Práctica 2

Gráficos 2D en mathematica

Mathematica dispone de varias instrucciones para representar gráficamente funciones, curvas o elementos geométricos en el plano.

La instrucción **`Plot`** nos permite representar la gráfica de una función, $y = f(x)$. Sin embargo, no todas las curvas del plano pueden representarse como la gráfica de una función. Por ejemplo, la circunferencia unitaria centrada en el origen viene dada por la ecuación:

$$x^2 + y^2 = 1.$$

Dicha curva no se corresponde con la gráfica de una función dado que hay dos valores de la variable y para cada valor de la variable x:

$$y = +\sqrt{1-x^2} \ \text{ e } \ y = -\sqrt{1-x^2}.$$

Si bien, en este caso, podemos considerar la circunferencia como la gráfica de dos funciones, no siempre será posible despejar la variable y en una expresión del tipo $\phi(x, y) = 0$. *Mathematica* incorpora la instrucción **`ImplicitPlot`** para la representación gráficas curvas dadas por una expresión implícita (donde la y no está despejada).

Como sabemos, la ecuación de la circunferencia también puede venir dada por las ecuaciones paramétricas,

$$x = \cos t,\ y = sen\, t, \quad t \in [0, 2\pi],$$

Para la representación gráfica de una curva dada mediante las ecuaciones paramétricas:

$$x = x(t),\ y = y(t), \quad t \in I,$$

donde I es un cierto intervalo real, utilizaremos la instrucción **`ParametricPlot`**.

1.- Representación de curvas dadas en forma explícita

Una curva en forma explícita viene dada por una ecuación del tipo $y = f(x)$, donde f es una función definida en algún subconjunto D de la recta real que se denomina *dominio de la función*. La *gráfica de una función* es el conjunto de puntos dado por

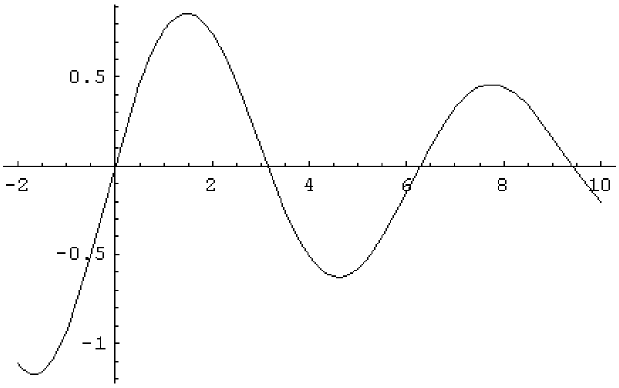
$$Gr(f) = \{(x, y) : x \in D, y = f(x)\}$$

La representación gráfica de este conjunto de puntos en un sistema de coordenadas **XY** nos proporciona, por lo general, una curva en el plano. Para representar la gráfica de una curva dada en forma explícita con el programa *Mathematica* se utiliza la instrucción **`Plot.`**

Ejemplo 1.1

Representar la gráfica de la función $y = e^{\frac{-x}{10}}\, sen\, x$ *en el intervalo* [-2,10].

```
In[1]:= Plot[Exp[-x/10] Sin[x], {x, -2, 10}]
```




```
Out[1]= - Graphics -
```

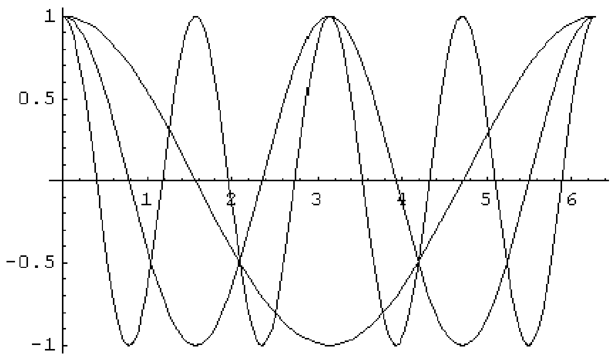
Mathematica nos permite visualizar simultáneamente la gráfica de varias funciones dadas en forma explícita: $y = f_1(x), \quad y = f_2(x), ..., y = f_n(x)$, en un mismo intervalo

---

Ejemplo 1.2

Representar en unos mismos ejes coordenados la gráfica de las funciones y = cos x, y = cos $2x$ e y = cos $4x$, *en el intervalo* $[0,2\pi]$.

```
In[2]:= Plot[{Cos[x], Cos[2 x], Cos[4 x]}, {x, 0, 2 π}]
```



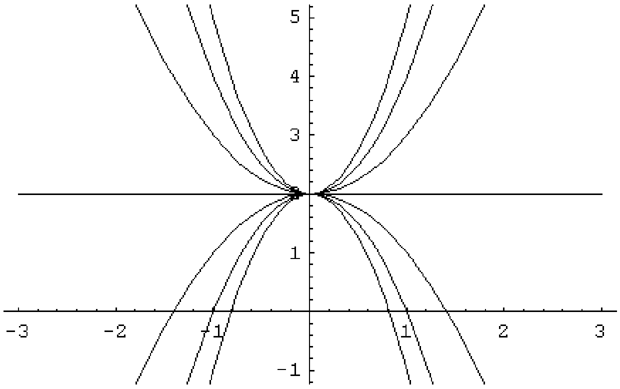

```
Out[2]= - Graphics -
```

O una familia de curvas definidas en función de un parámetro:

Ejemplo 1.3

Representar gráficamente la familia de parábolas $y = a\,x^2 + 2$, *en el intervalo* [-5,5], *para valores de a entre* -3 *y* 3.

In[3]:= `Plot[Evaluate[Table[a x^2 + 2, {a, -3, 3}]], {x, -3, 3}]`



Out[3]= `- Graphics -`

La instrucción

`Evaluate[Table[a x^2 + 2, {a, -3, 3}]]`

genera una tabla con las funciones que definen a las parábolas que queremos representar para valores enteros de **a** comprendidos entre **-3** y **3**.

`Evaluate[Table[a x^2 + 2, {a, -3, 3}]]`

{2 - 3 x^2, 2 - 2 x^2, 2 - x^2, 2, 2 + x^2, 2 + 2 x^2, 2 + 3 x^2}

2.- Representación gráfica de curvas dadas en forma paramétrica

Una curva en el plano viene dada mediante una aplicación:

$$s: I \to R^2$$
$$t \to s(t) = (x(t), y(t))$$

donde I es un intervalo de la recta real. La variable t recibe el nombre de *parámetro*. A cada valor del parámetro t le hacemos corresponder un punto del plano que denotamos por $s(t)$ y cuyas coordenadas vienen dadas por: $(x(t), y(t))$. Las ecuaciones,

$$x = x(t),\ y = y(t), \quad t \in I$$

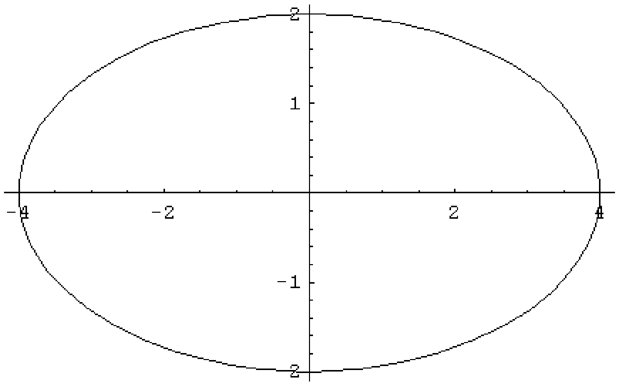
reciben el nombre de *ecuaciones paramétricas* de la curva s.

- La gráfica de una función $y = f(x)$ con $x \in D$, siempre puede expresarse en forma paramétrica mediante las ecuaciones:
$$x = t,\ y = f(t), \quad t \in D.$$
- Las ecuaciones paramétricas de una recta que pasa por el punto P(x_0, y_0) y tiene la dirección del vector $\vec{v} = (v_1, v_2)$, son:
$$x = x_0 + v_1 t,\ y = y_0 + v_2 t, \quad t \in R.$$
- Las ecuaciones paramétricas de una elipse de centro (x_0, y_0) y semiejes a y b vienen dadas por
$$x = x_0 + a\cos t,\ y = y_0 + b\,sen\,t, \quad t \in [0,2\pi].$$
En el caso particular de que $a = b = r$ se tratará de una circunferencia de centro (x_0, y_0) y radio r.

Ejemplo 2.1.

a) *Una elipse con centro el origen y de semiejes* 4 *y* 2.

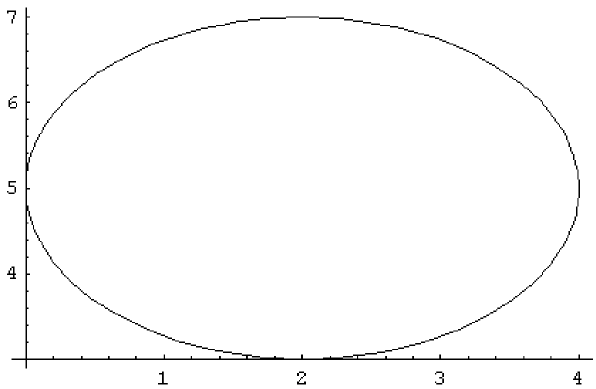
```
In[5]:= ParametricPlot[{4 Cos[t], 2 Sin[t]}, {t, 0, 2 π}]
```




```
Out[5]= - Graphics -
```

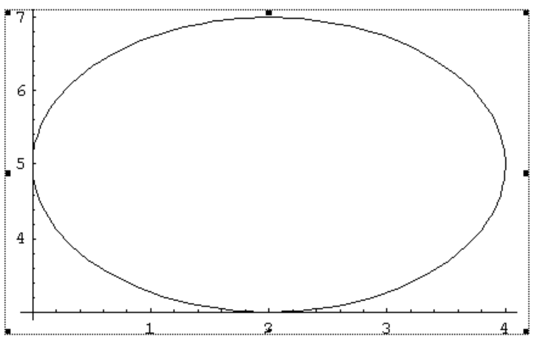
b) *Una circunferencia con centro el punto* (2,5) *y de radio* 2.

```
In[6]:= ParametricPlot[{2 + 2 Cos[t], 5 + 2 Sin[t]}, {t, 0, 2 π}]
```




```
Out[6]= - Graphics -
```

Observemos que la gráfica mostrada por *Mathematica* no parece una circunferencia (más bien, parece una elipse). Esto es debido a que, como puede comprobarse, la escala utilizada en ambos ejes coordenados no es la misma. *Mathematica* dibuja todas las gráficas en un rectángulo áureo, es decir, un rectángulo en el que la relación entre la anchura y la altura viene dada por el número de oro:



$$\Phi = \frac{1+\sqrt{5}}{2} = 1.61803$$

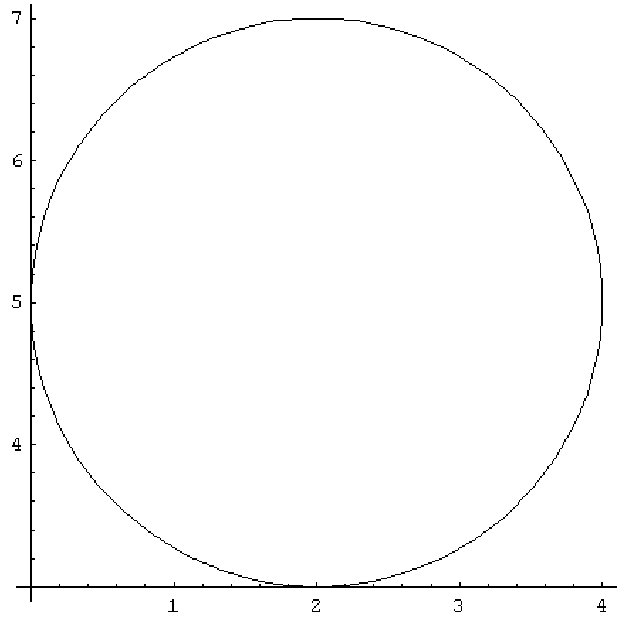
Relación entre la anchura y la altura del rectángulo

Sin embargo podemos cambiar este rectángulo mediante la opción:

`AspectRatio→ r`,

donde **`r`** indica el cociente entre la altura y la anchura del rectángulo donde deseamos dibujar el gráfico. Si utilizamos la opción **`AspectRatio→Automatic`**, las unidades del eje OX son tomadas igual que las del eje OY. Puede decirse que con la opción **`Automatic`** obtenemos la "forma verdadera" de la gráfica.

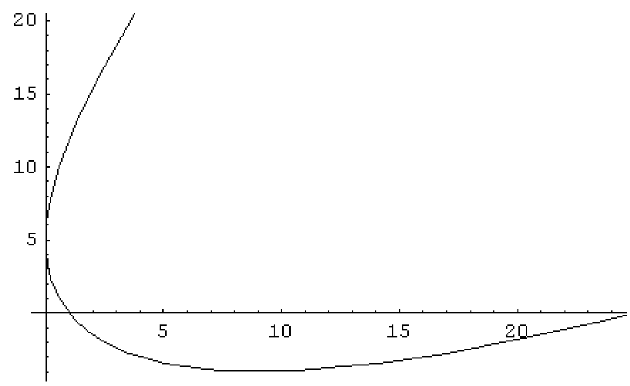
```
In[7]:= ParametricPlot[{2 + 2 Cos[t], 5 + 2 Sin[t]}, {t, 0, 2 π},
         AspectRatio → Automatic]
```




```
Out[7]= - Graphics -
```

c) *Una parábola*

```
In[8]:= ParametricPlot[{t^2 - 2 t + 1, t^2 + 4 t}, {t, -5, 4}]
```



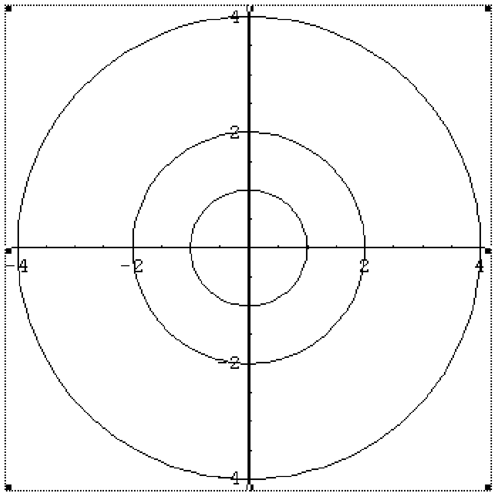

```
Out[8]= - Graphics -
```

De forma análoga a la instrucción **`Plot`** también podemos representar un conjunto de curvas expresadas en forma paramétrica en el mismo sistema de coordenadas, en función del mismo parámetro y en el mismo rango de variación del parámetro.

Ejemplo 2.2

a) *Dibujar tres círculos concéntricos en el origen de radios* 1, 4 y 6.

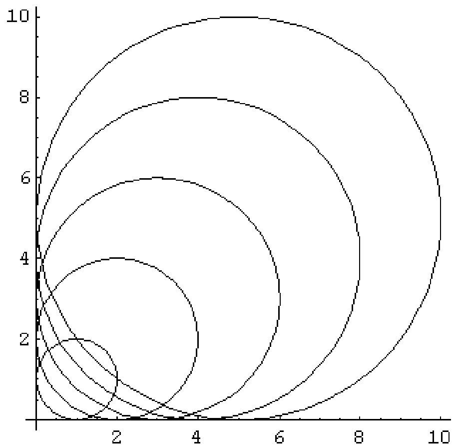
```
In[9]:= ParametricPlot[{{Cos[t], Sin[t]}, {2 Cos[t], 2 Sin[t]},
          {4 Cos[t], 4 Sin[t]}}, {t, 0, 2 π}, AspectRatio → Automatic]
```




```
Out[9]= - Graphics -
```

b) *Representar la familia de circunferencias que tienen su centro en la bisectriz del primer cuadrante y son tangentes a los ejes coordenados.*

```
In[10]:= ParametricPlot[
           Evaluate[Table[{a + a Cos[t], a + a Sin[t]}, {a, 1, 5}]],
           {t, 0, 2 π}, AspectRatio → 1]
```




```
Out[10]= - Graphics -
```

3.- Representación gráfica de curvas dadas en forma implícita.

Mathematica permite también representar la gráfica de una curva dada en forma implícita mediante una ecuación del tipo $\Phi(x, y) = 0$, es decir, donde la variable *y* no está dada en función de la variable *x*. Para ello utilizaremos la instrucción **`ImplicitPlot.`** Sin embargo, a diferencia de las instrucciones anteriores (**`Plot`** y **`ParametricPlot`**) que se cargan automáticamente en el núcleo (**Kernel**) del programa *Mathematica*, para utilizar esta instrucción hemos de cargar expresamente en el núcleo el paquete (**package**), **`Graphics`ImplicitPlot`**`.
Para cargar este paquete utilizamos la instrucción:

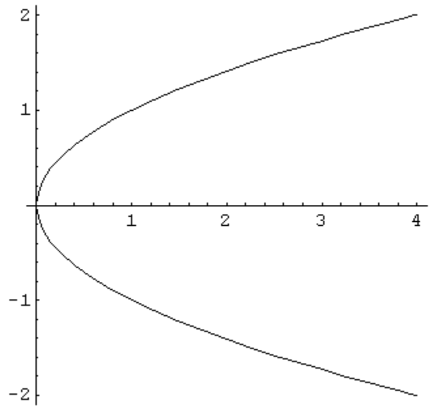
```
In[11]:= << Graphics`ImplicitPlot`
```

A partir de este momento la instrucción **`ImplicitPlot`** es reconocida por el programa y podemos utilizarla.

Ejemplo 3.1

a) *Representar la parábola* $x = y^2$ *para valores de x en el intervalo* [0,4].

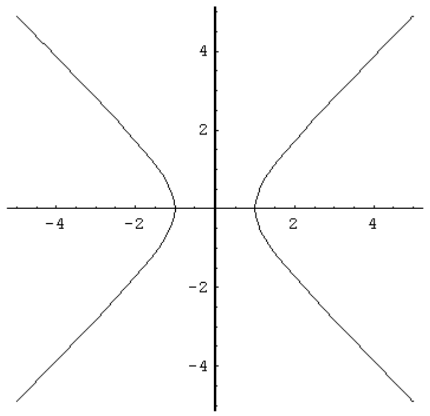
```
In[12]:= ImplicitPlot[x == y^2, {x, 0, 4}]
```




```
Out[12]= - Graphics -
```

b) *Representar la hipérbola equilátera de ecuación* $x^2 - y^2 = 1$, *para valores de x en el intervalo* [-5,5].

```
In[13]:= ImplicitPlot[x^2 - y^2 == 1, {x, -5, 5}]
```



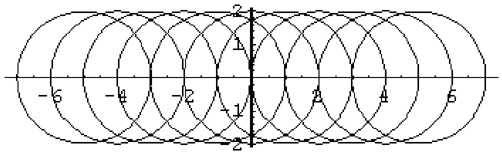

```
Out[12]= - Graphics -
```

También podemos dibujar varias curvas o una familia de curvas dadas en forma implícita:

Ejemplo 3.2

Dibujar una serie de circunferencias de radio 2 *con centro en el eje* OX

```
In[14]:= ImplicitPlot[Evaluate[Table[(x - a)^2 + y^2 == 4, {a, -5, 5}],
         {x, -8, 8}, {y, -2, 2}, AspectRatio → Automatic]]
```




```
Out[14]= - Graphics -
```

4.- Representación gráfica de un conjunto finito de puntos en el plano.

En la práctica, cuando trabajamos con datos o medidas obtenidas experimentalmente, suele ser habitual que no conozcamos una fórmula explícita de nuestra función sino una serie de valores y_i en determinados puntos x_i, i=1,…,n. Se dice entonces que la función está dada mediante la tabla de valores $\{(x_i, y_i)\}_{i=1,2,\ldots,n}$.

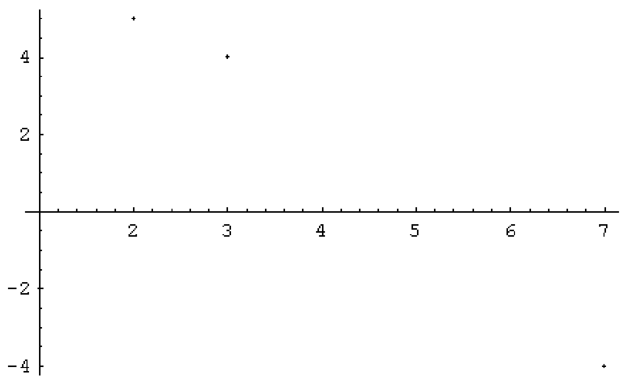
El programa *Mathematica* permite la representación gráfica de estos puntos mediante la instrucción:

ListPlot[{{x_1,y_1},{x_2,y_2},…,{x_n,y_n}}]

Ejemplo 4.1

Dibujar la tabla de puntos {(1,-2), (2,5), (3,4), (7,-4)}

```
In[15]:= ListPlot[{{1, -2}, {2, 5}, {3, 4}, {7, -4}}]
```

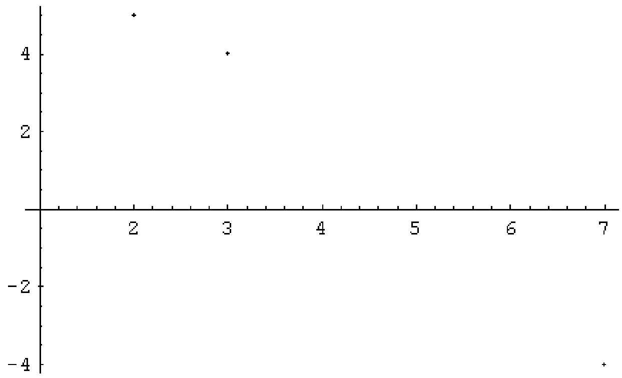



```
Out[15]= - Graphics -
```

- En ocasiones puede resultar aconsejable almacenar la tabla de valores en una variable. Esto nos permitirá volver a utilizar la misma tabla sin necesidad de tener que volver a escribirla.

```
In[16]:= puntos = {{1, -2}, {2, 5}, {3, 4}, {7, -4}};
```

```
In[17]:= ListPlot[puntos]
```



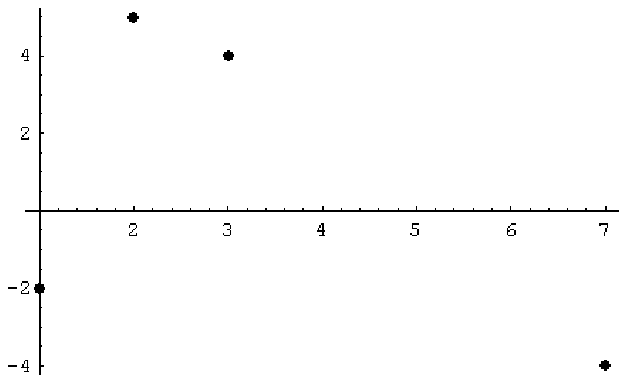
```
Out[17]= - Graphics -
```

- Podemos conseguir agrandar el tamaño de los puntos para visualizarlos mejor, mediante la opción:

  `PlotStyle→PointSize[`*tamañodepunto***`]`**,

  donde *tamañodepunto* es un número entre **`0`** y **`1`**, que indica el porcentaje del tamaño del punto respecto del tamaño total del dibujo. Por ejemplo: **`0.02`** significa que el tamaño del punto será el 2% del tamaño total del gráfico.

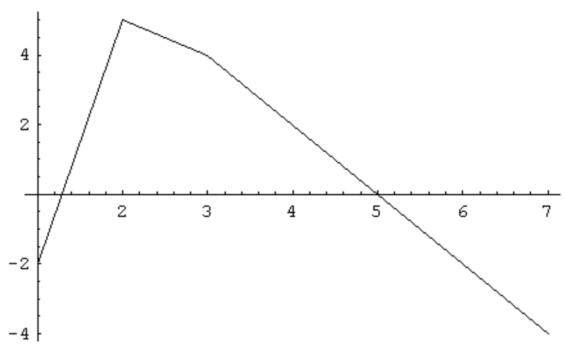
```
In[18]:= ListPlot[puntos, PlotStyle → PointSize[0.02]]
```



```
Out[18]= - Graphics -
```

El programa *Mathematica* también nos permite dibujar la gráfica de la poligonal obtenida al unir los puntos, mediante la opción: **`PlotJoined→True`**

```
In[19]:= ListPlot[puntos, PlotJoined → True]
```



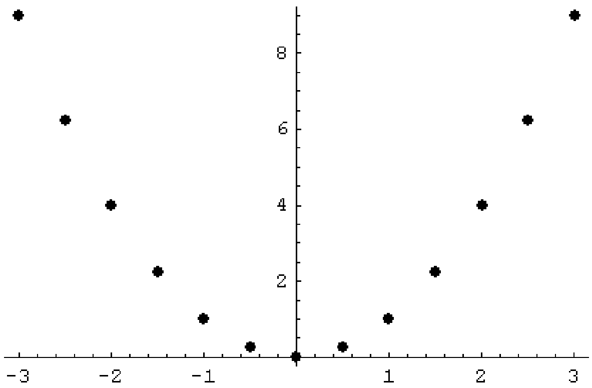
```
Out[19]= - Graphics -
```

En el siguiente ejemplo generamos una tabla de puntos sobre la parábola $y = 5x^2$ usando la instrucción **Table.**

Ejemplo 4.2

Representar la tabla de valores $\{(k, k^2)\}$ *para valores de k entre* -3 y 3 *con un incremento de* 0.5.

```
In[20]:= datos = Table[{k, k^2}, {k, -3, 3, 0.5}];
         ListPlot[datos, PlotStyle → PointSize[0.02]]
```



```
Out[21]= - Graphics -
```

5.- Combinando gráficos: La instrucción Show.

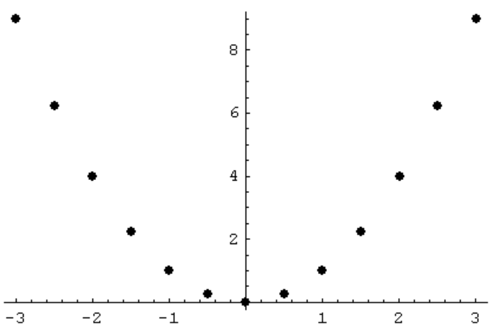
La instrucción **`Show`** permite visualizar un gráfico ya existente o visualizar simultáneamente varios gráficos previamente creados, aún cuando se trate de gráficos de diferentes tipos. Esto es posible porque *Mathematica* guarda la información de cualquier gráfico realizado con anterioridad, de forma que se puede volver a visualizar usando la instrucción **`Show`**.

> **IMPORTANTE**: Si dibujamos una gráfica que posteriormente tenemos intención de volver a utilizar es aconsejable guardarla en una variable asignándole un nombre apropiado que nos permita referirnos a ella con facilidad.

Ejemplo 4.2

Visualizar la gráfica la tabla de puntos creada en el ejemplo anterior junto con la gráfica de la parábola $y = x^2$.

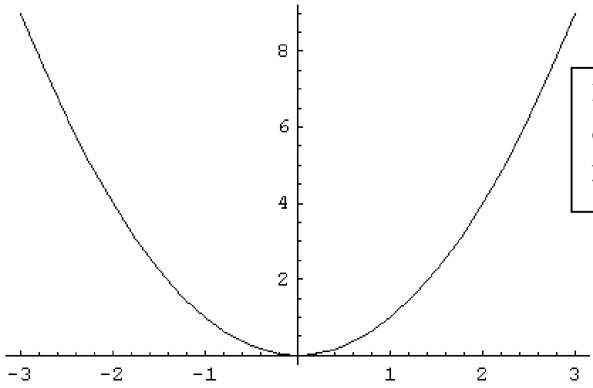
```
In[22]:= g1 = ListPlot[datos, PlotStyle → PointSize[0.02]]
```



> Representamos la gráfica de los puntos y la guardamos con el nombre **`g1`**.

```
Out[22]= - Graphics -
```

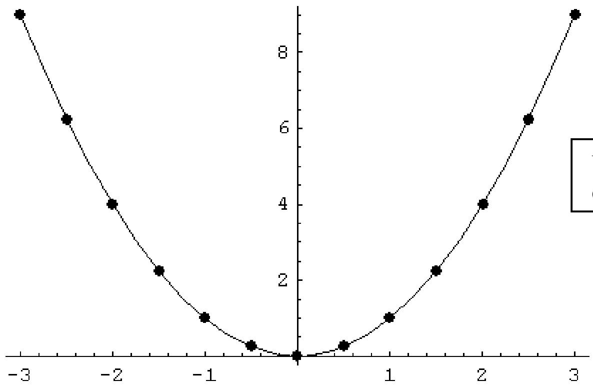
In[23]:= `g2 = Plot[x^2, {x, -3, 3}]`



Representamos la gráfica de la función $y = x^2$ en el intervalo [-3,3] y la guardamos en la variable **g2**.

Out[23]= `- Graphics -`

In[24]:= `Show[g1, g2]`



Visualizamos ambas gráficas con la instrucción **Show**.

Out[24]= `- Graphics -`

6.- Mejorando nuestros gráficos

A continuación damos una lista de opciones que podemos utilizar con cualquier da las instrucciones **Plot**, **ListPLot**, **ParametricPlot** y **Show**, y el valor asignado por defecto.

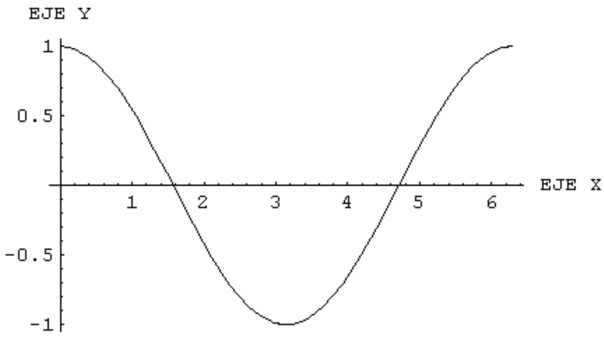
AspectRatio→1/GoldenRatio	Indica la razón entre la altura y la anchura del rectángulo donde se mostrará un gráfico.
Axes→True	Determina si se han de dibujar los ejes
AxesLabel→None	Proporciona rótulos para los ejes
AxesOrigin→Automatic	Determina el punto donde se ha de colocar el origen de coordenadas.
AxesStyle→Automatic	Especifica las opciones para el estilo de los ejes.
Background→Automatic	Selecciona el color de fondo del gráfico
DefaultColor→Automatic	Color por omisión de los elementos del gráfico
Frame→False	Determina si el gráfico se realiza con marco
FrameLabel→None	Especifica los rótulos del marco

FrameStyle→Automatic	Especifica el estilo del marco
FrameTicks→Automatic	Para marcas en el marco
GridLines→None	Para trazar rejillas
PlotLabel→None	Rótulo para el gráfico
PlotRange→Automatic	Rango de valores que se van a incluir
PlotRegion→Automatic	Indica la región que se va a rellenar
RotateLabel→True	Determina si se han de girar los rótulos
Ticks→Automatic	Indica en qué puntos del eje OX y del eje OY se van a escribir las marcas en los ejes

Ejemplo 4.2

a) *La opción* **AxesLabel**

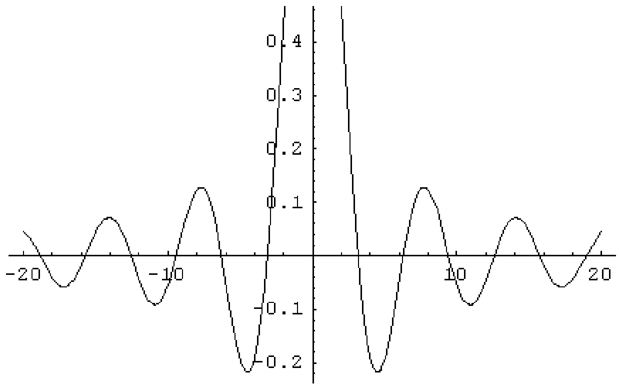
```
In[25]:= Plot[Cos[x], {x, 0, 2 Pi},
          AxesLabel → {"EJE X", "EJE Y"}]
```



Out[25]= - Graphics -

b) *La opción* **PlotRange→All**

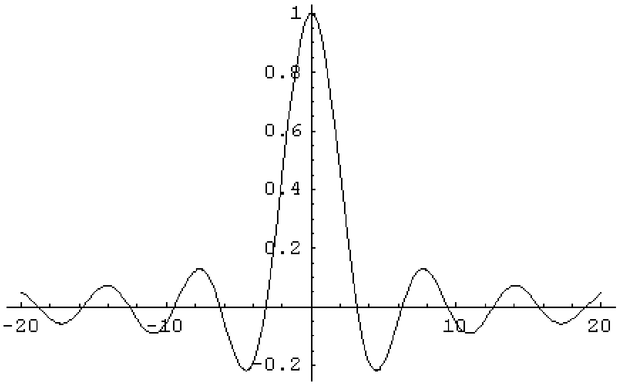
```
In[26]:= Plot[Sin[x] / x, {x, -20, 20}]
```



Out[26]= - Graphics -

Si queremos visualizar la gráfica completa hemos de modificar el rango de visualización mediante la opción **PlotRange→All**

```
In[27]:= Plot[Sin[x] / x, {x, -20, 20}, PlotRange → All]
```



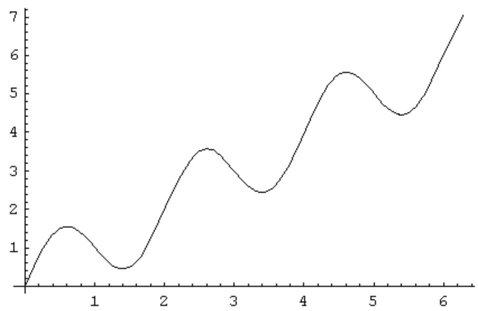

```
Out[27]= - Graphics -
```

c) *La opción* **DisplayFunction**

```
In[28]:= g1 = Plot[x + Sin[π x], {x, 0, 2 π},
              DisplayFunction → Identity];
```

Con la opción **DisplayFunction→Identity** generamos el gráfico pero no éste no se muestra en pantalla. Cuando queramos visualizarlo podemos utilizar la instrucción **Show**

```
In[29]:= Show[g1, DisplayFunction → $DisplayFunction]
```



Observemos que hemos utilizado la opción:

DisplayFunction→$DisplayFunction

para poder visualizar el gráfico.

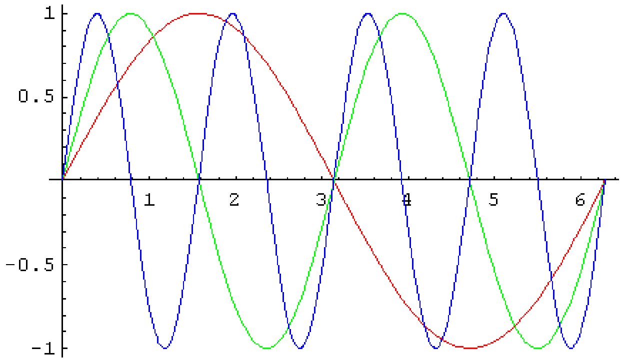
d) *La opción* **RGBColor**

Cuando visualizamos varias gráficas simultáneamente puede resultar útil dibujar cada una de las gráficas con un color distinto. Esto se consigue con la opción:

RGBColor[c1,c2,c3].

Los argumentos **c1,c2,c3** pueden tomar un valor comprendido entre 0 y 1 e indican el porcentaje de rojo, verde y azul que se utilizarán para formar nuestro color.

```
In[30]:= Plot[{Sin[x], Sin[2 x], Sin[4 x]}, {x, 0, 2 π},
           PlotStyle → {RGBColor[1, 0, 0], RGBColor[0, 1, 0],
             RGBColor[0, 0, 1]}]
```



Out[30]= - Graphics -

7.- Representación gráfica de funciones definidas a trozos.

Supongamos que queremos representar la gráfica de la función

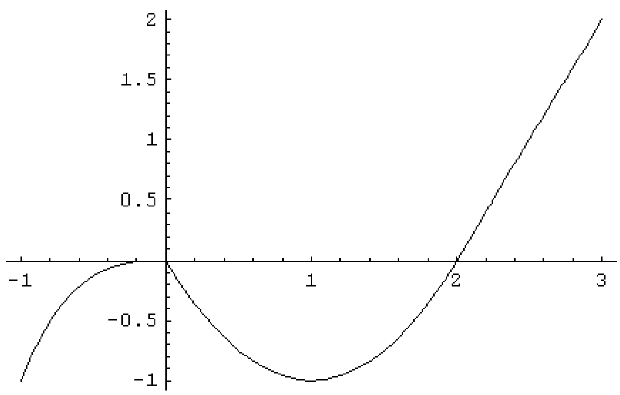
$$f(x)=\begin{cases} x^3 & , \quad x<0 \\ x^2-2x & , \quad 0\le x<2 \\ 2x-4 & , \quad x\ge 2 \end{cases} \quad \text{en el intervalo } [-1,3]$$

Una primera opción para ello sería representar la gráfica de cada uno de los trozos que forman la función f y a continuación utilizar la instrucción **Show** para visualizar toda la gráfica.

```
In[31]:= g1 = Plot[x^3, {x, -1, 0}, DisplayFunction → Identity];
         g2 = Plot[x^2 - 2 x, {x, 0, 2}, DisplayFunction → Identity];
         g3 = Plot[2 x - 4, {x, 2, 3}, DisplayFunction → Identity];
```

Observemos que hemos utilizado la opción **DisplayFunction→Identity** para generar cada uno de los gráficos pero sin mostrarlos en pantalla. Ahora podemos utilizar la instrucción **Show,** con la opción **DisplayFunction→Identity,** para visualizar la gráfica completa de la función

```
In[34]:= Show[g1, g2, g3, DisplayFunction → $DisplayFunction]
```



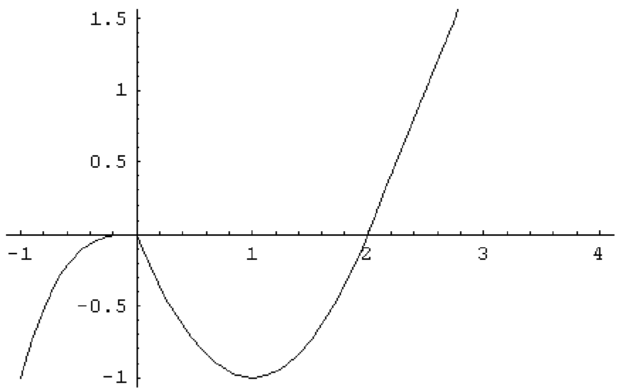
Out[34]= - Graphics -

Otra forma de representar la gráfica de la función *f* es utilizando la instrucción **`Which`** que nos permite definir una función a trozos:

```
In[35]:= f[x_] := Which[x < 0, x^3, 0 ≤ x < 2, x^2 - 2 x, x ≥ 2, 2 x - 4]
```

Ahora podemos utilizar la instrucción **`Plot`** para dibujar su gráfica en el intervalo [-1,4]

```
In[36]:= Plot[f[x], {x, -1, 4}]
```

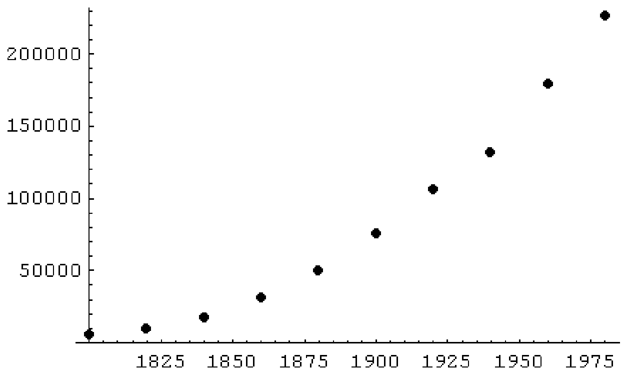



```
Out[36]= - Graphics -
```

8.- Un poco más sobre gráficos

Mathematica no siempre dibuja la intersección de los ejes coordenados en el punto (0,0), veámoslo con el siguiente ejemplo:

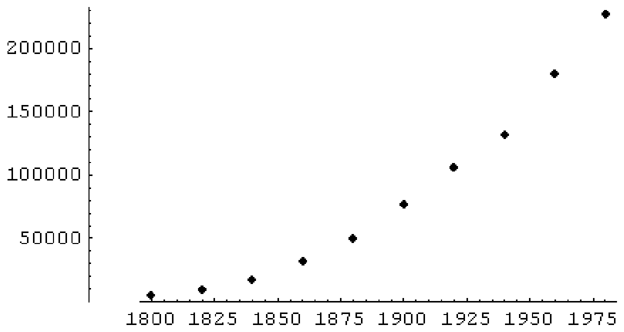
```
In[37]:= datos = {{1800, 5308}, {1820, 9638}, {1840, 17069},
            {1860, 31443}, {1880, 50156}, {1900, 75995},
            {1920, 105711}, {1940, 131669}, {1960, 179323},
            {1980, 226505}};

In[38]:= ListPlot[datos, PlotStyle → PointSize[0.02]]
```




```
Out[38]= - Graphics -
```

Sin embargo, podemos situar el punto de intersección de los ejes donde a nosotros nos interese:

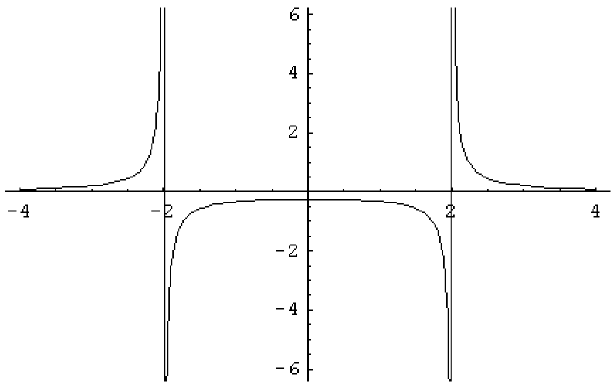
In[39]:= ListPlot[datos, PlotStyle → PointSize[0.02],
    AxesOrigin → {1775, 0}]



Out[39]= - Graphics -

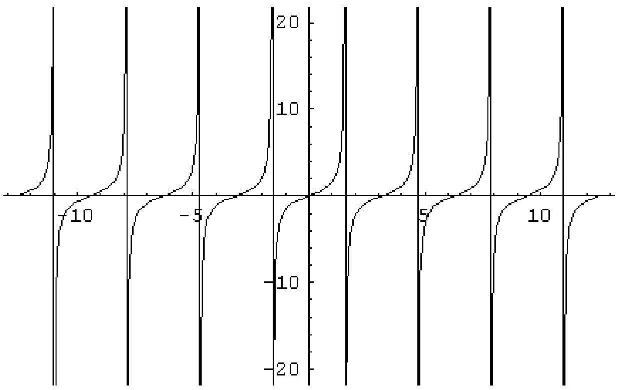
Si una función $y = f(x)$ presenta una asíntota horizontal en un $x = a$, el programa *Mathematica* la localiza automáticamente.

In[40]:= Plot[$\frac{1}{x^2 - 4}$, {x, -4, 4}]



Out[40]= - Graphics -

In[41]:= Plot[Tan[x], {x, -4 π, 4 π}]



Out[41]= - Graphics -

9.- Ejercicios propuestos

1. Dibuja en unos mismos ejes coordenados las gráficas de las funciones $y = x$, $y = \ln x$ e $y = e^x$, dibujando cada gráfica de un color diferente.
2. Dibuja, en unos mismos ejes coordenados, la gráfica de la parábola $y = x^2 - 5x$ y la de su recta tangente en el punto $x = 5$.
3. Representa la gráfica de la espiral $x = t\,sent,\ \ y = t\cos t,$ para valores del parámetro $t \in [0,4\pi]$.
4. Dibuja la gráfica del astroide dado por la ecuación implícita $|x|^{2/3} + |y|^{2/3} = 1$ para los valores $-1 \le x \le 1,\ \ -1 \le y \le 1$.
5. Representar la familia de circunferencias que tienen su centro en el eje OX y que son tangentes al eje OY.