 *Revista digital Matemática, Educación e Internet* (www.cidse.itcr.ac.cr/). Vol. 8, No 1. (Julio., 2008)

Algebra Lineal con Mathematica. Introducción.

Carlos Arce S.
carce@cariari.ucr.ac.cr
Escuela de Matemática
Universidad de Costa Rica

Resumen

Este es el capítulo introductorio para el material que se usó en el curso MA0429 Matemática para Computación IV. Esta primera parte trata sobre conceptos básicos que se deben recordar al trabajar con Mathematica. Actualizado al 5 de agosto de 2004.

Palabras claves: Cálculo simbólico, Mathematica.

Contenido

1.1 Cómo transmitir órdenes a Mathematica 3

1.2 Convenciones básicas del lenguaje *Mathematica* 3

1.2.1 Los nombres de comandos comienzan con mayúscula 3

1.2.2 Las funciones se evalúan usando paréntesis cuadrados 4

1.3 El operador de multiplicación puede omitirse pero ... 5

1.4 Los símbolos especiales % y ; 6

1.5 La aritmética usual en Mathematica es exacta 7

1.6 Usando aritmética aproximada y la función N. 7

1.7 Operaciones con expresiones algebraicas 9

1.8 Gráficos 9

1.9 Animación 11

1.10 Definición de funciones 13

1.11 Evitando la aritmética compleja 14

1.12 Solución de ecuaciones 15

1.13 Cálculo de derivadas 17

1.14 Cálculo de Integrales 18

1.1 CÓMO TRANSMITIR ÓRDENES A MATHEMATICA

Las órdenes se transmiten presionando, simultáneamente, las teclas [Shift][Enter].

■ **EJEMPLO 1.1**

Cálculo de 2^8 y solución de la ecuación $ax^2+bx+c=0$.

```
In[ ]:=   2^8
In[ ]:=   Solve[ax^2 + bx + c == 0 , x]
```

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

1.2 CONVENCIONES BÁSICAS DEL LENGUAJE *MATHEMATICA*

1.2.1 Los nombres de comandos comienzan con mayúscula

Los nombres de las funciones, operadores, comandos, constantes, etc., que *Mathematica* utiliza son palabras completas del inglés que siempre comienzan con MAYUSCULA:

```
 In[ ]:=   Integrate[x Sin[x], {x, 0, Pi}]
Out[ ] =   π
```

Naturalmente, la interface gráfica que aporta el FrontEnd, permite utilizar la simbología clásica, utilizando las "paletas"

```
 In[ ]:=   ∫_0^π x Sin[x]dx
Out[ ] =   π
```

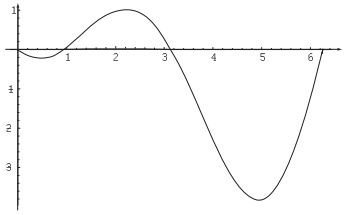
Un error frecuente al comenzar a trabajar en Mathematica es escribir los nombres con minúscula, como en el siguiente caso:

```
 In[ ]:=   ∫_0^π x cos[x]dx

          General::spell1:Possible spelling error: new
             symbol name "cos" is similar to existing symbol "Cos". More...

Out[ ] =   ∫_0^π x cos[x]dx
```

1.2.2 Las funciones se evalúan usando paréntesis cuadrados

Las funciones y comandos de Mathematica se evalúan utilizando paréntesis cuadrados (y no redondos como es usual en la matemática). Los paréntesis redondos sólo se utilizan para agrupar, en expresiones aritméticas o lógicas y las llaves definen listas. En los tres casos `---[], (),{}---` sus roles están claramente determinados y no pueden intercambiarse.

```
 In[ ]:=  Plot[(x - 1) Sin[x], {x, 0, 2π}];
```

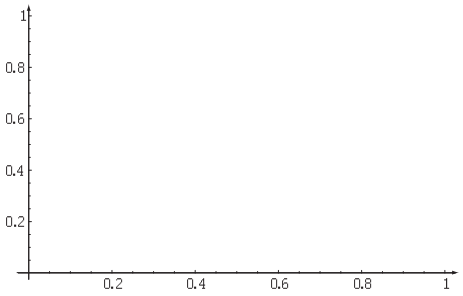


Observe el uso de: a) `[ ]` en **Plot** y **Sin** —comando y función—, b) `( )` para agrupar x-1 en la expresión que define la función a graficar y c) `{ }` en la lista `[x, 0, 2π]`, que constituye el segundo parámetro.

Un error frecuente es escribir `Sin(x)` en vez de `Sin[x]`, como en el siguiente caso:

```
 In[ ]:=  Plot[x Sin(x), {x, 0, 2π}];
```

```
Plot::plnr :
xSinx is not a machine-size real number at x=2.617993877991494‘*^-7. More...

Plot::plnr :
xSinx is not a machine-size real number at x=0.25488992540742256‘. More...

Plot::plnr :
xSinx is not a machine-size real number at x=0.5328694051959509‘. More...

General::stop:
 Further output of Plot::plnr will be suppressed during this calculation. More...
```



1.3 EL OPERADOR DE MULTIPLICACIÓN PUEDE OMITIRSE PERO ...

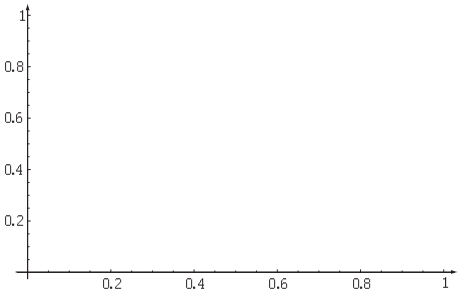
El operador de multiplicación en *Mathematica* es el * (asterisco) o un espacio en blanco, los cuales pueden ser omitidos siempre que no haya ambigüedad.

```
 In[ ]:=  {2*Cos[π/3], 2 Cos[π/3], 2Cos[π/3]}
Out[ ] =  {1, 1, 1}
 In[ ]:=  {Factor[a x + a], Factor[ax + a]}
Out[ ] =   a (1 + x), a + ax
```

Observe en el ejemplo anterior, que *Mathematica* entiende ax como un nuevo símbolo y no como el producto de a por x. Este tipo de ambigüedad origina errores frecuentes como el siguiente:

```
 In[ ]:=  Plot[xSin[x], {x, 0, π}]


Out[ ]=    General::spell1 : Possible spelling error: new
           symbol name "xSin" is similar to existing symbol "Sin". More...

         Plot::plnr :
          xSin[x] is not a machine-size real number at x = 1.308996938995747‘*^-7. More...
```

```
Plot::plnr :
 xSin[x] is not a machine-size real number at x = 0.12744496270371128‘. More...

Plot::plnr :
 xSin[x] is not a machine-size real number at x = 0.26643470259797547‘. More...

General::stop:
  Further output of Plot::plnr will be suppressed during this calculation. More...
```




```
Out[ ] =  -Graphics-
```

1.4 LOS SÍMBOLOS ESPECIALES % Y ;

El símbolo %

Mathematica utiliza el símbolo % para hacer referencia a resultados anteriores:

% para hacer referencia al último resultado obtenido.

%% para referir el penúltimo resultado, y así sucesivamente.

%n hace referencia al resultado Out[n].

```
 In[ ]:=  (1/2)^10
Out[ ] =  1/1024
 In[ ]:=  %^(1/10)
Out[ ] =  1/2
```

El símbolo ;

El símbolo ; permite concatenar comandos para formar secuencias de operaciones. El resultado de una secuencia de operaciones es el resultado de la última expresión evaluada.

```
 In[ ]:=   a = 1; b = 3; a + b
Out[ ] =   4
```

Muchas veces se usa el símbolo ; para terminar un comando y evitar la respuesta que *Mathematica* produce. Esto porque al terminar un comando con ; se crea una secuencia cuya última operación es nula y su evaluación resulta vacía.

```
 In[ ]:=  a = 1; b = 3; a + b;
 In[ ]:= %
Out[ ] =   4
```

1.5 LA ARITMÉTICA USUAL EN MATHEMATICA ES EXACTA

La aritmética de Mathematica es exacta siempre que se trabaje con números exactos, es decir, . enteros, racionales (cocientes de enteros), radicales, números e y π.

```
 In[ ]:=   1/3 + 4/7
Out[ ] =   4 19/21
 In[ ]:=   3^100
Out[ ] =   515377520732011331036461129765621272702107522001
 In[ ]:=   Sqrt[8] Sin[π/2]
Out[ ] =   2√2
 In[ ]:=   √8 ∛2
Out[ ] =   4√2
 In[ ]:=   Solve[x^2 - 3 == 0,x]
Out[ ] =   {{x → -√3}, {x → √3}}
```

Esta aritmética incluye el cálculo simbólico con expresiones algebraicas.

```
 In[ ]:=   (3x + 2x + 6y)(x-1)
Out[ ] =   (-1 + x) (5x + 6y)
```

1.6 USANDO ARITMÉTICA APROXIMADA Y LA FUNCIÓN N.

Cuando se utilizan números con punto decimal, denominados números aproximados, *Mathematica* realiza aritmética aproximada o de punto flotante.

```
 In[ ]:=   4.0/7 + 1/2
Out[ ] =   1.07143
 In[ ]:=   Sqrt[8.]
Out[ ] =   2.82843
```

La función N[exp] determina una aproximación numérica para exp utilizando la aritmética en punto flotante que provee el CPU (esto supone aritmética con aproximadamente 16 dígitos decimales, en los procesadores Pentium). Cuando se desea que *Mathematica* utilice

aritmética usando un número . mayor de decimales, debe especificarse este como segundo argumento.

```
 In[ ]:=   N[Sqrt[2],30]
Out[ ] =   1.41421356237309504880168872421
 In[ ]:=   N[Pi,500]


Out[ ]=         3.14159265358979323846264338327950288419716939937510582
                09749445923078164062862089986280348253421170679821480
                86513282306647093844609550582231725359408128481117450 2
                84102701938521105559644622948954930381964428810975665 9
                33446128475648233786783165271201909145648566923460348 6
                10454326648213393607260249141273724587006606315588174 8
                81520920962829254091715364367892590360011330530548820 4
                66521384146951941511609433057270365759591953092186117 3
                81932611793105118548074462379962749567351885752724891 2
                279381830119491
```

Mathematica no garantiza que los resultados haciendo aritmética aproximada tengan el número de dígitos exactos que se indica en la función `N`, sino más bien que en la aritmética empleada para hacer la aproximación, se han empleado esa cantidad de dígitos decimales para representar cada número aproximado.

Además como toda función en Mathematica, `N` puede ser utilizada en notación postfija utilizando el operador `//` :

```
 In[ ]:=   √8 // N
Out[ ] =   2.82843
 In[ ]:=   π/2.0//Sin
Out[ ] =   1.
```

Al usar la función `N` no tiene sentido indicar un número menor que 16, porque la menor precisión en la aritmética de punto flotante de *Mathematica* es la que provee el procesador central (16 o 17 dígitos). Por otra parte, cuando la precisión es la normal (16 o 17 dígitos) Mathematica no despliega todos los decimales obtenidos, es decir, no debe confundirse el número de dígitos de precisión de la aritmética con el número de dígitos con que se despliegan los resultados, el cual por omisión es 6. Este es un valor fijado en la siguiente opción del FrontEnd:

Edit $\rightarrow$ *Preferences* $\rightarrow$ *Formatting Options* $\rightarrow$ *Expression Formatting* $\rightarrow$ *Display Options* $\rightarrow$ *PrintPrecision*

Usted puede verificar que *Mathematica* retiene más decimales de precisión que los mostrados, seleccionando la celda con la salida y activando la opción **Show Expression** del menú **Format**.

1.7 OPERACIONES CON EXPRESIONES ALGEBRAICAS

Mathematica provee numerosos recursos para trabajar con expresiones algebraicas, a continuación se introducen los comandos **Factor**, **Expand** y **Simplify**, cuyas acciones pueden deducirse fácilmente.

```
 In[ ]:=   Expand[(x+a)^2 (2x + 1) + 4x + 2]
Out[ ] =   2x^3 + 4ax^2 + x^2 + 2a^2x + 2ax + 4x + a^2 + 2
 In[ ]:=   Factor[%]
Out[ ] =   (2x + 1)(a^2 + 2xa + x^2 + 2)
```

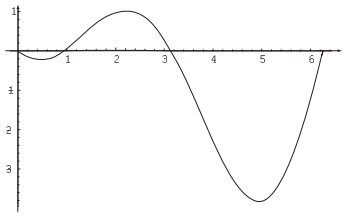
`Simplify` escribe la expresión en la forma que "parece" más simple, en general, empleando el menor número de subexpresiones.

```
 In[ ]:=   Simplify[(2x^3 + 5x^2 + 4x + 1) / (x+1)]
Out[ ] =   1 + 3x + 2x^2
 In[ ]:=   Factor[2x^3 + 5x^2 + 4x + 1]
Out[ ] =   (1 + x)^2 (1 + 2x)
 In[ ]:=   Factor[(2x^3 + 5x^2 + 4x + 1)/(x + 1)]
Out[ ] =   (1 + x)(1 + 2x)
```

1.8 GRÁFICOS

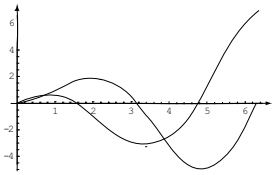
La primitiva `Plot`, el comando básico de *Mathematica* para construir gráficos de funciones en una variable, requiere como mínimo dos parámetros: la función a representar y el intervalo donde varía la variable, todo esto en la siguiente forma:

```
 In[ ]:=   Plot[ x Sin[x],{x,0,2Pi}];
```



Se puede construir el gráfico de varias funciones simultáneamente, por ejemplo:

```
 In[ ]:=   Plot[{x Cos[x], x Sin[x]},{x,0,2Pi}];
```

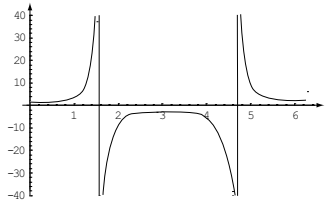


Muchas características del gráfico, como los ejes y sus marcas, están determinadas por ciertos valores por omisión que pueden ser cambiados mediante parámetros opcionales. La lista completa de estos parámetros y sus valores por omisión se obtiene con el comando Options:

```
 In[ ]:= Options[Plot]
Out[ ] =
     {AspectRatio → 1/GoldenRatio, Axes → Automatic, AxesLabel → None,
     AxesOrigin → Automatic, AxesStyle→ Automatic, Background→ Automatic,
     ColorOutput → Automatic, Compiled→ True, DefaultColor→ Automatic,
     DefaultFont :→ $DefaultFont, DisplayFunction :→ $DisplayFunction,
     Epilog→ {}, FormatType :→$FormatType, Frame → False,
     FrameLabel→ None, FrameStyle→ Automatic, FrameTicks→ Automatic,
     GridLines→ None, ImageSize→ Automatic, MaxBend→ 10.,
     PlotDivision→ 30., PlotLabel → None, PlotPoints → 25,
     PlotRange→ Automatic, PlotRegion→ Automatic, PlotStyle→ Automatic,
     Prolog→ {}, RotateLabel→ True, TextStyle :→ $TextStyle,
     Ticks→ Automatic}
```

En el capítulo 2, del libro Introducción a *Mathematica* en la construcción de gráficos, describe las opciones más frecuentemente empleadas, por ejemplo observe que la opción **PlotRange** → **{-40,40}** permite especificar el intervalo para el eje Y que muestra el gráfico.

```
 In[ ]:= Plot[x/Cos[x],{x,0,2Pi},PlotRange->{-40,40}];
```

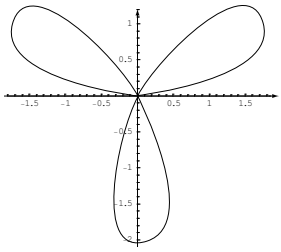


■ **EJEMPLO 1.2**

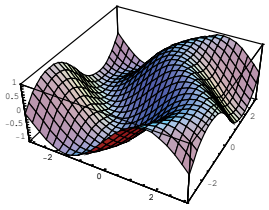
Con los siguientes comandos se muestran otras primitivas de *Mathematica*, que permiten construir gráficos en 2 y 3 dimensiones.

```
 In[ ]:= r[t_] := 2 Sin[3t]
 In[ ]:= ParametricPlot[{r[t] Cos[t],r[t] Sin[t]},
                        {t,0,2Pi},AspectRatio->Automatic,
```

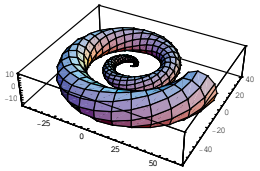
```
TextStyle->{FontSize->8}]
```



```
In[ ]:= Plot3D[Sin[x+Sin[y]],{x,-Pi,Pi},{y,-Pi,Pi},TextStyle->{FontSize->8}]
```



```
Out[ ] = -SurfaceGraphics-
 In[ ]:= ParametricPlot3D[{u Cos[u](4 + Cos[u+v]),
                           u Sin[u](4 + Cos[u+v]),u Sin[u+v]},
                           {u,0,4Pi},{v,0,2Pi},PlotPoints->{60,12},
                           TextStyle->{FontSize->8}];
```



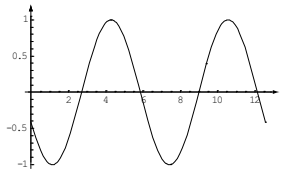
1.9 ANIMACIÓN

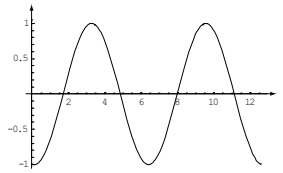
Una función que dependa de un cierto parámetro a, puede dar origen a varios gráficos haciendo variar el valor del parámetro a. Cada gráfico refleja el comportamiento de la curva para un valor determinado de a.

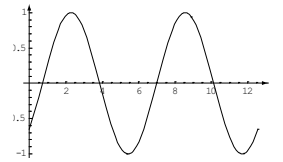
Para construir una secuencia de gráficos con una sola orden de Mathematica, puede utilizarse el comando Do en la siguiente forma:

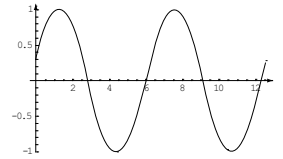
```
In[ ]:= Do[Plot[ Cos[x + a], {x, 0, 4Pi}], {a, 2, 8}]
```

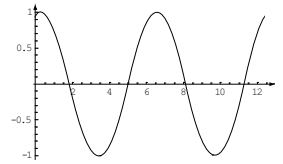
12

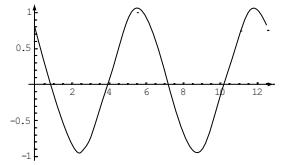

1
0.5
2
4
6
8
10
12
−0.5
−1

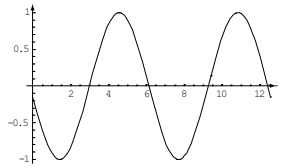

1
0.5
2
4
6
8
10
12
−0.5
−1


1
2
4
6
8
10
12
−1


1
0.5
2
4
6
8
10
12
−0.5
−1


1
0.5
2
4
6
8
10
12
−0.5
−1


1
0.5
2
4
6
8
10
12
−0.5
−1

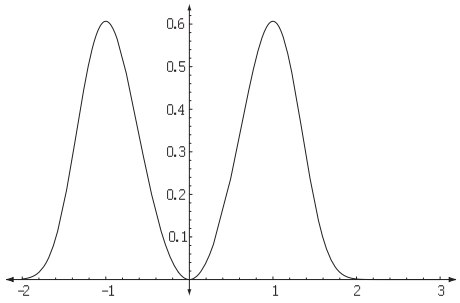

1
0.5
2
4
6
8
10
12
−0.5
−1

La animación de estos gráficos, se observa al seleccionar la celda que los contiene y activar el comando del menú *cell→Animate Selected Graphics*, o presionar **Ctrl+Y**. O también, al dar un doble click sobre la celda que contiene los gráficos, estos se ocultan mostrando solo el primero, y sobre este gráfico nuevamente con un doble click se obtiene la animación.

1.10 DEFINICIÓN DE FUNCIONES

La definición de funciones en *Mathematica* emplea los símbolos especiales _ (Blanck[]) y := (SetDelayed). El primero es un patrón que indica que _ (Blanck[]) sustituye a cualquier expresión, y debe ser usado a la izquierda de := . Cuando se escribe x_ significa que la expresión que sustituye _ (Blanck[]) se llamará x.

```
In[ ]:= f[x_]  := xExp[-x^2/2]
In[ ]:= Plot[f[t^2], {t, -2, 3}];
```



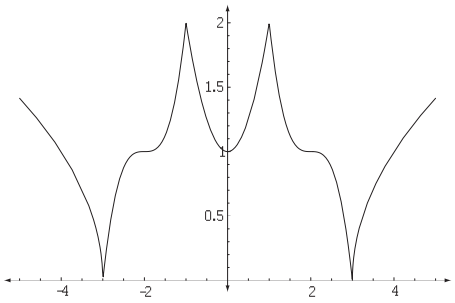
Al definir f[x], es un error que el símbolo _ (Blank[]) aparezca a la derecha de := (SetDelayed), puesto que a la derecha debe aparecer la regla en términos de x (el nombre de expresión que recibirá f como argumento) que se usa para sustituir f[x].

Funciones con criterio dividido se definen utilizando el operador condicional /;, el cual puede ser traducido como si condicional. Observe la definición de la siguiente función f, que comienza con la orden Clear[f], para borrar cualquier regla que pudo haber sida asociada a f con anterioridad:

$$f[x] = \begin{cases} \sqrt{-3-x} & \text{Si} \quad x \leq -3 \\ (x+2)^3+1 & \text{Si} \quad -3 < x \leq -1 \\ 1+x^2 & \text{Si} \quad -1 < x \leq 1 \\ 1-(x+2)^3 & \text{Si} \quad 1 < x \leq 3 \\ \sqrt{-3+x} & \text{Si} \quad x > 3 \end{cases}$$

```
In[ ]:=  Clear[f];\\
         f[x_] := Sqrt[-3 - x]  /; x \mt{\leq}\verb# -3;#\\
         f[x_] := (x + 2)^3 + 1 /; -3 < x \mt{\leq}\verb# -1;#\\
         f[x_] := 1 + x^2       /; -1 < x \mt{\leq}\verb# 1;#\\
         f[x_] := 1 - (x - 2)^3 /; 1 < x \mt{\leq}\verb#  3;#\\
         f[x_] := Sqrt[-3 + x]  /; x > 3;\\
```

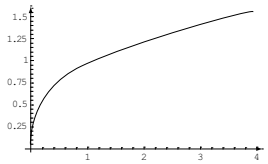
```
In[ ]:= Plot[f[x], {x, -5, 5}];
```



1.11 EVITANDO LA ARITMÉTICA COMPLEJA

Al construir el gráfico de algunas funciones, como $\sqrt{x}$, en un dominio que incluya números negativos — claramente $f(x)$ está bien definida para números negativos— puede ser que el resultado que produce *Mathematica* sea inesperado.

```
In[ ]:=   Plot[√x, {x, -4, 4}]
          Plot::plnr : x^(1/3) is not a machine-size real number at x=-4.. More...
          Plot::plnr : x^(1/3) is not a machine-size real number at x= -3.67546. More...
          Plot::plnr : x^(1/3) is not a machine-size real number at x= -3.32153. More...
          General::stop :
           Further output of Plot::plnr will be suppressed during this calculation. More...
```



Entre los mensajes de error se indica que $x^{\frac{1}{3}}$ no es un número real para $x = -3.999999667$, etc. Y efectivamente,como puede observarse a continuación, para *Mathematica* números como $\sqrt{-4.0}$ son complejos:

```
In[ ]:= ∛(-4.0)
Out[ ] = 0.793701 + 1.37473 I
```

El problema se produce por la definición que *Mathematica* da a las raíces de índice impar de los números negativos. Como para cada número real a hay un número real y dos complejos que satisfacen la ecuación $x^3 = a$, cualquiera de estos tres valores puede ser la raíz cúbica de a.

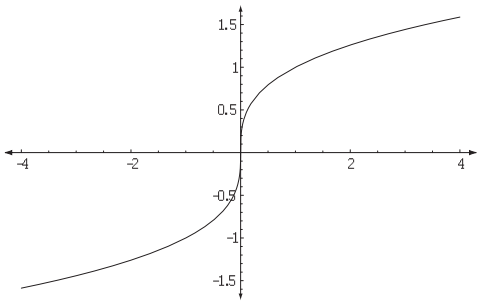
```
In[ ]:= Solve[x^3 == -8.0, x]
Out[ ] = {{x → -2.}, {x → 1. - 1.73205 I}, {x → 1. + 1.73205 I}}   In[ ]:= ∛(-8.0)
Out[ ] = 1. + 1.73205 I
```

De hecho entre las raíces mostradas en el ejemplo anterior, *Mathematica* elige la raíz principal, es decir, la raíz con el menor argumento positivo. Para un número complejo $z = a + bi$, su parte real es a, su parte imaginaria b y su argumento es el ángulo $-\pi < \theta < \pi$ tal que $z = r(\cos(\theta) + \text{sen}(\theta)I)$. En el caso de las tres raíces de la ecuación $x^3 = -8.0$, la raíz con menor argumento positivo es $1.0 + 1.73205I$, entonces $\sqrt[3]{-8.0} = 1.0 + 1.73205I$.

```
 In[ ]:=   {Arg[-2], Arg[1. - 1.73205I], Arg[1. + 1.73205I]}
Out[ ] =   {π, -1.0472, 1.0472}
```

Este tipo de problemas se resuelve indicándole a *Mathematica* que estamos interesados sólo en los resultados reales. Para ello se debe activar la biblioteca RealOnly, como se muestra a continuación.

```
 In[ ]:= Needs["Miscellaneous`RealOnly`"]
 In[ ]:= Plot[∛x, {x, -4, 4}]
```



1.12 SOLUCIÓN DE ECUACIONES

La primitiva **Solve** trata de resolver en forma exacta ecuaciones y sistemas de ecuaciones, principalmente ecuaciones polinomiales, aunque también se aplica a ecuaciones con radicales, trigonométricas o ecuaciones logarítmicas.

Solución exacta.

```
 In[ ]:= Solve[x^4 - 2x^3 + 5x -10 == 0, x]

Out[] = {{x -> 2}, {x -> (-5)^(1/3)},
        {x -> -5^(1/3)},{x -> (-(-1)^(2/3))*5^(1/3)}}

 In[ ]:= Solve[Sqrt{x}-1/Sqrt[x]== 1, x]

Out[ ] = {{x → 1/2 (3 + √5)}}

 In[ ]:= Solve[2 (Cos[x])^2 - √3 Cos[x] ==0, x]
```

```
          Solve::ifun :
           Inverse functions are being used by Solve, so some solutions may
             not be found; use Reduce for complete solution information. More...
```

```
Out[ ] = {{x → -π/2}, {x → π/6}, {x → π/6}, {x → π/2}}
```

```
 In[ ]:= Solve[{2x + y == 6,-3x + 2y == 6},{x,y}]
```

```
Out[ ] = {{x → 6/7, y → 30/7}}
```

```
 In[ ]:= Solve[{2x^2 + y^2 == 12,-3x + 2y == 6},{x,y}]
```

`Solve` procura obtener soluciones exactas de la ecuación dada, problema que puede resultar muy complejo, aún con ecuaciones relativamente sencillas, como por ejemplo:

```
 In[ ]:= Solve[x Sin[x] - 1 == 0, x]
```

```
           Solve::tdep :The equations appear to involve the variables
              to be solved for in an essentially non-algebraic way.More...
```

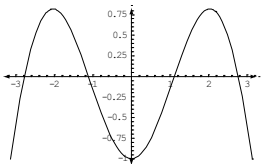
```
Out[ ] =  Solve[-1 + x Sin[x] == 0, x]
```

Solución aproximada: `FindRoot`.

Ecuaciones como la anterior se resuelven en forma numérica, esto es, se buscan aproximaciones a sus raíces por métodos numéricos.

Para el primer paso es construir un gráfico que permita visualizar los ceros y determinar algunos valores cercanos a la raíz buscada.

```
 In[ ]:= Plot[x Sin[x] - 1, {x, -Pi, Pi}]
```



Observe del anterior gráfico que `x Sin[x]==1` tiene cuatro raíces en el intervalo $[-3,3]$. Para calcular aproximadamente una de estas raíces, se puede utilizar el comando **FindRoot**, en la siguiente forma:

```
FindRoot[x Sin[x] - 1 == 0, {x,1}]
```

FindRoot utiliza el método de Newton para resolver aproximadamente la ecuación, el cual requiere dar un valor cercano a la raíz buscada. En el ejemplo anterior, al proveer `{x,1}`

como segundo parámetro se dice que resuelva respecto de la variable x y que el valor aproximado a la raíz es 1. Si se desea calcular aproximadamente la raíz más cercana a -3, el comando puede ser:

```
FindRoot[x Sin[x] == 1, {x,-3}]
```

No hay garantía de que el método de Newton converja a la raíz buscada, a partir del valor aproximado que usted provea. En algunos casos puede converger con casi cualquier valor y en otros podría requerir un valor muy cercano a la raíz. Las condiciones teóricas que requiere este método para aproximar con éxito la raíz buscada se verán en el capítulo ...

1.13 CÁLCULO DE DERIVADAS

El operador D permite el cálculo de derivadas:

$D[f[x],x]$ derivada de $f[x]$ respecto a x, con las sintaxis alternativas: $f'(x)$ y $\partial_x f[x]$.

$D[f[x],x,2]$ ó $f''(x)$: segunda derivada de $f[x]$ respecto a x.

$D[f[x,y],x,y]$ ó $\partial_{x,y} f[x,y]$: segunda derivada parcial de $f[x,y]$, respecto de x y de y.

```
 In[ ]:=  Clear[f]
          {D[f[x], x], f'[x], ∂x f[x]}
Out[ ] =  {f'[x], f''[x], f'''[x]}}
 In[ ]:=  D[x^5, x]
Out[ ] =  5x^4

 In[ ]:=  D[x^5 Sin[x] Exp[x^2], x]
Out[ ] =  exp^x^2 x^5 cos[x] + 5 exp^x^2 x^4 sin[x] + 2 exp^x^2 x^6 sin[x]

 In[ ]:=  D[x^5 Sin[x] Exp[x^2], {x, 2}]
Out[ ] =  2(5 exp^x^2 x^4 + 2 exp^x^2 x^6) cos[x] - exp^x^2 x^5 sin[x] + (20 exp^x^2 x^3 + 20 exp^x^2 x^5 +
x^5 (2 exp^x^2 + 4 exp^x^2 x^2)) sin[x]

 In[ ]:=  D[x^5 Sin[y] Exp[x^2], x, y]
Out[ ] =  5 exp^x^2 x^4 cos[y] + 2 exp^x^2 x^6 cos[y]
```

La paleta **BasicInputs** aporta símbolos apropiados para denotar los procesos del cálculo de derivadas, en la notación tradicional de la matemática.

```
 In[ ]:=  Clear[f];
          Table[D[f[x] == x Sin[x], {x, n}], {n, 0, 4}] // TableForm
```

$$\texttt{Out[ ] = TableForm =}\quad \begin{array}{rcrcr} f[x] & == & x\sin[x] & & \\ f'[x] & == & x\cos[x] & + & \sin[x] \\ f''[x] & == & 2\cos[x] & - & x\sin[x] \\ f^{(3)}[x] & == & -x\cos[x] & - & -3\sin[x] \\ f^{(4)}[x] & == & -4\cos[x] & + & x\sin[x] \end{array}$$

1.14 CÁLCULO DE INTEGRALES

Las integrales, tanto indefinidas como definidas, se calculan con el procedimiento **Integrate**. Nuevamente la paleta **BasicInputs** incluye los símbolos usuales para operar con integrales, sin embargo en esta introducción, se ha preferido usar los nombres de los operadores más que sus símbolos gráficos, para hacer énfasis en el lenguaje de comandos de *Mathematica*.

■ **EJEMPLO 1.3**

Calcular $\int xSin[x]dx$ y $\int_0^{\pi} xSin[x]dx$

```
 In[ ]:= Integrate[x Sin[x], x]
Out[ ] = -x Cos[x] + Sin[x]
 In[ ]:= Integrate[x Sin[x], {x, 0, Pi}]
Out[ ] = π
```

1.15 COMANDO TABLE

Con alguna frecuencia se requiere la construcción de listas de números u otros objetos, para lo que el comando **Table** resulta especialmente apropiado.

```
 In[ ]:= Table[2^k, {k, 1, 10}]
Out[ ] = {2, 4, 8, 16, 32, 64, 128, 256, 512, 1024}
```

Observe que **Table** requiere de dos parámetros: una expresión que depende de un índice y un "iterador" u objeto que indica el nombre del índice, su valor inicial y valor final. Este "iterador" puede incluir el incremento, como cuarto parámetro:

```
 In[ ]:= Table[a^k, {k, 1, 20, 2}]
Out[ ] = {a, a^3, a^5, a^7, a^9, a^11, a^13, a^15, a^17, a^19}
```

Este iterador puede asumir cuatro posibles formas:

1. `{n}`: se repite la expresión, como constante, n veces.
2. `{k, n}`: k varía de 1 a n, con incrementos de 1. Equivalente a `{k,1,n}`.
3. `{k,a,n}`: k asume los valores $a, a+1, a+2$, hasta un valor $b \leq n$ tal que $b+1 > n$.

4. `{k,a,n,h}`: k asume los valores $a, a+h, \ldots, a+ih \leq n$, hasta $a+(i+1)h > n$.

En todos los casos se genera una lista con n elementos.

```
 In[ ]:= Table[a^k , {10}]
Out[ ] = {a^k, a^k, a^k, a^k, a^k, a^k, a^k, a^k, a^k, a^k}
 In[ ]:= Table[a^k , {k, 10}]
Out[ ] = {a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^10}
 In[ ]:= Table[a^k , {k, 4,10}]
Out[ ] = {a^4, a^5, a^6, a^7, a^8, a^9, a^10}
 In[ ]:= Table[a^k , {k, 4, 10, 1/2}]
Out[ ] = {a^4, a^(9/2), a^5, a^(11/2), a^6, a^(13/2), a^7, a^(15/2), a^8, a^(17/2), a^9, a^(19/2), a^10}
```