioridad de operaciones. Las operaciones que hemos visto están ordenadas según un orden de prioridades de mayor a menor prioridad de manera que cuando se analiza un cálculo se realizan primero las operaciones que tienen mayor prioridad. El orden de prioridades de mayor a menor es el siguiente:

- mayor prioridad: potenciación ^.
- prioridad intermedia: producto y división (* y /).
- prioridad baja: suma y diferencia (+ y -).

Cuando dos operaciones tienen la misma prioridad (por ejemplo + y - o también * y /) se efectúan de derecha a izquierda.

Así pues en la expresión $3+1\cdot 5$ se efectuará primero la operación de mayor prioridad, el producto, y luego la de menor +, con lo cual el resultado será $3+(1\cdot 5)$ = 8 y no 20.

El orden de ejecución de las operaciones se puede alterar mediante el uso de los paréntesis “(“ y “)” en la manera habitual.

5. TIPOS DE DATOS

Hemos examinado las operaciones elementales que podemos utilizar con MATHEMATICA y nos interesa ahora saber qué tipos de datos podemos combinar con tales operaciones. Estos tipos son:

1) **Enteros:** Números enteros: -5,-4,..,0,1,..3,....
2) **Racionales:** Números que se obtienen como el cociente de dos enteros: $\frac{1}{3},\frac{-2}{7},\frac{1}{1000},...$
3) **Reales exactos:** Los números irracionales como el número e ó el número π tienen infinitas cifras decimales no periódicas. Debido a ello, es imposible escribir su expresión decimal con total

exactitud y la única manera de referirnos a ellos con precisión absoluta es mediante sus símbolos 'e' y ' π '. Podemos usarlos en MATHEMATICA de la siguiente manera:

Número	Lenguaje interno	Paleta	Paleta-Método de teclado
e	`E`	ⅇ	ESCeeESC
π	`Pi`	π	ESCpiESC

4) **Números en coma flotante o reales aproximados:** Número en coma flotante es el concepto informático que corresponde a número con decimales o aproximado. Este tipo de dato engloba por tanto los números con decimales o aproximados con una mayor o menor precisión. El símbolo para la coma decimal es el signo de puntuación "`.`". Por ejemplo podremos poner: `3.4`, `0.56456`, etc.
5) **Complejos:** Un número complejo es un número de la forma a+bi, donde i es la unidad imaginaria definida por la ecuación $i^2=-1$. La unidad imaginaria "i" se escribe en MATHEMATICA como sigue:

Lenguaje interno	Paleta	Paleta-Método de teclado
`I`	ⅈ	ESCiiESC

6) **Símbolos:** Un símbolo es cualquier combinación de letras y números que comienza por una letra minúscula. Por ejemplo: `x`, `radio`, `x1`, etc.

Téngase en cuenta que siempre que añadimos el símbolo de puntuación "`.`" al final de un número el ordenador lo considerará como aproximado, así por ejemplo: `5` es un número exacto mientras que `5.` es un número aproximado.

Ejemplo 5*:*

```
In[1]:= (3+2ⅈ)(2-ⅈ)
Out[1]= 8+ⅈ
In[2]:= ⅈ^2
Out[2]= -1
In[3]:= 3.145*1.5
Out[3]= 4.7175
In[4]:= 3x+7x
Out[4]= 10x
```

Podemos realizar operaciones entre números complejos, aproximados o entre símbolos.

6. TIPOS DE DATOS Y PRECISION

MATHEMATICA considera como exactos a los tipos de datos siguientes:

- Enteros
- Racionales
- Reales exactos
- Símbolos que no tengan asignado un valor
- Complejos con coeficientes exactos (es decir, un complejo `a+bⅈ` en el que, a su vez, a y b son los dos exactos).

A la hora de evaluar una expresión el programa sigue las siguientes reglas:

1) Si todos los datos que intervienen en la expresión son exactos el resultado ofrecido tiene que ser exacto o de lo contrario no se evalúa.
2) Si alguno de los datos que intervienen en la expresión es aproximado el resultado se evalúa y se devuelve como aproximado.

***Ejemplo* 6:**

In[1]:= $\frac{2}{3}$

Out[1]= $\frac{2}{3}$

In[2]:= $\sqrt{2}$

Out[2]= $\sqrt{2}$

In[4]:= e^3

Out[4]= e^3

En todos estos cálculos aparecen 2, 3, e que son datos exactos. Si los datos de entrada son exactos el ordenador intentará que los resultados proporcionados también lo sean. Sin embargo, $\frac{2}{3}$, $\sqrt{2}$ ó e^3 son operaciones que dan como resultado números con infinitas cifras decimales y por tanto es imposible escribir con total exactitud su expresión decimal. Puesto que no es posible dar un resultado exacto el ordenador opta por dejar los cálculos sin evaluar.

In[5]:= $\frac{2.}{3}$

Out[5]= 0.66666

In[6]:= $\sqrt{2.}$

Out[6]= 1.41421

`2.` es un dato aproximado (el signo de puntuación indica que es un número aproximado independientemente de que tenga decimales o no). Si en los datos de entrada hay números aproximados, MATHEMATICA ya no intentará ofrecer como resultado un número exacto sino aproximado. En este caso los cálculos sí son evaluados.

Se puede forzar a que el programa evalúe una expresión que es exacta proporcionándonos un dato aproximado. Esto se consigue por medio del comando `N`.

Comando: Comando N para aproximación de números exactos

Sintaxis:

Formato 1: `N[a]`
Formato 2: `N[a,n]`

Resultado:

Formato 1: Da una aproximación de `a` con 16 dígitos de precisión.

Formato 2: Da una aproximación de `a` con `n` dígitos de precisión.

Téngase en cuenta que n dígitos de precisión no es lo mismo que n números decimales, por ejemplo:

3123	es un número con 4 dígitos de precisión.
3123.4	es un número con 5 dígitos de precisión y un decimal.
3123.000000	es un número con 10 dígitos de precisión y 6 dígitos decimales.
3.1415	es un número con 5 dígitos de precisión y 4 dígitos decimales.

***Ejemplo* 7:**

In[1]:= **N[π]**
Out[1]= 3.14159
In[2]:= **N[π,30]**
Out[2]= 3.14159265358979323846264338328

MATEMÁTICA realiza todos los cálculos con una precisión mínima de 16 dígitos. Sin embargo, para evitar que la pantalla se llene de información, solo mostrará las 5 primeras cifras decimales (véase el resultado que ofrece **N[π]**). Para pedir que se muestren más cifras decimales debemos indicarlo al utilizar el comando `N` (como en **N[π,30]**).

Posteriormente veremos que podemos aplicar el comando `N` no solamente a número sino también a expresiones de cualquier tipo.

7. FUNCIONES

En matemáticas es común el concepto de función y, así pues, estamos acostumbrados a emplear expresiones del tipo: sen (3), tan (π), arccos (1/2), etc. En todas estas expresiones aparece:

- **Una función**. En los ejemplos anteriores las funciones empleadas son: sen, tan, arccos
- **Un argumento** al que se aplica la función. En los ejemplos los argumentos a los que se les han aplicado las distintas funciones han sido 3, π y 1/2. Obsérvese que en la nomenclatura matemática los argumentos a los que se les aplica una función aparecen siempre entre paréntesis.

MATHEMATICA incorpora la práctica totalidad de funciones matemáticas usuales. A continuación enumeramos algunas de ellas:

- `Sin` = seno de un ángulo en radianes.
- `Cos` = coseno de un ángulo en radianes.
- `Tan` = tangente de un ángulo en radianes.
- `Sqrt` = raíz cuadrada de un número real o complejo.
- `ArcTan` = arco cuya tangente.
- `ArcSin` = arco cuyo seno.
- `ArcCos` = arco cuyo coseno.
- `Log` = logaritmo natural.

Cada una de ellas se emplea de la misma manera que lo haríamos en el lenguaje matemático habitual con la salvedad de que en MATHEMATICA los argumentos de una función han de ir encerrados entre corchetes y no entre paréntesis, es decir:

- Para calcular el seno de 3 pondremos `Sin[3]` en lugar de sen (3).
- Para calcular el arco cuya tangente es 1 pondremos `ArcTan[1]` en lugar de arctan(1).
- Para calcular la raíz cuadrada de 2 pondremos `Sqrt[2]`.

Para la escritura de la raíz cuadradas disponemos además de los tres mecanismos alternativos habituales:

Lenguaje interno	Paleta	Paleta-Metodo teclado
`Sqrt`	$\sqrt{\blacksquare}$	CTRL+2

Ejemplo 8:

```
In[1]:= Sin[π]
Out[1]= 0
In[2]:= ArcTan[1]
Out[2]= π/4
In[3]:= Cos[1]
Out[3]= Cos[1]
In[4]:= N[Cos[1]]
Out[4]= 0.540302
```

En la expresión Cos[1] solamente intervienen datos exactos (en este caso el único dato es el número 1 que es un dato exacto) y el ordenador intentará dar un resultado exacto. Sin embargo cos(1) es un número irracional con infinitas cifras decimales así que es imposible escribir de forma exacta su valor. Por tanto el ordenador deja la expresión sin evaluar

8. VARIABLES, SÍMBOLOS Y ASIGNACIÓN DE VALORES

En diversas disciplinas científicas es habitual el uso de variables a las que se puede luego asignar un valor. Así por ejemplo, para calcular el área de un triángulo utilizamos la fórmula

$$\text{area} = \frac{\text{base} \cdot \text{altura}}{2}$$

en la que aparecen las variables 'area', 'base' y 'altura'. Cuando estemos ante un triángulo concreto, asignaremos a las variable base y altura el valor que le correspondan y podremos calcular luego el área aplicando la fórmula. Por ejemplo:

$$\begin{array}{l} \text{base} = 4 \\ \text{altura} = 5 \end{array} \Rightarrow \text{area} = \frac{4 \cdot 5}{2} = 10 .$$

Al estudiar los tipos de datos vimos que con MATHEMATICA también podemos manejar variables o símbolos. Un símbolo o variable es cualquier agrupación de caracteres y números que comienza con una letra minúscula. Ejemplos de posibles símbolos son: `x`, `z`, `hola`, `z1`, `hola318x`, `altura`, etc.
Ha de tenerse en cuenta que:

- No se puede insertar un espacio en blanco en el nombre de un símbolo (un espacio en blanco es considerado por el programa como una multiplicación).
- MATHEMATICA distingue entre mayúsculas y minúsculas.

Ejemplo 9:

`X1`	Símbolo correcto.
`3a`	Símbolo incorrecto ya que no comienza por una letra. `3a` se interpretará como el producto de `3` por `a`.
`x 1`	Incorrecto puesto que incluye un espacio en blanco. El programa lo interpretará como el producto de `x` por `1`.
`solucionecuacion`	símbolo correcto.
`solucion ecuacion`	Símbolo incorrecto por incluir espacio en blanco. Será interpretado como el producto del símbolo `solucion` por el símbolo `ecuación`.
`primervalor ≠ PrimerValor`	Vemos un ejemplo de dos símbolos en los que intervienen las mismas letras pero que son distintos por no coincidir mayúsculas y minúsculas.

A un símbolo se le puede asignar un valor que puede ser un número, lista, otro símbolo y en general cualquier cosa. Tenemos pues, que una variable o símbolo puede encontrarse en dos estados:

a) Sin tener un valor asignado: En esta situación la variable es considerada como un dato de tipo exacto y nunca se evalúa.
b) Teniendo asignado un valor: Entonces en cada expresión o comando en el que aparezca tal símbolo, este es inmediatamente sustituido por su valor, antes incluso de que la expresión sea evaluada.

Para asignar un valor a una variable empleamos el símbolo de asignación '=' en la siguiente forma:

Comando: Operador de asignación inmediata `=`

Sintaxis:

```
nombresimbolo = expresionvalor
```

Resultado: Evalúa la expresión `expresionvalor` y la asigna al símbolo `nombresimbolo`, de manera que cada vez que en adelante aparece el símbolo `nombresimbolo` este es sustituido por `expresionvalor`. Tras ejecutar la asignación se produce como resultado el valor `expresiovalor`.

Ejemplo 10:

A partir de esta orden, siempre que en cualquier expresión aparezca el símbolo "`a`" este será sustituido por 5

```
In[1]:= a=5
Out[1]= 5
```

In[2]:= a^2+1

Out[2]= 26

In[3]:= $b=\sqrt{2}+1$

Out[3]= $1+\sqrt{2}$

In[4]:= $\frac{a}{b}$

Out[4]= $\frac{5}{1+\sqrt{2}}$

In[5]:= $p=x^3+6x^2+11x+6$

Out[5]= $6+11x+6x^2+x^3$

In[6]:= $\frac{p}{x+3}$

Out[6]= $\frac{6+11x+6x^2+x^3}{3+x}$

In[7]:= x=3

Out[7]= 3

In[8]:= p

Out[8]= 120

In[9]:= x=-4

Out[9]= -4

In[10]:= p

Out[10]= -6

Una vez que hemos asignado a al variable `a` el valor 5, cada vez que el programa vuelve a encontrar el símbolo `a` lo sustituirá por 5.

En una variable podemos almacenar expresiones en las que intervienen otras variables. En este caso en la variable `p` hemos almacenado un polinomio en el que interviene otra variable `x`.

Si modificamos el valor de la variable `x` también variará el valor de `p`.

Podemos modificar el valor asignado a una variable realizando una nueva asignación. El nuevo valor quedará reflejado en las futuras operaciones. Véase que ahora, en el polinomio `p`, `x` es sustituida por –4 dando lugar a un resultado diferente.

Tal y como se ve en el ejemplo anterior, podemos manejar expresiones en las que aparecen variables o símbolos. Para estos casos MATHEMATICA proporciona instrucciones que permiten simplificar o efectuar calculos.

Comando: `Simplify`

Sintaxis:

`Simplify[expresión]`

Resultado: Simplifica, si ello es posible, la expresión encerrada entre los corchetes.

Comando: `Expand`

Sintaxis:

`Expand[expresión]`

Resultado: Desarrolla los cálculos que aparecen pendientes de evaluación en la expresión encerrada entre corchetes.

Ejemplo 11:

In[1]:= $p=y^3+6y^2+11y+6$

Out[1]= $6+11y+6y^2+y^3$

In[2]:= $\frac{p}{y+3}$

Out[2]= $\frac{6+11y+6y^2+y^3}{3+y}$

In[3]:= $\text{Simplify}[\frac{p}{y+3}]$

Out[3]= $2+3y+y^2$

```
In[4]:= Expand[(y+z)^2]
Out[4]= y^2+2yz+z^2
In[5]:= Expand[(y+z)^3]
Out[5]= y^3+3y^2z+3yz^2+z^3
In[6]:= Simplify[%]
Out[6]= (y+z)^3
```

8.1. Eliminación de valores asignados a un símbolo.

La supresión de valores asignados a un símbolo se puede realizar por medio de los comandos "`=.`" y `Remove`. Veamos cada uno de ellos:

Comando: Comando `=.` para eliminación de valores de variables
Sintaxis:

`simb=.`

Resultado: Elimina el valor asignado a la variable `simb` de mandera que dicha variable queda sin tener ningún valor asignado.

Comando: `Remove`
Sintaxis:

1) `Remove[simb1,simb2,...,simbn]`
2) `Remove["nomgen1",... "nomgenn"]`

Resultado: **1)** Borra completamente todos los símbolos enumerados `simb1`, `simb2`,..., `simbn` y los valores que tengan asignados.
2) Borra completamente todos los símbolos determinados por cada una de los nombres genéricos `muestra1`,..., `muestran`.

En el recuadros anterior hemos entendido por nombre genérico a cualquier expresión genérica en la que aparece el símbolo `*`. Recuérdese que en MATHEMATICA el símbolo `*` se emplea de modo parecido al MSDOS y así por ejemplo "`a*`" hace referencia a todo aquello que comienza por una "`a`".

Ejemplo 12:

```
In[1]:= a1=1;a2=2;a3=3;b=2;c=b;
In[2]:= a3
Out[2]= 3
In[3]:= a3=.
In[4]:= a3
Out[4]= a3
In[5]:= a1
Out[5]= 1
In[6]:= a2
Out[6]= 2
In[7]:= Remove[a1,b]
In[8]:= b
Out[8]= b
In[9]:= Remove["a*"]
In[10]:= a2
Out[10]= a2
```

`a3` valía `3` pero tras esta instrucción pierde su valor.

Intentamos con esta orden eliminar el valor de todos los símbolos que comiencen por `a`.

En el caso de que deseemos que la operación de borrado afecte a todas la variables que hallamos definido hasta el momento usaremos la instrucción `Remove["Global`@*"]` (el símbolo ` de acento grave se encuentra en la misma tecla que ^ contigua a la de la p) ya que el designador genérico `Global`@*` hace referencia a todas las variables definidas por el usuario hasta el momento.

8.2. Ejercicios

1 Indicar un comando mediante el cual se produzca el borrado de todos los símbolos terminados en "`w3`".

2 Indicar un comando para el borrado de todos los símbolos que comiencen por "`z`".

9. LISTAS

En la sesión anterior hemos estudiado distintos tipos de datos básicos o elementales que podemos manejar con MATHEMATICA. Existe sin embargo otro tipo de dato denominado 'lista' que permiten combinar los tipos básico que ya conocemos.

<u>DEFINICIÓN 1:</u> Una lista es un conjunto de datos separados por comas y encerrados entre llaves. A los datos que forman una lista los llamamos elementos de la lista.

Ejemplo 13:

1. `{1.21,2,2+i,x`^{2}`+1}` es una lista formada por los elementos `1.21`, `2`, `2+i` y `x`^{2}`+1`. Véase que todos ellos son de distinto tipo (real, entero, complejo y fórmula simbólica).
2. `{1,1,1,1}` es una lista con cuatro elementos todos iguales a `1`.
3. `{}` es una lista sin ningún elemento (lista vacía).

En una lista se suponen que los datos están ordenados, así, debemos tener en cuenta que:

- En una lista pueden existir elementos repetidos. Por ejemplo las listas `{1,1}` y `{1,1,1,1}` son distintas.
- En una lista los elementos están ordenados. Por ello `{1,2}` y `{2,1}` son listas diferentes.
- Una lista puede ser un elemento de otra lista. De este modo en la lista

  `{1,2,{3,{2,1}},1,3}`

  tenemos que el segundo elemento es `{3,{2,1}}` que es a su vez otra lista que tiene por elementos a `3` y a `{2,1}` que nuevamente es una lista. Decimos en este caso que `{3,{2,1}}` es una sublista de la lista principal y que `{2,1}` es una subsublista de la lista principal.

Se puede acceder a los elementos de una lista mediante el uso de dobles corchetes, `[[]]`, como en los siguientes ejemplos:

Ejemplo 14:

```
In[1]:= {a,b,c}[[2]]
Out[1]= b
In[2]:= {1,2,{a,b,c}}[[3]]
Out[2]= {a,b,c}
In[3]:= a={3,4.5,{1, π},6,2};
In[4]:= a[[2]]
Out[4]= 4.5
In[5]:= a[[3]]
Out[5]= {1, π}
In[6]:= a[[3]][[2]]
Out[6]:= π
In[7]:= a[[3,2]]
In[6]:= π
```

Véase que:
```
a[[3]][[2]]=
={3,4.5,{1, π},6,2}[[3]][[2]]
={1, π}[[2]]
=π
```

En lugar de `a[[3]][[2]]` podemos escribir para abreviar `a[[3,2]]`.

Como vemos en los ejemplos anteriores, cuando un lista tiene sublistas, subsublistas, etc., es posible obtener cualquiera de los elementos que forman estas sublistas indicando adecuadamente su posición. De manera general tenemos que

`a[[i1]][[i2]]...[[ik]]` ó `a[[i1,i2,...,ik]]`

designa al `ik`-ésimo elemento del (`ik`-1)-ésimo elemento ... del `i2`-ésimo elemento del `i1`-ésimo elemento de la lista `a`.

Ejemplo 15:

In[1]:= **a={0,1,{2,3.5,{6,-9,** $\frac{1}{2}$**,{ⅇ,** $\sqrt{2}$ **},8},Cos[10],-1},11,12};**

In[2]:= **a[[3]]**

Out[2]= {2,3.5,{6,-9, $\frac{1}{2}$,{ⅇ, $\sqrt{2}$},8},Cos[10],-1}

In[3]:= **a[[3,3]]**

Out[3]= {6,-9,1/2,{ⅇ, $\sqrt{2}$},8}

In[4]:= **a[[3,3,4]]**

Out[4]= {ⅇ, $\sqrt{2}$}

In[5]:= **a[[3,3,4,1]]**

Out[5]= ⅇ

In[6]:= **{1,{2,1}}[[1,2]];**

Part::partw: Part specification {1, {2,1}}[[1,2]] is longer than depth of object.

Out[6]= {1,{2,1}}[[1,2]]

Puede verse que ⅇ es el primer elemento del cuarto elemento del tercer elemento del tercer elemento

Véase que si especificamos una posición inexistente obtenemos un mensaje de error.

Ejercicios

Almacénese en una variable la lista siguiente y obténgase mediante [[]] los elementos que aparecen en negrita:

{a,**b**,c,{1,**a**},x+2,{1+x,{**1-x**,a,b,{**-4**,{-5,**6**,8},**-1**,{2,**3**},4,c}},**q**},-12,**E**}

Podemos generar una lista de forma automática mediante el uso del comando `Table`. Así por ejemplo si queremos obtener la lista formada por los 10 primeros números impares tendremos:

In[1]:= **Table[2n-1,{n,1,10}]**

Out[1]= {1,3,5,7,9,11,13,15,17,19}

La sintaxis más general del comando Table es la siguiente:

- Table[a,{n}] produce una lista en la que aparece el elemento a repetido n veces.
- Table[a_n,{n,n_i,n_f}] produce un lista formada por los elementos a_n donde n varia desde el valor inicial n_i hasta el valor final n_f.
- Table[a_n,{n,n_i,n_f,s}] produce un lista formada por los elementos a_n donde n varia desde el valor inicial n_i hasta el valor final n_f dando saltos de longitud s.

Ejemplo 16:

In[1]:= **Table[n,{5}]**

Out[1]= {n,n,n,n,n}

In[2]:= **Table[n,{n,3,7}]**

Out[2]= {3,4,5,6,7}

In[3]:= **Table[n,{n,4,10,2}]**

Out[3]= {4,6,8,10}

Ejercicios

1. Obtener una lista formada por los 100 primeros número pares.
2. Obtener la lista de los cosenos de los 50 primeros números naturales. Calcular el valor aproximado de cada uno con 10 dígitos de precisión.
3. Repetir el ejercicio anterior pero construyendo la liste de manera que a cada coseno le acompañe el número natural correspondiente (es decir, algo del tipo {{1,Cos[1]}, {2,Cos[2]}, {3,Cos[3]},...}).

10. MATRICES Y VECTORES DE $\mathbb{R}^n$

Como ya hemos visto en distintas ocasiones, muchos comandos y símbolos especiales pueden escribirse con MATHEMATICA de tres formas distintas (lenguaje interno, paleta, método abreviado del teclado).

Veamos a continuación los tres métodos para manejar matrices y vectores con MATHEMATICA. En este caso prestaremos especial atención a la forma en la que se manejan matrices y vectores en el lenguaje interno del programa (es decir al primero de los tres métodos):

1. Representación de matrices y vectores en MATHEMATICA (lenguaje interno)

En MATHEMATICA tanto elementos de $\mathbb{R}^n$ como matrices se representan mediante listas de la siguiente manera:

- El elemento $(a_1, a_2, \ldots, a_n) \in \mathbb{R}^n$ se representa mediante la lista
  $\{a_1,\ a_2, \ldots, a_n\}$
  es decir, se representa mediante la lista formada por todos sus elementos.
- La matriz

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

  se representa mediante la lista formada por todas sus filas expresadas estas a su vez en forma de lista, es decir, en la forma:
  $\{\{a_{11}, a_{12}, \ldots, a_{1n}\}, \{a_{21}, a_{22}, \ldots, a_{2n}\}, \ldots, \{a_{m1}, a_{m2}, \ldots, a_{mn}\}\}$.

Ejemplo 17:

$(-7,2,0) \in \mathbb{R}^3$	se representa como	`{-7,2,0}`.
$(4,2,1,-11) \in \mathbb{R}^4$	se representa como	`{4,2,1,-11}`.
$\begin{pmatrix} -1 & 0 & 3 \\ 2 & 7 & 1 \end{pmatrix}$	se representa como	`{{-1,0,3},{2,7,1}}`.
$\begin{pmatrix} 3 & 1 \\ 6 & 2 \\ 0 & 12 \\ 4 & -7 \end{pmatrix}$	se representa como	`{{3,1},{6,2},{0,12},{4,-7}}`.
La matriz fila $(-7\ \ 2\ \ 0)$	se representa como	`{-7,2,0}`.
La matriz columna $\begin{pmatrix} -7 \\ 2 \\ 0 \end{pmatrix}$	se representa como	`{{-7},{2},{0}}`.

Véase que un elemento de $\mathbb{R}^n$ y una matriz fila se representa de la misma manera.

El programa MATHEMATICA escribe por defecto las matrices en forma de lista. Mediante el uso de la instrucción `MatrixForm` podemos conseguir que una matriz expresada en forma de lista se muestre en pantalla de forma similar a como se representaría en la notación matemática habitual.

Ejemplo 18:

```
In[1]:= {{3,-3,0},{2,1,6}}
Out[1]:= {{3,-3,0},{2,1,6}}
In[2]:= MatrixForm[{{3,-3,0},{2,1,6}}]
Out[2]//MatrixForm=
```

$$\begin{pmatrix} 3 & -3 & 0 \\ 2 & 1 & 5 \end{pmatrix}$$

```
In[3]:= MatrixForm[{{2,-1},{3,6},{5,1}}]
Out[3]//MatrixForm=
```

$$\begin{pmatrix} 2 & -1 \\ 3 & 6 \\ 5 & 1 \end{pmatrix}$$

MATHEMATICA, si no se le indica lo contrario, escribe siempre las matrices en forma de listas.

Las matrices

$$\begin{pmatrix} 3 & -3 & 0 \\ 2 & 1 & 6 \end{pmatrix} \text{ y } \begin{pmatrix} 2 & -1 \\ 3 & 6 \\ 5 & 1 \end{pmatrix}$$

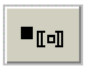
se representan en MATHE-MATICA como:

- `{{3,-3,0},{2,1,6}}`.
- `{{2,-1},{3,6},{5,1}}`.

Mediante el uso de la instrucción `MatrixForm` podemos obtener una representación parecida a la habitual.

NOTA:

1. Tal y como vimos en la sección anterior es posible obtener un elemento de una matriz mediante el uso de `[[]]`. Por tanto el elemento (i,j) de la matriz `a` se obtendrá mediante `a[[i,j]]`. Como siempre, existen tres formas de escribir los dobles corchetes:

Lenguaje interno	Paleta	Metodo abreviado teclado
`[[]]`		para `[[`: ESC `[[` ESC para `]]`: ESC `]]` ESC para el subíndice: CTRL+ –

2. Podemos obligar al programa a que siempre muestre en pantalla las matrices en formato matemático. Ello lo haremos mediante la instrucción `$Post=MatrixForm`. A partir del momento en que ejecutamos esta instrucción todas las matrices aparecerán en formato matemático. Para deshabilitar esta opción utilizaremos el comando `$Post=.`.

Ejemplo 19:

```
In[1]:= a={{a11,a12,a13},{a21,a22,a23}};
In[2]:= a[[2,3]]
Out[2]= a23
In[3]:= a
Out[3]= {{a11,a12,a13},{a21,a22,a23}}
In[4]:= $Post=MatrixForm
Out[4]\\MatrixForm= MatrixForm
In[5]:= a
Out[4]\\MatrixForm=
```

$$\begin{pmatrix} a11 & a12 & a13 \\ a21 & a22 & a23 \end{pmatrix}$$

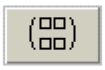
```
In[6]:= $Post=.
In[6]:= a
Out[6]= {{a11,a12,a13},{a21,a22,a23}}
```

Si activamos la opción `$Post=MatrixForm` las matrices aparecerán en pantalla en formato matemático. Por contran, antes de activar la opción y después de desactivarla se muestran como listas.

Debe tenerse en cuenta que al activar la opción `MatrixForm` de modo permanente el programa intentará ofrecer todas las salidas en forma de matriz lo cual puede, en algunas ocasiones, producir resultados inesperados.

2. Representación de matrices y vectores en MATHEMATICA (paleta)

Podemos insertar una matriz con la notación matemática habitual empleando la paleta. Para ello pulsamos el botón:



Tras pulsar el botón de matriz completaremos en los recuadros insertando los distintos elementos de la matriz, cada uno en su posición. La paleta permite solamente insertar matrices 2×2. Si pretendemos insertar con formato matemático una matriz con tamaño diferente habremos de emplear el método abreviado de teclado que se explica a continuación.

3. Representación de matrices y vectores en MATHEMATICA (paleta-método abreviado de teclado)

Para teclear una matriz con el método abreviado de teclado seguimos el siguiente proceso:

1. Escribimos los paréntesis que delimitan la matriz.
2. Situado el cursor entre los paréntesis teclearemos:

CTRL+ENTER	Para añadir una nueva fila a la matriz.
CTRL+, (CTRL +"coma")	Para añadir una nueva columna a la matriz.

3. Completamos los matriz rellenando los espacios correspondientes a cada uno de sus elementos. Podemos recorrer directamente los espacios a rellenar de la matriz pulsando la tecla del tabulador TAB.

Ejemplo 20:

Para escribir la matriz $\begin{pmatrix} 1 & 2 & 0 \\ -1 & 2 & 1 \end{pmatrix}$ seguimos los siguientes pasos:

Paso 1	Paso 2	Paso 3
Introducimos los paréntesis.	Situamos el cursos entre ambos paréntesis y tecleamos: ▪ 1 vez CTRL+ENTER (2 filas) ▪ 2 veces CTRL+, (3 columnas)	Insertamos cada elemento de la matriz en el lugar que le corresponde.
( )	$\begin{pmatrix} \square & \square & \square \\ \square & \square & \square \end{pmatrix}$	$\begin{pmatrix} 1 & 2 & 0 \\ -1 & 2 & 1 \end{pmatrix}$

10.1. Operaciones entre matrices

Podemos efectuar con MATHEMATICA las operaciones que conocemos para matrices. Así tenemos:

Operación	Símbolo con que se efectúa en MATHEMATICA
Suma de dos matrices.	+ (símbolo de suma)
Diferencia de dos matrices.	- (símbolo de diferencia)
Producto de una matriz por un número real.	• * (asterisco=símbolo de producto) • Por yuxtaposición (es decir, situando el número junto a la matriz).
Producto de dos matrices.	. (signo de puntuación)

Ejemplo 21:

In[1]:= $3*\begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & -1 \end{pmatrix}$

Out[1]//MatrixForm=

$$\begin{pmatrix} 9 & 6 & 3 \\ 6 & 3 & -3 \end{pmatrix}$$

In[2]:= $\begin{pmatrix} 1 & 2 \\ 4 & 7 \end{pmatrix}+\begin{pmatrix} 1 & 2 \\ -13 & 0 \end{pmatrix}$

Out[2]//MatrixForm=

$$\begin{pmatrix} 2 & 4 \\ -9 & 7 \end{pmatrix}$$

En lugar de utilizar el asterisco podríamos denotar el producto por yuxtaposición situando el número junto a la matriz:

$$3\begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & -1 \end{pmatrix}$$

In[3]:= $\begin{pmatrix} 1 & 3 & 6 \\ 2 & -1 & 1 \end{pmatrix} . \begin{pmatrix} 3 & 1 \\ 6 & -5 \\ 2 & 0 \end{pmatrix}$

Out[3]//MatrixForm=

$\begin{pmatrix} 33 & -14 \\ 2 & 7 \end{pmatrix}$

Véase que para el producto de matrices empleamos el signo de puntuación '`.`' y no el asterisco que se usa solamente para el producto de un número por una matriz.

Podemos utilizar variables para almacenar matrices y luego efectuar operaciones sobre ellas tal y como se observa en el siguiente ejemplo:

Ejemplo 22:

In[1]:= **a** $= \begin{pmatrix} 5 & 2 & 0 & 0 \\ 1 & 2 & 2 & 3 \\ -1 & 2 & 0 & 3 \end{pmatrix}$; **b** $= \begin{pmatrix} 6 & 3 \\ 0 & 1 \\ 2 & 3 \\ -5 & 8 \end{pmatrix}$; **c** $= \begin{pmatrix} 7 & 0 \\ 4 & 8 \\ 4 & 7 \\ 7 & 5 \end{pmatrix}$;

Out[1]//MatrixForm=

Null

In[2]:= **b+c**

Out[2]//MatrixForm=

$\begin{pmatrix} 13 & 3 \\ 4 & 9 \\ 6 & 10 \\ 2 & 13 \end{pmatrix}$

In[3]:= **4a**

Out[3]//MatrixForm=

$\begin{pmatrix} 20 & 8 & 0 & 0 \\ 4 & 8 & 8 & 12 \\ -4 & 8 & 0 & 12 \end{pmatrix}$

In[4]:= **a.b**

Out[4]//MatrixForm=

$\begin{pmatrix} 30 & 17 \\ -5 & 35 \\ -21 & 23 \end{pmatrix}$

Ejercicios

1. Dadas las matrices $A = \begin{pmatrix} -1 & 2 & 12 \\ 32 & 10 & -25 \\ 1 & 0 & 2 \end{pmatrix}$, $B = \begin{pmatrix} 4 & 2 \\ 21 & 12 \\ 3 & -9 \end{pmatrix}$, $C = \begin{pmatrix} 7 & 4 \\ 2 & 7 \\ -1 & 2 \end{pmatrix}$ y $D = \begin{pmatrix} -7 & 2 \\ 0 & 4 \end{pmatrix}$, calcúlese la matriz $A \cdot B - C \cdot D$ (a la hora de trabajar en MATHEMATICA hay que recordar que los nombres de variables deben comenzar por minúscula).
2. Efectúese alguna operación con matrices de dimensiones incorrectas (por ejemplo sumar matrices de tipos distintos o multiplicar matrices cuyos tipos no son los adecuados) y obsérvese cual es el resultado.

10.2. Funciones básicas para el tratamiento de matrices

A continuación veremos una lista de funciones que nos permiten manejar matrices de diferentes formas:

- `Dimensions[a]` nos proporcionará el tipo de la matriz `a`. Si la matriz es de tipo $m \times n$ obtendremos `{m,n}`.
- `Transpose[a]` nos proporciona la matriz transpuesta de la matriz `a`.
- `a[[i,j]]` nos proporciona el elemento (i,j) de la matriz `a`.

- `a[[i]]` nos proporciona la fila i de la matriz `a`.
- `DiagonalMatrix[{`a_1`,`a_2`,...,`a_n`}]` nos proporciona la matriz diagonal cuya diagonal principal es $(a_1\ \ a_2\ \dots\ a_n)$.
- `IdentityMatrix[n]` genera la matriz identidad de orden n.
- `Inverse[a]` calcula la matriz inversa de la matriz cuadrada `a`. En ningún caso podremos obtener la inversa de la matriz `a` mediante la instrucción `a^(-1)`.
- `MatrixPower[a,n]` calcula la `n`-ésima potencia de la matriz `a` (es decir,la matriz a^n). La instrucción `MatrixPower` admite valores negativos de `n`. Téngase en cuenta que si intentamos efectuar el cálculo a^n mediante la instrucción `a^n` el resultado no será el esperado, de manera que la única forma de efectuar un cálculo de este tipo es mediante el uso de la función `MatrixPower`.
- `Det[a]` calcula el determinante de la matriz cuadrada `a`.
- `Minors[a,k]` proporciona una lista con los determinantes de todos los menores de orden `k` de la matriz `a`.
- `RowReduce[a]` simplifica la matriz `a` mediante la aplicación de operaciones elementales por filas.

Ejemplo 23:

Veremos a continuación una serie de ejemplos en los que aparecen las funciones que acabamos de describir. Para ejecutar el ejemplo hemos activado previamente la opción `$Post=MatrixForm`.

In[1]:= $\mathbf{a}=\begin{pmatrix}1&4&5&6\\2&4&-5&6\\3&5&6&2\end{pmatrix};\mathbf{b}=\begin{pmatrix}3&2&5\\3&7&5\\1&8&4\end{pmatrix};$

Out[1]//MatrixForm=
Null

El uso de la opción MatrixForm produce, en ocasiones; resultados inesperados

In[2]:= **`Dimensions[a]`**
Out[2]//MatrixForm=
$\begin{pmatrix}3\\4\end{pmatrix}$

In[3]:= **`Transpose[a]`**
Out[3]//MatrixForm=
$\begin{pmatrix}1&2&3\\4&4&5\\5&-5&6\\6&6&2\end{pmatrix}$

In[4]:= **`a[[3]]`**
Out[4]//MatrixForm=
$\begin{pmatrix}3\\5\\6\\2\end{pmatrix}$

MATHEMATICA no distingue entre matrices fila y matrices columna. Para el programa son iguales las matrices

$\begin{pmatrix}3&4\end{pmatrix}$ y $\begin{pmatrix}3\\4\end{pmatrix}$

que se representa en el lenguaje interno como `{3,4}`; que se representa en el lenguaje interno como `{{3},{4}}`

Cuando se aplica `MatrixForm`, las matrices fila y columna se representan ambas como columnas. Así por ejemplo,

- La salida Out[2], si no estuviera activada la opción `MatrixForm` sería `{3,4}` que es una matriz fila que `MatrixForm` representa como columna.
- La salida Out[4] corresponde a la entrada `a[[3]]` que denota la fila 3 de la matriz `a`. Por tanto en el lenguaje interno esta salida es en realidad `{3,5,6,2}` que es una matriz fila pero `MatrixForm` la representa como columna.

In[5]:= **`DiagonalMatrix[{1,2,-1,1}]`**
Out[5]//MatrixForm=
$\begin{pmatrix}1&0&0&0\\0&2&0&0\\0&0&-1&0\\0&0&0&1\end{pmatrix}$

In[6]:= **`IdentityMatrix[3]`**

Out[6]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In[7]:= **Inverse[b]**

Out[7]//MatrixForm=

$$\begin{pmatrix} -\frac{12}{35} & \frac{32}{35} & -\frac{5}{7} \\ -\frac{1}{5} & \frac{1}{5} & 0 \\ \frac{17}{35} & -\frac{22}{35} & \frac{3}{7} \end{pmatrix}$$

In[8]:= **MatrixPower[b,3]**

Out[8]//MatrixForm=

$$\begin{pmatrix} 285 & 820 & 580 \\ 460 & 1295 & 930 \\ 424 & 1180 & 849 \end{pmatrix}$$

In[9]:= **Det[b]**

Out[9]//MatrixForm=

35

In[10]:= **Minors[a,2]**

Out[10]//MatrixForm=

$$\begin{pmatrix} -4 & -15 & -6 & -40 & 0 & 60 \\ -7 & -9 & -16 & -1 & -22 & -26 \\ -2 & 27 & -14 & 49 & -22 & -46 \end{pmatrix}$$

In[11]:= **Minors[a,3]**

Out[11]//MatrixForm=

$$\begin{pmatrix} -69 & 22 & 186 & 220 \end{pmatrix}$$

Sabemos que el rango de la matriz `a` es el orden (el tamaño) del mayor menor con determinante distinto de cero. Esta matriz tiene menores de tamaño 1, 2 y 3. Tenemos que:

- La matriz `a` tiene 18 menores de orden 2. El determinante de cada uno de ellos puede calcularse con la instrucción `Minors[a,2]`. Podemos ver que la mayoría de estos menores tienen determinante distinto de cero.
- La matriz `a` tiene 4 menores de orden 3 y podemos calcular sus determinantes mediante `Minors[a,3]`.

Los menores de mayor tamaño con determinante no nulo son los de orden 3 así que el rango de la matriz es 3.

In[12]:= **RowReduce[a]**

Out[12]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & -\frac{220}{69} \\ 0 & 1 & 0 & \frac{62}{23} \\ 0 & 0 & 1 & -\frac{22}{69} \end{pmatrix}$$

Una segunda manera para calcular el rango de una matriz consiste en aplicar operaciones elementales tanto por filas como por columnas hasta reducirla a su forma diagonal. Entonces, contando los elementos no nulos que aparezcan en la diagonal obtenemos el rango. La instrucción RowReduce intenta reducir la matriz aplicándole operaciones por filas. Si cambiamos las filas por columnas haciendo la transpuesta, la instrucción RowReduce actuará entonces sobre columnas. Podemos aplicar Transpose una vez más para disponer filas y columnas como al principio.

In[13]:= **Transpose[RowReduce[Transpose[RowReduce[a]]]]**

Out[13]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Una vez reducida la matriz `a` tiene tres unos en diagonal y por tanto su rango es 3.

Ejercicios

1. Determínese si la matriz

$$\begin{pmatrix} -1 & 0 & 1 & 2 \\ 3 & 2 & -1 & 1 \\ 2 & 1 & 0 & 2 \\ 1 & 3 & 0 & 1 \end{pmatrix}$$

es regular y en caso de que lo sea obténgase su matriz inversa.

2. Calcúlese el valor de

$$\begin{pmatrix} -1 & 0 & 1 \\ 1 & 2 & -1 \\ 0 & 1 & 1 \end{pmatrix}^8 .$$

3. Obténgase la matriz adjunta de
$$\begin{pmatrix} -1 & 2 & -4 & 0 & 2 \\ 3 & 6 & -3 & 0 & 1 \\ 2 & 4 & 7 & 10 & 1 \\ 0 & 0 & 2 & 4 & 3 \\ 8 & 6 & 9 & -1 & 1 \end{pmatrix}.$$

4. Encontrar las matrices C_1 y C_2 que verifican las ecuaciones siguientes:

$$C_1 \cdot \begin{pmatrix} 1 & -2 & 1 \\ 4 & -1 & 2 \\ 1 & 1 & 6 \end{pmatrix} = \begin{pmatrix} 1 & -4 & 6 \\ 2 & 1 & 0 \\ -1 & 1 & 2 \end{pmatrix} \quad \text{y} \quad C_1^{\,4} \cdot C_2 \cdot C_1 = \begin{pmatrix} -1 & 4 & 2 \\ 0 & 3 & 2 \\ 0 & 0 & 1 \end{pmatrix}.$$

5. Almacenar en la variable `I30` la matriz identidad de orden 30.

6. Almacenar en la variable `diagonal` la matriz diagonal cuya diagonal principal es (-1,0,1,0,3,-1,3,4,5).

7. Dados los vectores (1,2,-1,1), (2,0,3,-1), (1,1,7,6) $\in \mathbb{R}^4$, contestar a las siguientes cuestiones:

   - ¿Son independientes?

   - ¿ Son una base de $\mathbb{R}^4$?

   - Puede obtenerse (3,10,19,29) combinando estos tres vectores. Dicho de otra manera:

     $$¿\,(3,10,19,29) \in \langle (1,2,-1,1),(2,0,3,-1),(1,1,7,6) \rangle\,?$$

     En caso de que la respuesta sea afirmativa, determinar como han de combinarse dichos vectores para obtener efectivamente el vector (3,10,19,29). Es decir, calcular los valores de los coeficientes α, β y γ tales que multiplicados por los tres vectores proporcionan el buscado en la forma,

     $$(3,10,19,29) = \alpha\,(1,2,-1,1) + \beta\,(2,0,3,-1) + \gamma\,(1,1,7,6).$$

     Repetir el ejercicio para el vector (1,2,3,4). Comprobar los resultados obtenidos.

8. Comprobar si el conjunto de vectores (1,2,-1,3,2), (1,-1,0,3,1), (3,7,2,-4,1), (2,6,2,1,1), (2,3,-8,7,1) forman base $\mathbb{R}^5$. Estudiar de qué manera han de combinarse estos vectores para obtener el vector (1,2,3,4,5), el vector (1,0,0,0,0) ó el vector (0,1,3,1,2) (comprobar los resultados obtenidos). ¿ Existe algún vector de $\mathbb{R}^5$ que no pueda ser obtenido combinando los cinco vectores indicados antes ?

NOTA: Para resolver los ejercicios 7 y 8 es de utilidad la instrucción `Solve` que permite obtener las soluciones de una ecuación o sistema de ecuaciones algebraicas. La sintaxis para el comando es la siguiente:

Comando: `Solve`
Sintaxis:

1) Ecuación de una variable: `Solve[ecuación,x]`

2) Sistema de ecuaciones:

`Solve[{ecuacion₁,...,ecuacionₘ},{x₁,...,xₙ}]`

Resultado: **1)** Resuelve la ecuación `ecuacion` despejando en ella la variable `x`.

2) Resuelve el sistema formado por las ecuaciones $ecuacion_1$,..., $ecuacion_m$ despejando en ellas las variables x_1,...,x_n.

■ Es importante tener en cuenta que en MATHEMATICA el operador de comparación que debe aparecer en una ecuación es == (doble =) nunca = (un = sencillo) que es el operador de asignación.

Por ejemplo:

In[1]:= `Solve[a x^2+b x+c==0,x]`

Out[1]= $\{\{x \to \frac{-b-\sqrt{b^2-4ac}}{2a}\},\{x \to \frac{-b+\sqrt{b^2-4ac}}{2a}\}\}$

In[2]:= `Solve[α{1,2,-1}+β{2,1,1}=={4,5,-1},{α,β}]`

Out[2]= `{{α→2,β→1}}`

Al resolver la ecuación de segundo grado obtenemos las dos soluciones posibles.

Muy importante emplear en las ecuaciones un doble igual (==) en lugar de un igual sencillo (=).

Si desarrollamos las operaciones, la ecuación

`α{1,2-1}+β{2,1,1}=={4,5,-1}`

equivale al sistema

$$\begin{cases} \alpha + 2\beta = 4 \\ 2\alpha + \beta = 5 \\ -\alpha + \beta = -1 \end{cases}$$

en el que las variables a despejar son α y β. Véase que lo que estamos calculando aquí son los coeficientes α y β por los que hay que multiplicar a los vectores (1,2,-1) y (2,1,1) para obtener el vector (4,5,1).

ELEMENTOS DEL PROGRAMA

- El símbolo % se utiliza para referirse a una salida determinada. A medida que vamos efectuando operaciones vamos obteniendo una lista de entradas (In) y salidas (Out) que se irán numerando ordenadamente. Es posible que necesitemos emplear un resultado que hemos calculado con anterioridad, empleando el símbolo % es posible evitar el teclear nuevamente tal resultado. La sintaxis de uso para % es:
  - % n = Out nº n.
  - % = Out inmediatamente anterior.
  - %% = Out penúltimo.
  - %%% = Out antepenúltimo.
  - etc.

Ejemplo 24:

In[1]:= $1+\frac{3}{2}$

Out[1]= $\frac{5}{2}$

In[2]:= `%+1`

Out[2]= $\frac{7}{2}$

In[3]:= `%%-1`

Out[3]= $\frac{3}{2}$

In[4]:= $(x+y)^2$

Out[4]= $(x+y)^2$

In[5]:= x^3

Out[5]= x^3

In[6]:= `Expand[%4+%5]`

In[6]:= $2xy+x^2+y^2+x^3$

- Podemos eliminar cualquiera de las celdas que aparezcan en el área de trabajo. Para ello pulsaremos con el ratón sobre la celda elegida para marcarla y posteriormente la eliminaremos con la tecla de borrado.
- Todo comando o palabra clave que forma parte o viene incorporado en el programa MATHEMATICA se escribe con la primera letra de cada palabra en mayúsculas y las demás en minúsculas. Por ejemplo: `Pi`, `Sin`, `Sqrt`, `MatrixForm` (palabra compuesta por `Matrix` y `Form`, véase que cada una de las palabras componentes sí empieza con mayúscula).
- Cuando cometemos un error de cualquier tipo el programa nos lo advierte mediante un mensaje de color azul.

Ejemplo 25*:*

In[1]:= $\frac{1}{0}$

Power::infty : Infinite expression $\frac{1}{0}$ encountered.

Out[1]:= `ComplexInfinity`

- La instrucción ? nos permite obtener información acerca del funcionamiento de los distintos comandos que forman parte del lenguaje de MATHEMATICA. Se utiliza en la forma

  `?comando`

  tras lo cual obtendremos la información solicitada.

Ejemplo 26:

```
In[1]:= ?Inverse
        Inverse[m] gives the inverse of a square matrix m.
In[2]:= ?Transpose
        Transpose[list] transposes the first two levels in list.
        Transpose[list, {n1,n2, ...}] transposes list so that the nk-th
        level in list is the k-th level in the result.
```

- En una misma línea y celda podemos incluir más de una instrucción siempre que las separemos mediante ”;”

```
In[1]:= a=3; b=2;
        a+b
Out[1]= 5
```

Un signo ”;” al final de una instrucción evita que se muestre el resultado producido por tal operación:

```
In[2]:= c=a+b
Out[2]= 3
In[3]:= c=a+b;
```

- Recuérdese que:

Los paréntesis ( )	Se utilizan para agrupar operaciones algebraícas. `(1+3)*2`
Los corchetes simples [ ]	Se utilizan para agrupar los argumentos de una función. `Cos[4]`
Las llaves { }	Se utilizan para agrupar los elementos de una lista. `{3,1,-1,2}`
Los dobles corchetes [[ ]]	Se utilizan para obtener un elemento concreto dentro de una lista. `{1,2,3}[[3]]`