

# **MATHEMATICA**

## **Gráficas de funciones y objetos gráficos**

**Ricardo Villafaña Figueroa**

## Contenido

Gráficas de funciones en dos dimensiones .....	3
Combinando gráficas (función Show) .....	5
Opciones para las gráficas de funciones .....	7
Opción PlotRange (ajustar el rango del eje-Y).....	7
Opción GridLines (rejillas en las gráficas).....	10
Opción Ticks (marcas sobre el eje-X y el eje-Y).....	12
Opción AspectRatio (escalas entre el eje-X y eje-Y).....	14
Opción PlotStyle (color y estilo de las gráficas) .....	16
Opción Filling (llenado de curvas) .....	18
Llenado a diferentes niveles.....	18
Llenando múltiples curvas.....	20
Llenado entre regiones .....	22
Gráficas Polares.....	23
Trébol de tres hojas.....	23
Trébol de cuatro hojas .....	24
Espiral .....	25
Espiral de Arquímedes .....	25
Cardioides .....	26
Rosa de n pétalos .....	27
Objetos gráficos .....	28
Puntos y líneas .....	28
Círculos, arcos y sectores de círculo.....	30
Rotación de objetos gráficos.....	37

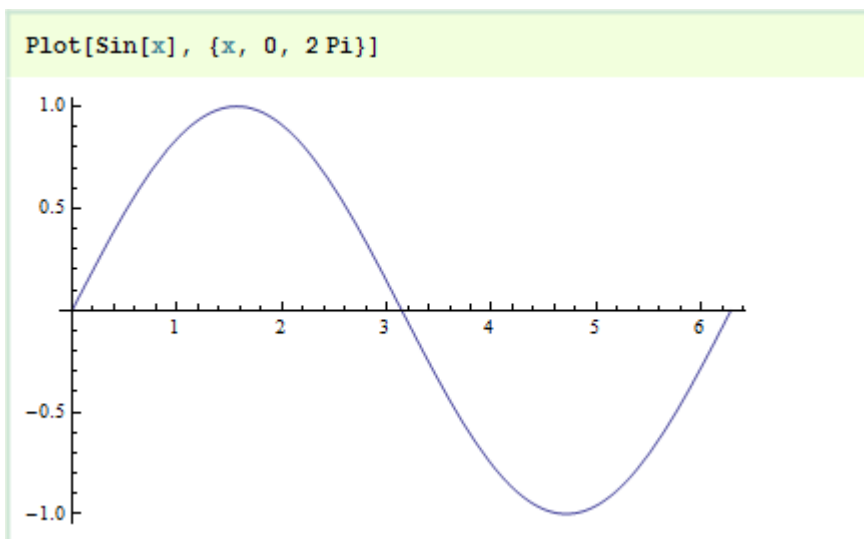
## GRÁFICAS DE FUNCIONES EN DOS DIMENSIONES

### Ejemplo

Graficar la función  $\sin(x)$  en el rango de  $0..2\pi$

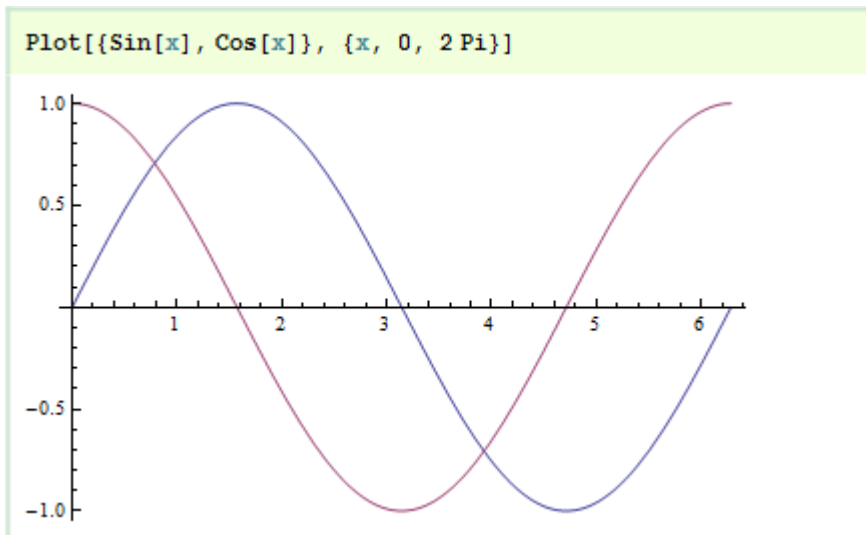
### Solución

Graficando la función



**Ejemplo**

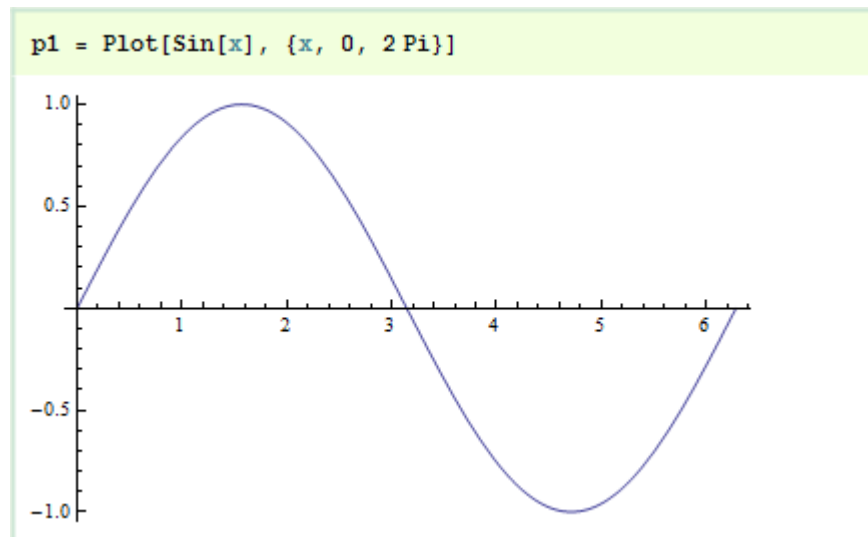
Graficar las funciones  $\sin(x)$ ,  $\cos(x)$  en el rango de  $0..2\pi$

**Solución**

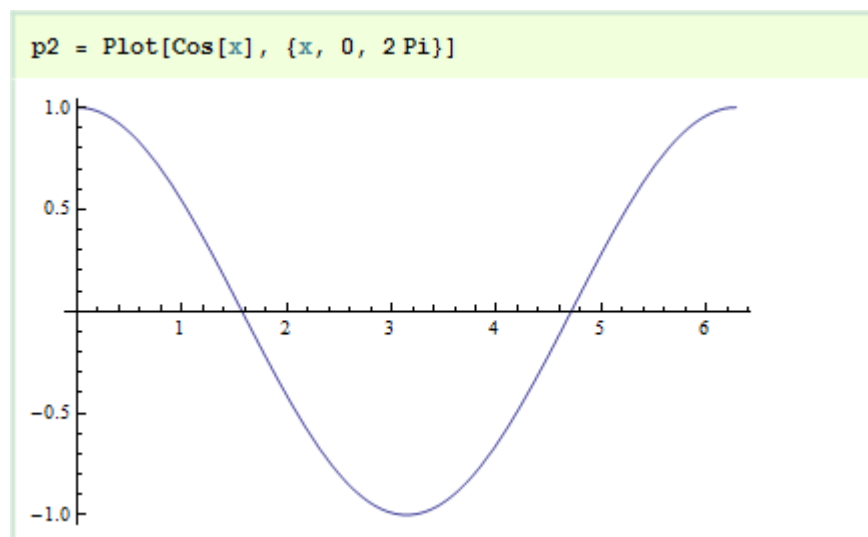
## COMBINANDO GRÁFICAS (FUNCIÓN SHOW)

### Ejemplo

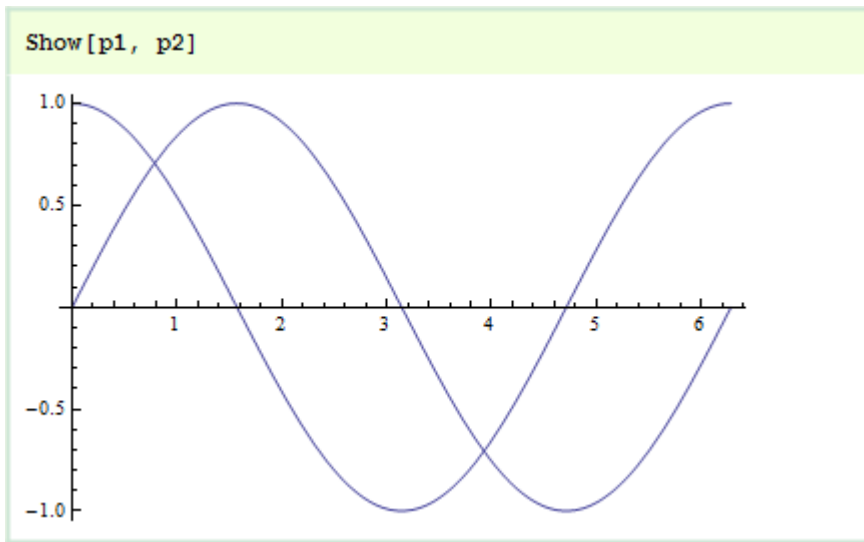
Definiendo la primera gráfica:



Definiendo la segunda gráfica:



Combinando gráficas con la función Show:



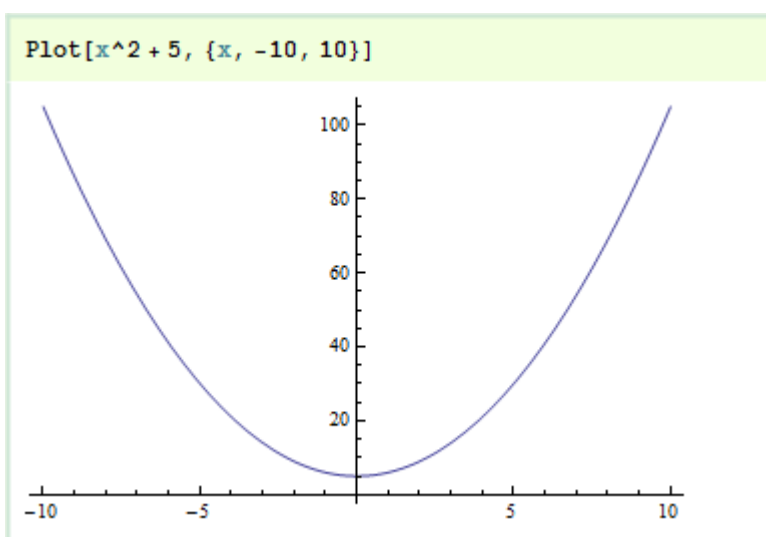
## OPCIONES PARA LAS GRÁFICAS DE FUNCIONES

### Opción PlotRange (ajustar el rango del eje-Y)

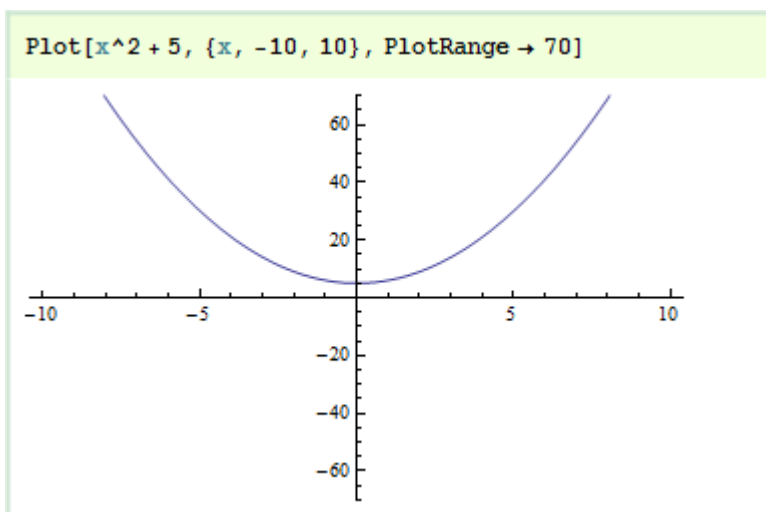
#### Ejemplo

Graficar la función  $x^2 + 5$

Primera aproximación:



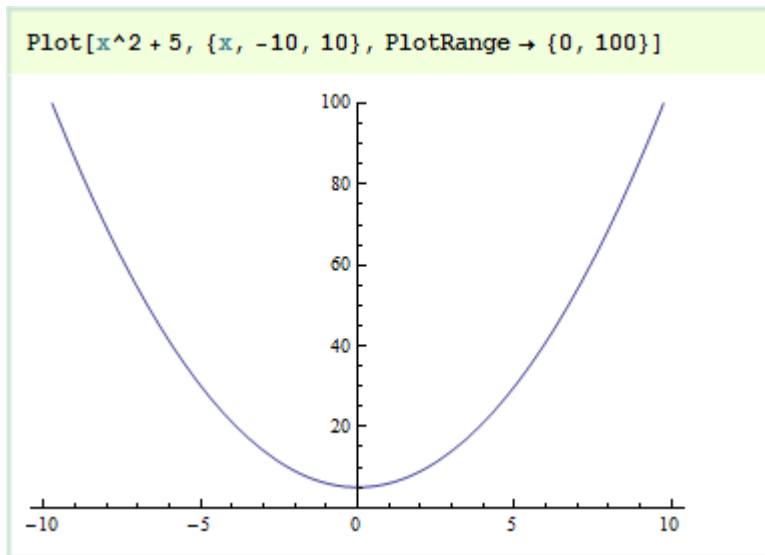
Segunda aproximación:







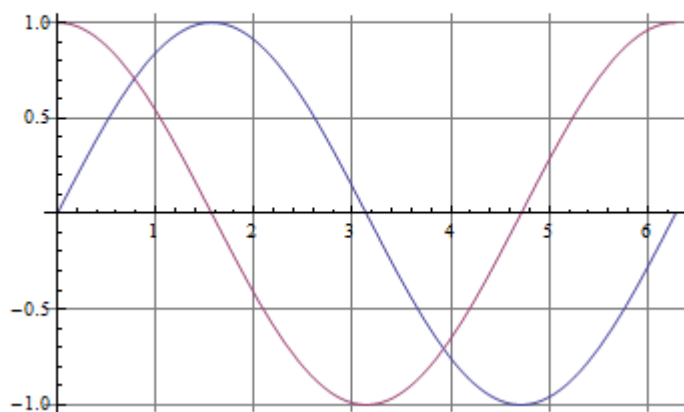
Tercera aproximación:



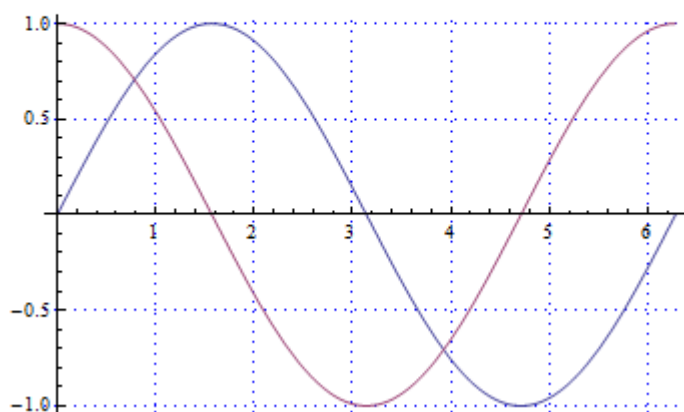
## Opción GridLines (rejillas en las gráficas)

### Ejemplo

```
Plot[{Sin[x], Cos[x]}, {x, 0, 2 Pi},  
GridLines -> Automatic]
```



```
Plot[{Sin[x], Cos[x]}, {x, 0, 2 Pi},  
GridLines -> Automatic,  
GridLinesStyle -> Directive[Blue, Dotted]]
```

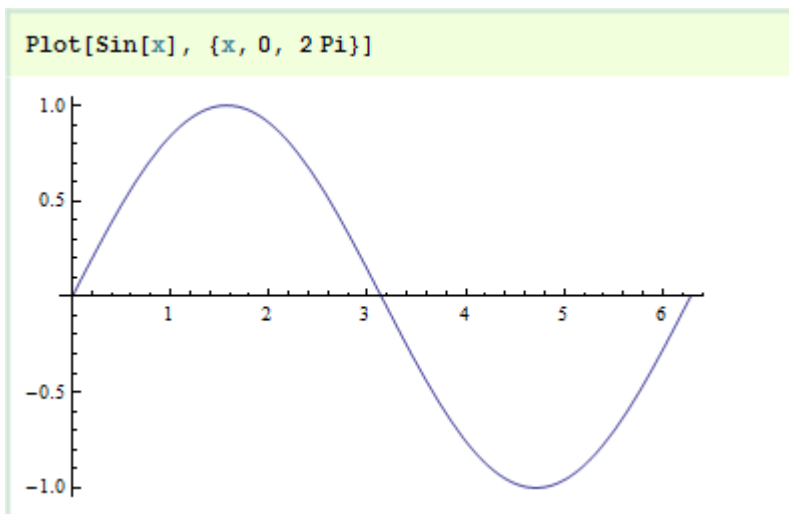




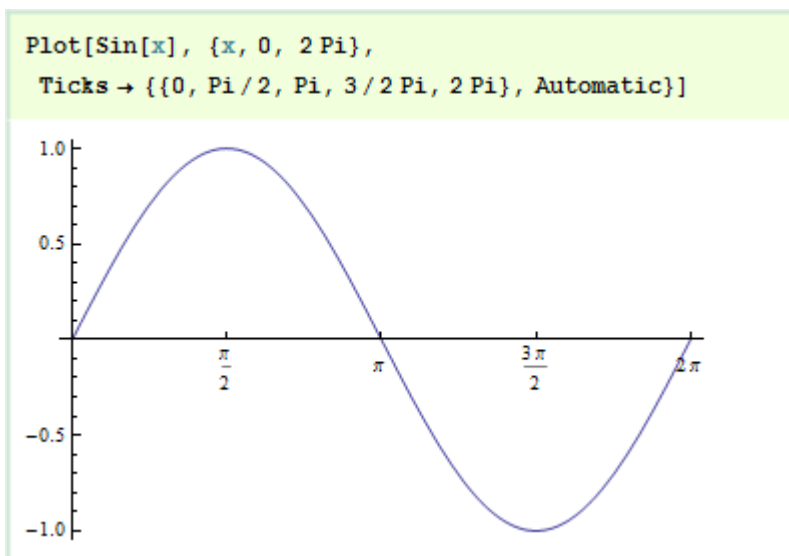
## Opción Ticks (marcas sobre el eje-X y el eje-Y)

### Ejemplo

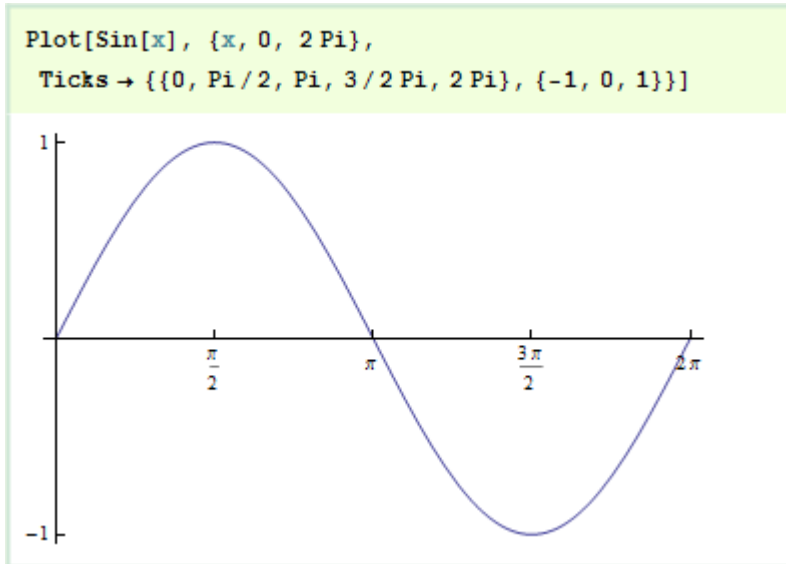
Primera aproximación (etiquetas por omisión):



Segunda aproximación (etiquetando el eje-X):



Tercera opción (etiquetando el eje-X y el eje-Y):



## Opción AspectRatio (escalas entre el eje-X y eje-Y)

### Ejemplo

Graficar la circunferencia  $x^2 + y^2 = 25$

### Solución

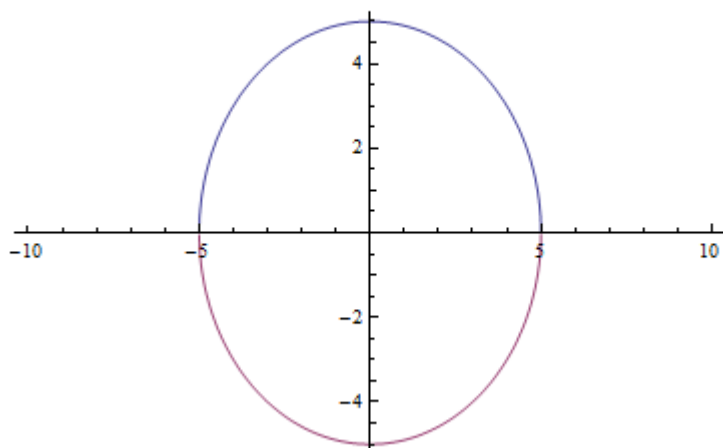
Definir dos funciones (parte superior e inferior de la circunferencia):

$$f1 = \sqrt{25 - x^2} ;$$

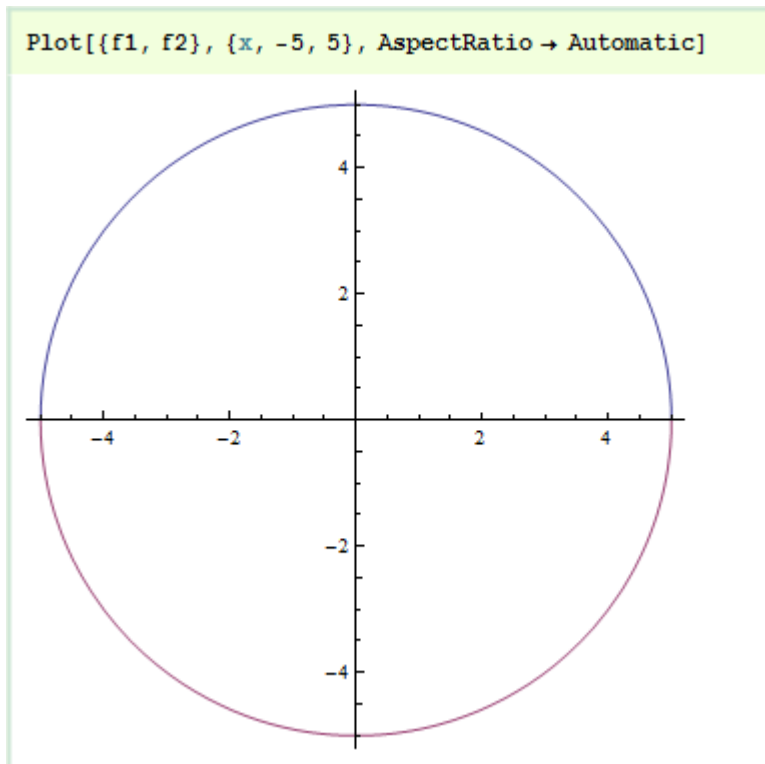
$$f2 = -\sqrt{25 - x^2} ;$$

Primera aproximación:

```
Plot[{f1, f2}, {x, -10, 10}]
```

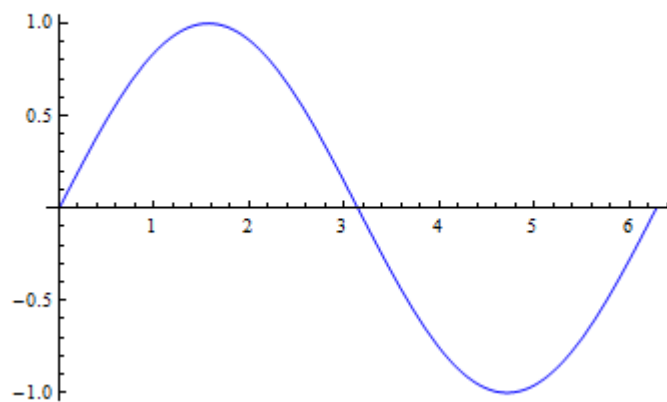


Segunda aproximación (usando la opción AspectRatio -> Automatic):

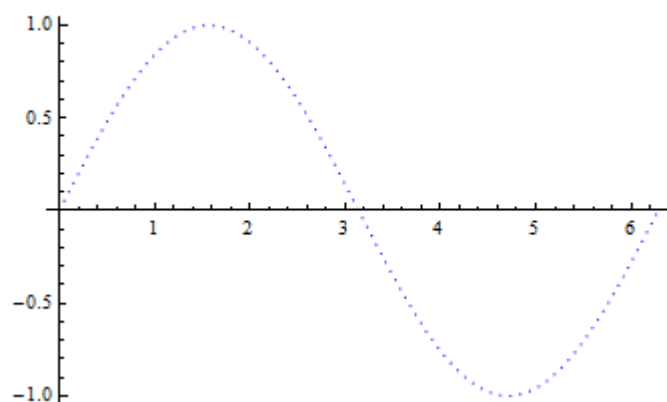


## Opción PlotStyle (color y estilo de las gráficas)

```
Plot[Sin[x], {x, 0, 2 Pi}, PlotStyle -> Blue]
```

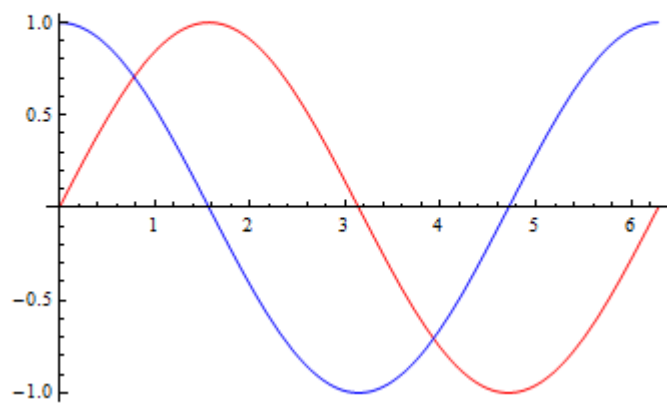


```
Plot[Sin[x], {x, 0, 2 Pi}, PlotStyle -> {Blue, Dotted}]
```

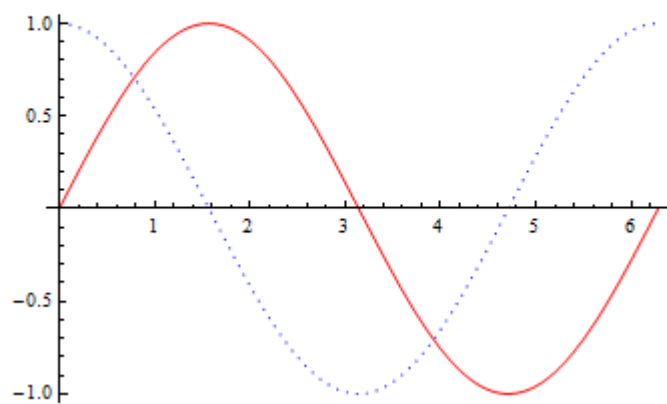




```
Plot[{Sin[x], Cos[x]}, {x, 0, 2 Pi},  
PlotStyle -> {Red, Blue}]
```



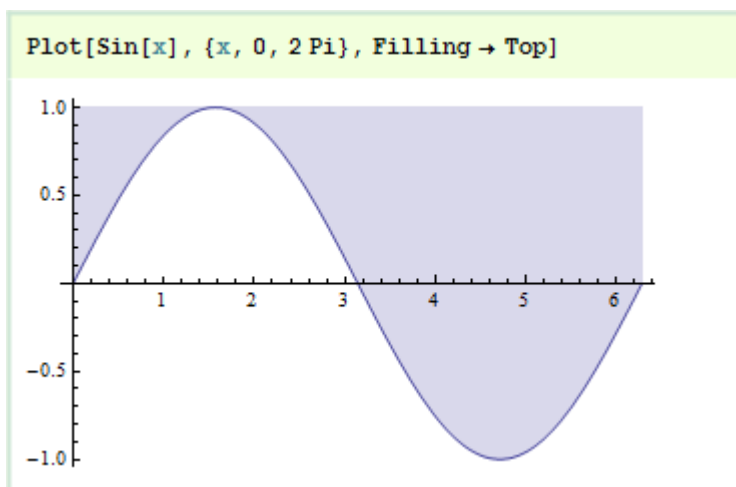
```
Plot[{Sin[x], Cos[x]}, {x, 0, 2 Pi},  
PlotStyle -> {Red, {Blue, Dotted}}]
```



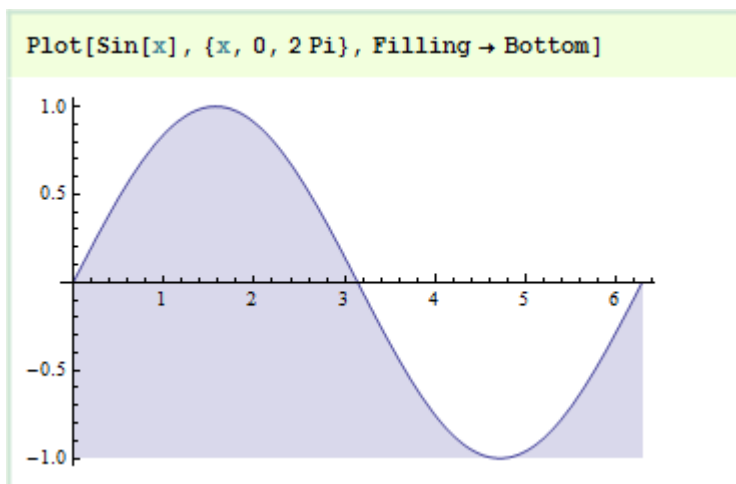
## Opción Filling (llenado de curvas)

### Llenado a diferentes niveles

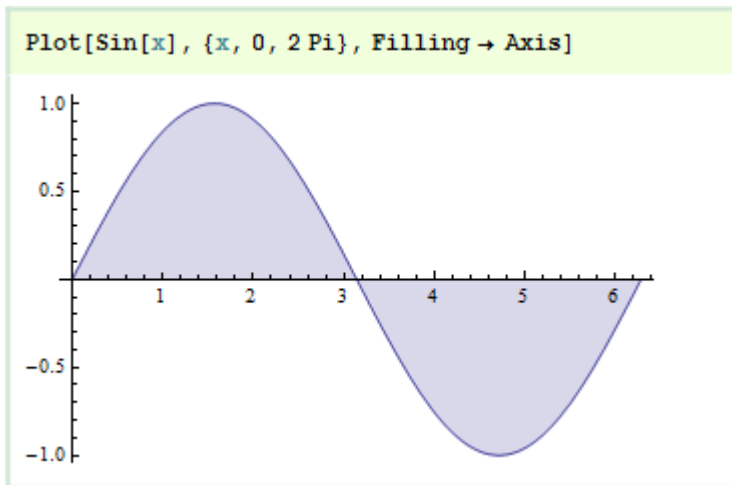
Arriba:



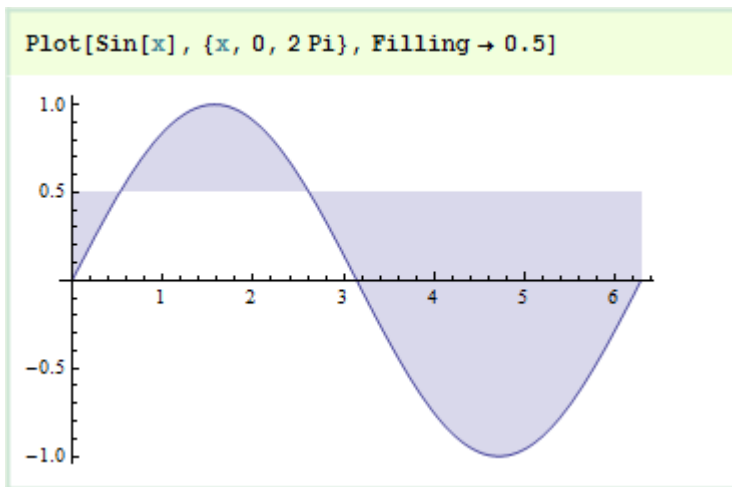
Abajo:



Entre los ejes:

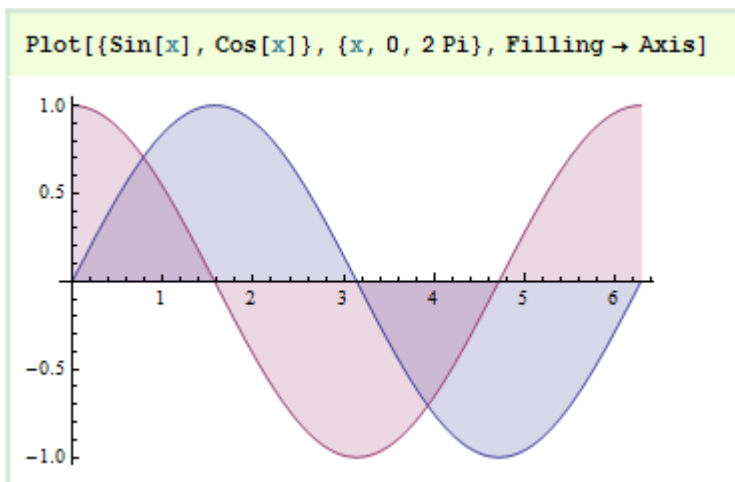


A partir de cierto valor del eje-Y:

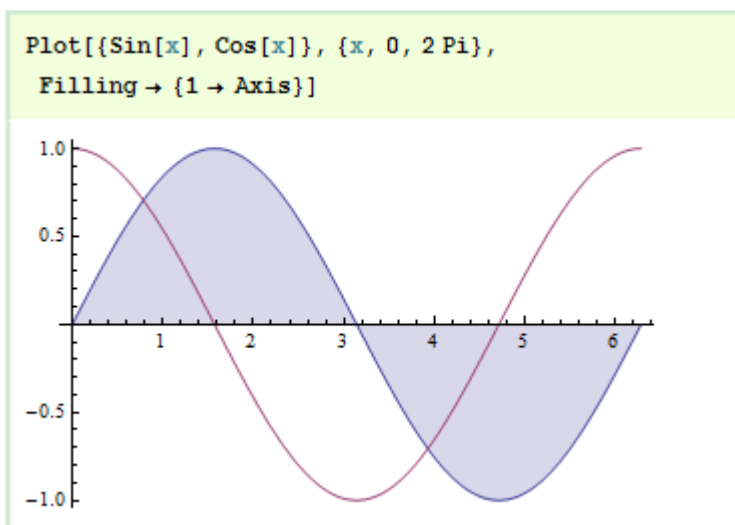


## Llenando múltiples curvas

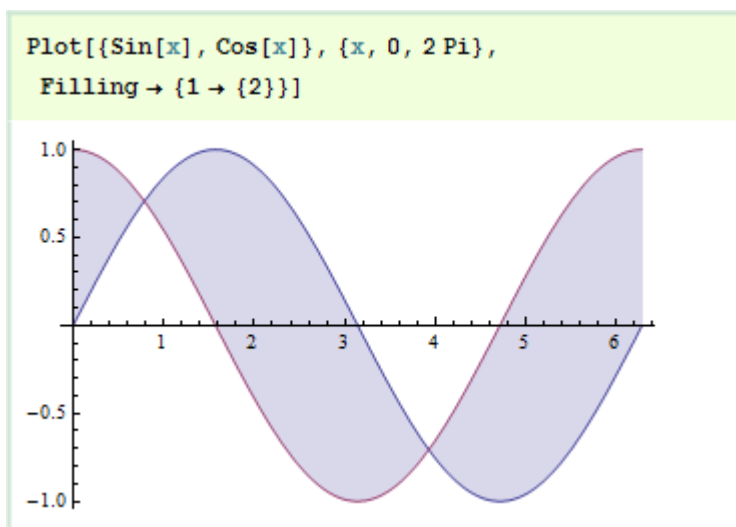
Curvas que solapan:



Llenando entre la curva 1 y el eje-X:

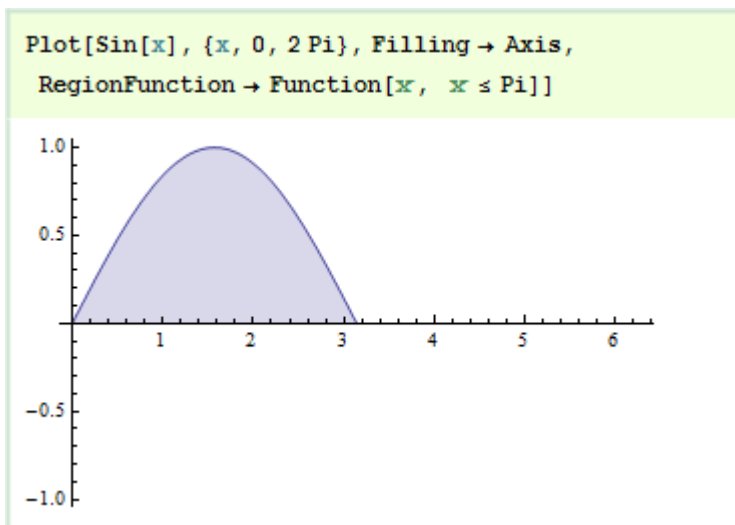


Llenado entre la curva 1 y 2:

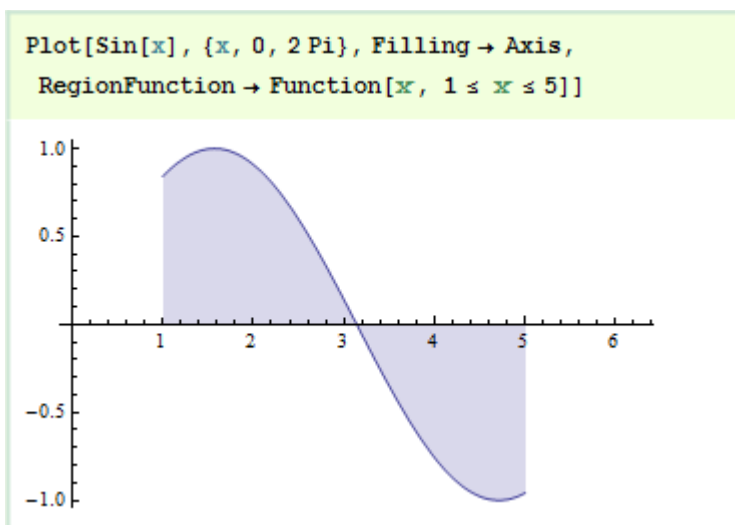


## Llenado entre regiones

Llenado de para  $x \leq \pi$ :



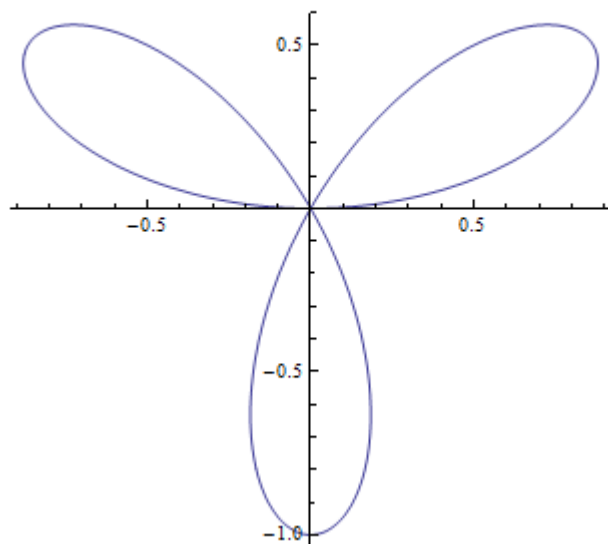
Llenado para  $1 \leq x \leq 5$ :



## GRÁFICAS POLARES

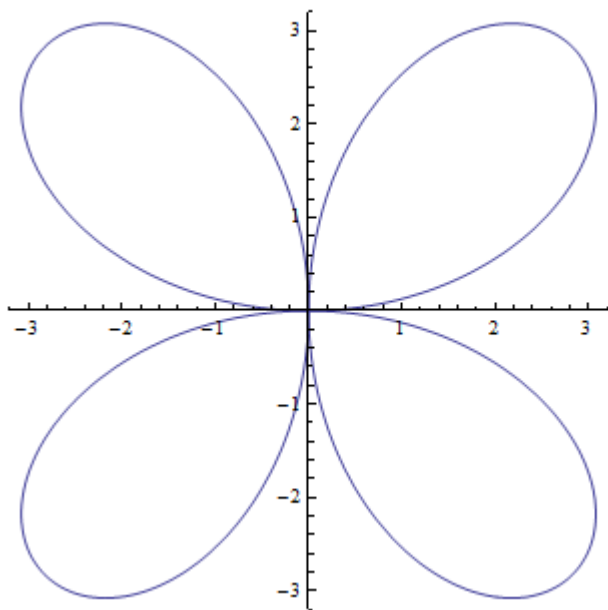
### Trébol de tres hojas

```
PolarPlot[Sin[3 t], {t, 0, Pi}, ImageSize -> 300]
```



## Trébol de cuatro hojas

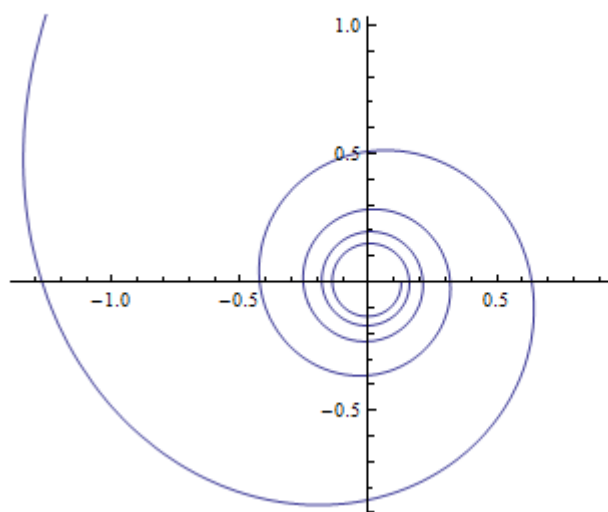
```
PolarPlot[4 Sin[2 t], {t, 0, 2 Pi}, ImageSize -> 300]
```





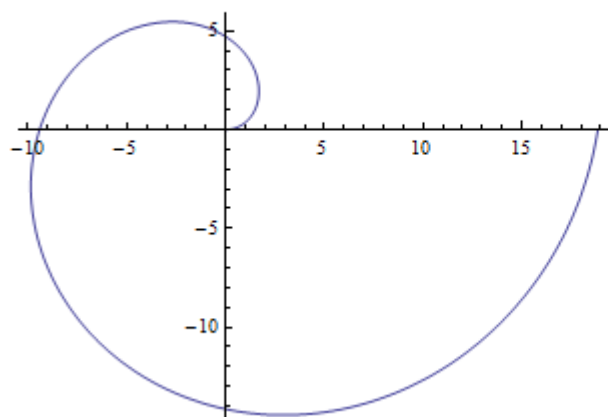
## Espiral

```
PolarPlot[4/t, {t, 0, 10 Pi}, ImageSize -> 300]
```



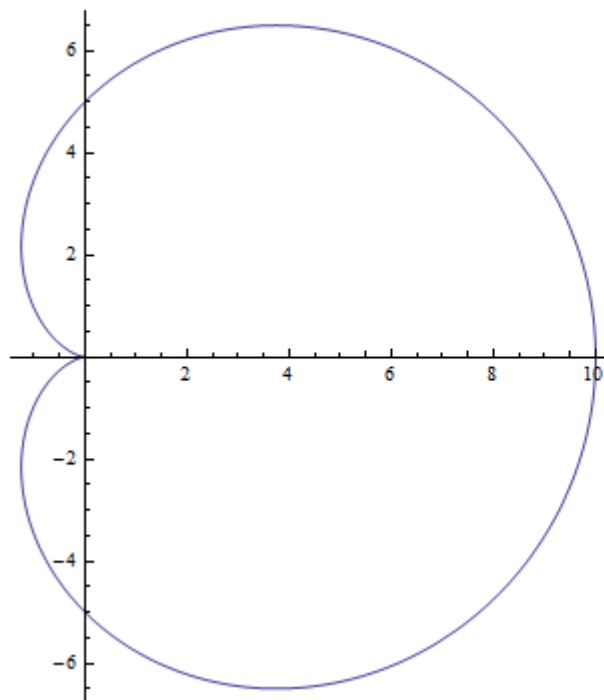
## Espiral de Arquímedes

```
PolarPlot[3 t, {t, 0, 2 Pi}, ImageSize -> 300]
```



## Cardioid

```
PolarPlot[5 (1 + Cos[t]), {t, 0, 2 Pi}, ImageSize -> 300]
```

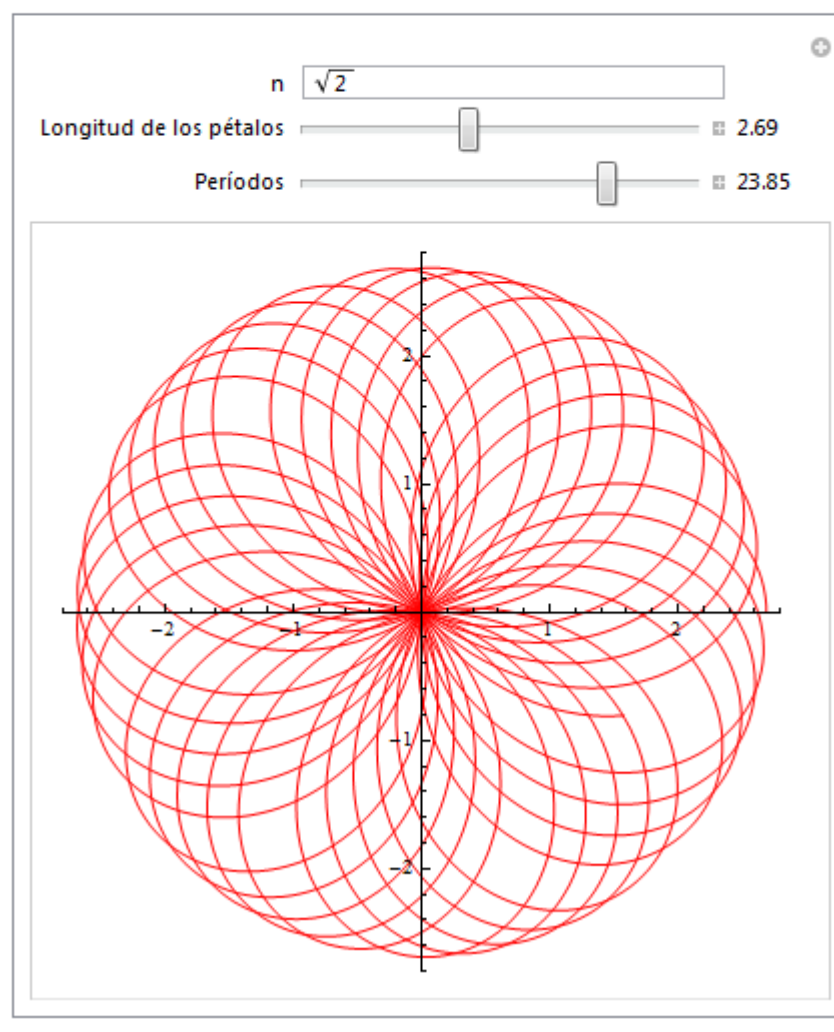


## Rosa de n pétalos

```

Manipulate[
  PolarPlot[a Cos[n  $\theta$ ], { $\theta$ , 0, p  $\pi$ },
    PlotStyle -> Directive[Red]],
  {n, Sqrt[2], 15, ControlType -> InputField},
  {{a, 3, "Longitud de los pétalos"}, 1, 5,
    Appearance -> "Labeled"},
  {{p, 2, "Períodos"}, 1, 30, Appearance -> "Labeled"}
]

```



## OBJETOS GRÁFICOS

### Puntos y líneas

#### Ejemplo

Dibujar un paralelogramo dadas sus cuatro vértices.

Definir cuatro puntos:

```
P1 = {0, 0};  
P2 = {0, 3};  
P3 = {5, 3};  
P4 = {5, 0};
```

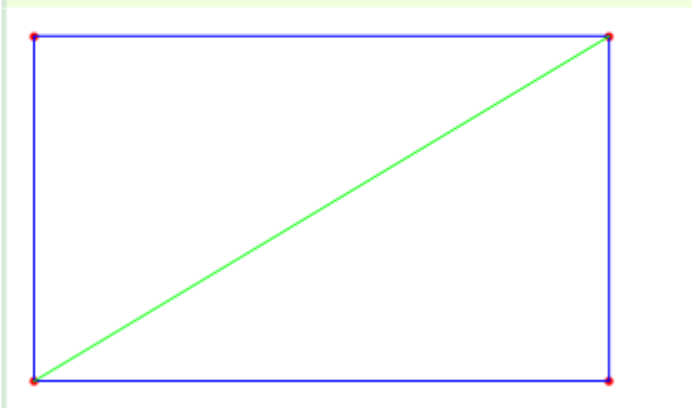
Dibujar los puntos y las líneas:

```
Graphics[  
  {Red, PointSize[Medium],  
   Point[{P1, P2, P3, P4}],  
   Blue,  
   Line[{P1, P2, P3, P4, P1}]},  
  ImageSize -> 300]
```



Dibujar la diagonal del paralelogramo:

```
Graphics[
  {Red, PointSize[Medium],
   Point[{P1, P2, P3, P4}],
   Blue,
   Line[{P1, P2, P3, P4, P1}],
   Green,
   Line[{P1, P3}]},
 ImageSize -> 300]
```



## Círculos, arcos y sectores de círculo

Círculo:

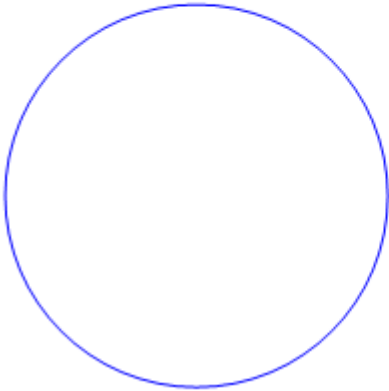
$Circle[\{x, y\}, radio]$

Arco:

$Circle[\{x, y\}, radio, \{\theta_1, \theta_2\}]$

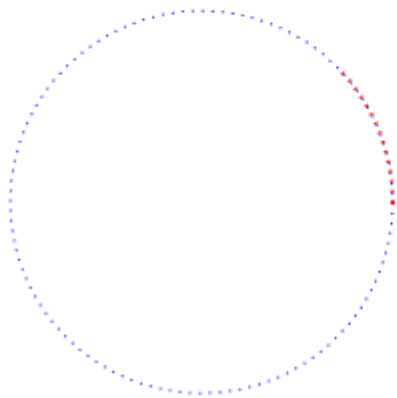
### Ejemplo

```
Graphics[{  
  Blue,  
  Circle[{0, 0}, 1],  
  ImageSize -> 200]
```



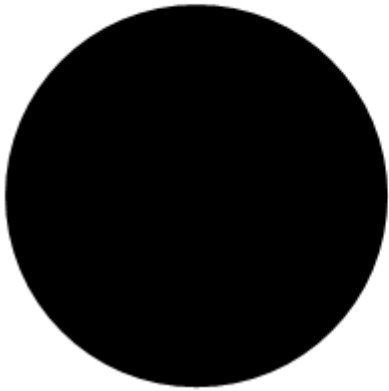
## Ejemplo

```
Graphics[
  {Blue, Dotted,
   Circle[{0, 0}, 1],
   Red, Thick,
   Circle[{0, 0}, 1, {0, Pi/4}]
  },
  ImageSize -> 200]
```



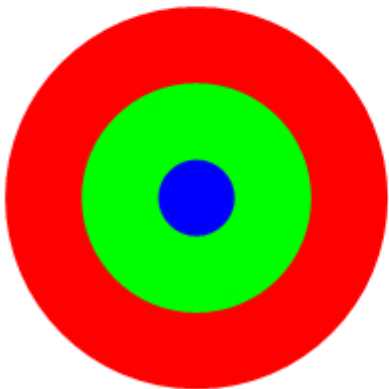
### Ejemplo

```
Graphics[  
  Disk[{0, 0}, 1],  
  ImageSize -> 200]
```



### Ejemplo

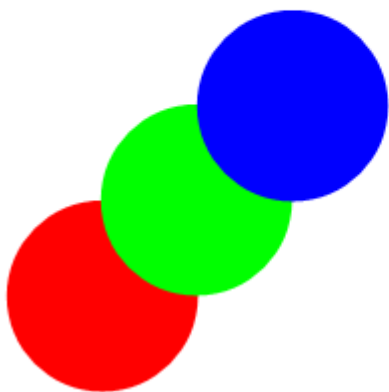
```
Graphics[  
  {Red, Disk[{0, 0}, 5],  
  Green, Disk[{0, 0}, 3],  
  Blue, Disk[{0, 0}, 1]},  
  ImageSize -> 200]
```





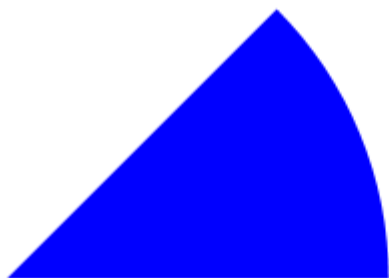
### Ejemplo

```
Graphics[  
  {Red, Disk[{0, 0}, 1],  
   Green, Disk[{1, 1}, 1],  
   Blue, Disk[{2, 2}, 1]},  
  ImageSize -> 200]
```



### Ejemplo

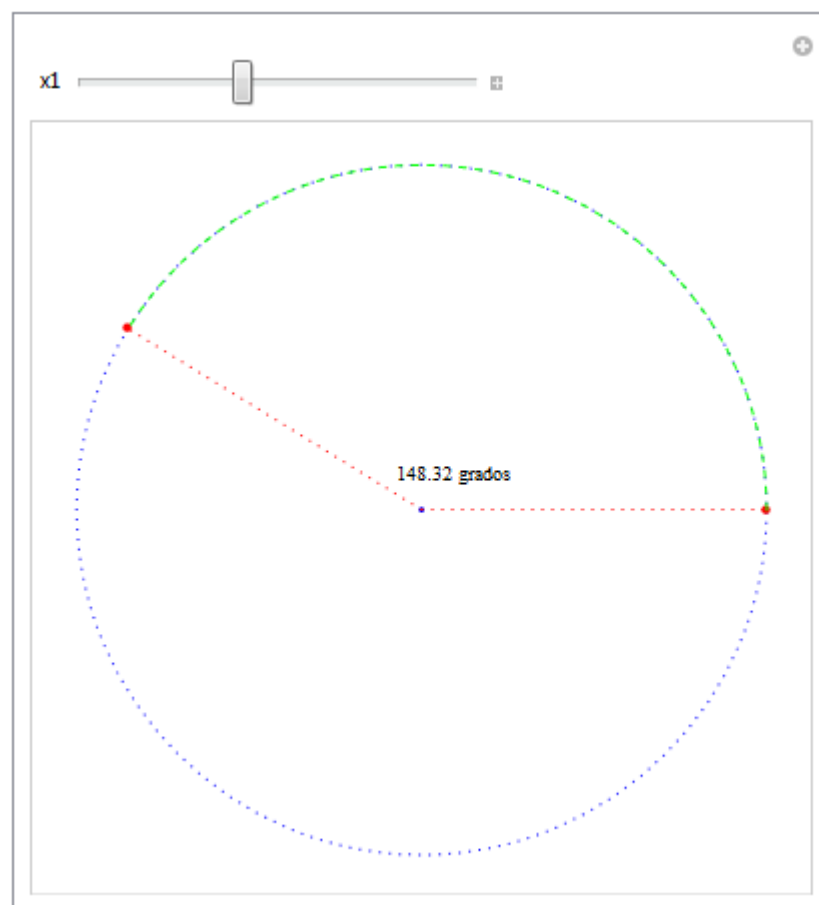
```
Graphics[{Blue, Disk[{0, 0}, 1, {0, Pi/4}]},  
  ImageSize -> 200]
```



## Ejemplo

Relación entre grados y radianes

```
Manipulate[
Module[{p1},
  p1 = {Cos[x1], Sin[x1]};
  Graphics[{Blue, Dotted, Circle[{0, 0}, 1],
    Point[{0, 0}],
    Red, PointSize[Medium], Point[{1, 0}],
    Line[{{0, 0}, {1, 0}}],
    Line[{{0, 0}, p1}],
    Red, PointSize[Medium], Point[p1],
    Green, Dashed, Circle[{0, 0}, 1, {0, x1}],
    Black, Text[ToString[x1 180 / Pi] <> " grados", {0.1, 0.1}]}]
, {x1, 0, 2 Pi}]
```

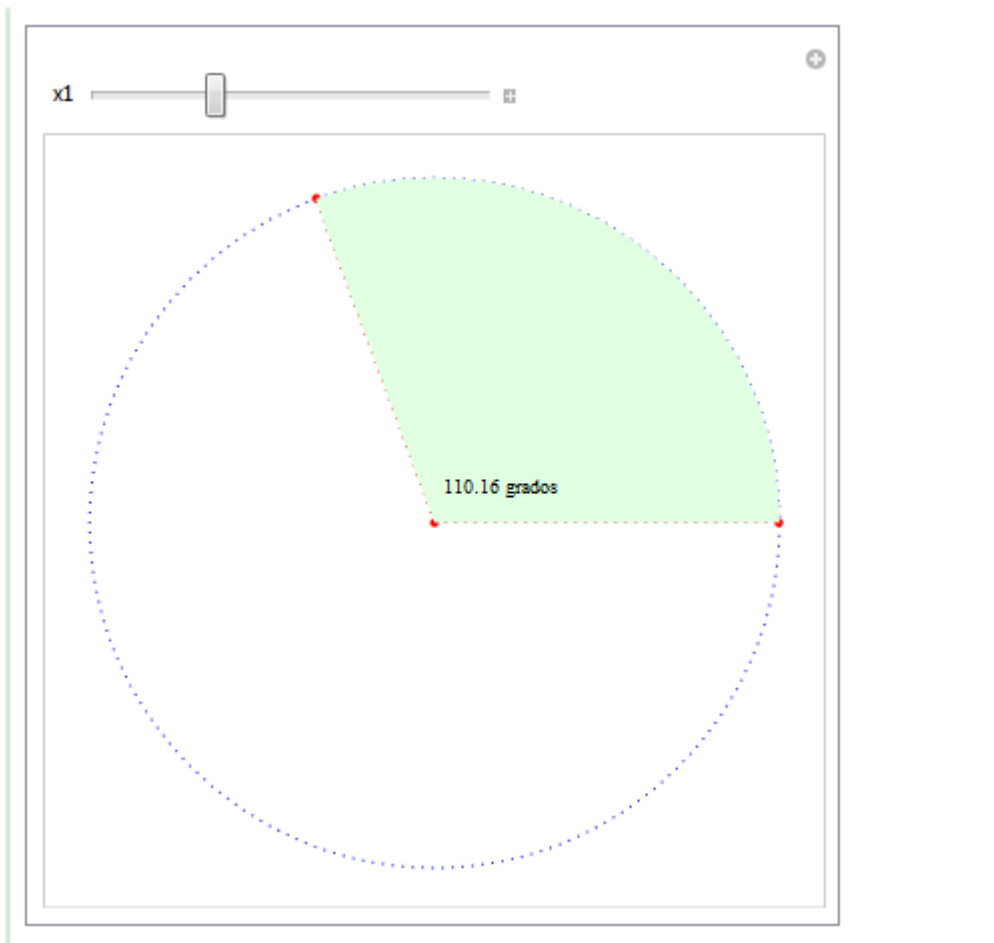


## Ejemplo

Sector de un disco:

$$\text{Disk}[\{x, y\}, \text{radio}, \{\theta_1, \theta_2\}]$$

```
Manipulate[
Module[{p1},
  p1 = {Cos[x1], Sin[x1]};
  Graphics[{{Blue, Dotted, Circle[{0, 0}, 1],
    Red, PointSize[Medium],
    Point[{0, 0}],
    Point[{1, 0}],
    Point[p1],
    Line[{{0, 0}, {1, 0}]},
    Line[{{0, 0}, p1]},
    LightGreen, Disk[{0, 0}, 1, {0, x1}],
    Black, Text[ToString[x1 180 / Pi] <> " grados", {0.2, 0.1}]}]
],
{x1, 0, 2 Pi}]
```



## Rotación de objetos gráficos

### Ejemplo

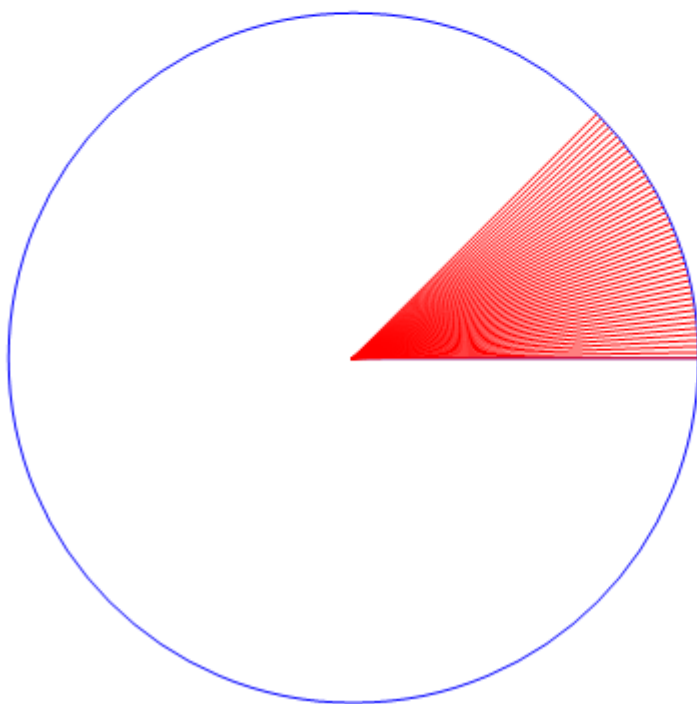
Rotate[objeto, ángulo, {x, y}]

Objeto: objeto de rotación

Ángulo: ángulo a ser rotado en radianes

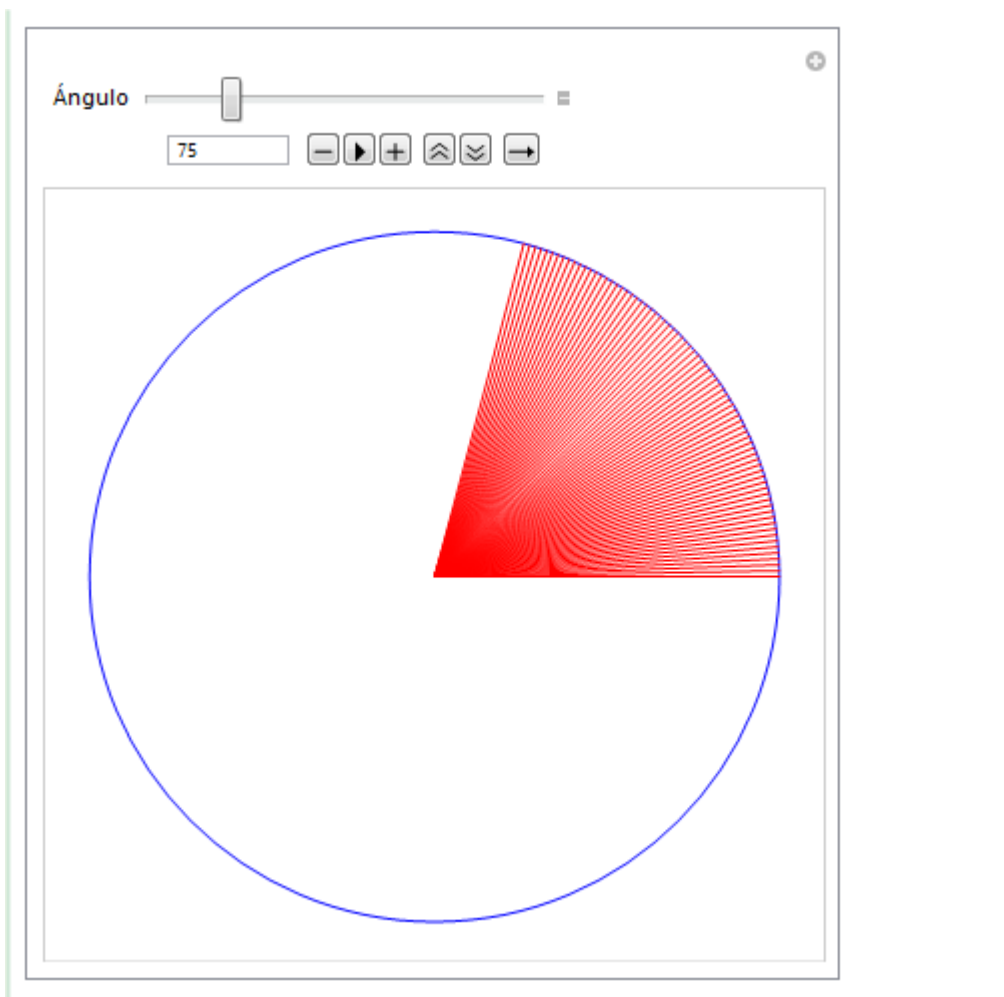
{x, y}: punto de rotación

```
Graphics[
{
  Blue,
  Circle[{0, 0}, 1],
  Line[{{0, 0}, {1, 0}}],
  Red,
  Table[
    Rotate[Line[{{0, 0}, {1, 0}}], i Degree, {0, 0}],
    {i, 0, 45}
  ]
}
]
```



Uso de la función `manipulate` para controlar la rotación de la línea:

```
Manipulate[
Graphics[
{
Blue,
Circle[{0, 0}, 1],
Line[{{0, 0}, {1, 0}}],
Red,
Table[
Rotate[Line[{{0, 0}, {1, 0}}], i Degree, {0, 0}],
{i, 0, angulo}
}
],
{{angulo, 45, "Ángulo"}, 1, 360, 1}]
```



## Ejemplo

Tiro parabólico

Supongamos que la ecuación  $-x^2 + x = -12$  representa el comportamiento seguido por una pelota de beisbol después de que un jugador le ha pegado con cierta velocidad y con cierto ángulo. Calcular los siguientes datos:

- El alcance horizontal de la pelota y
- la altura máxima alcanzada.

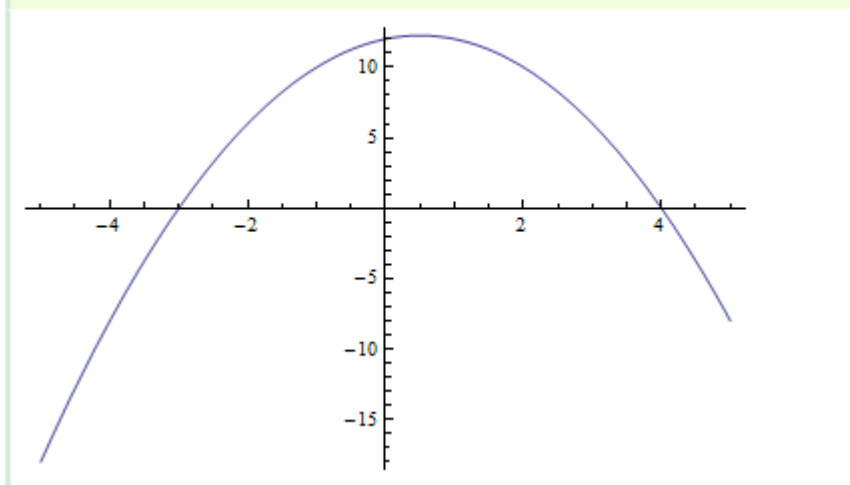
## Solución

Definir la función del tiro parabólico:

```
f[x_] := -x^2 + x + 12
```

Graficar la función original:

```
Plot[f[x], {x, -5, 5}]
```





Calcular el alcance horizontal.

Encontrar las intersecciones con el eje-X:

```
soluciones = Solve[f[x] == 0, x]
{{x → -3}, {x → 4}}
```

Crear los puntos de intersección con el eje-X:

```
a = x /. soluciones[[1]];
b = x /. soluciones[[2]];
puntoA = {a, 0};
puntoB = {b, 0};
```

Calcular la distancia entre los puntos de intersección:

```
distancia = EuclideanDistance[puntoA, puntoB]
7.
```

Calcular la altura alcanzada:

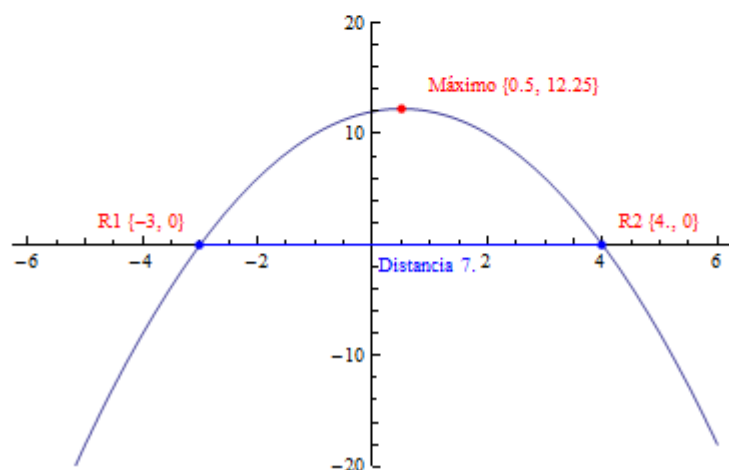
```
maximo = FindMaximum[f[x], {x, a, b}]
{12.25, {x → 0.5}}
```

Crear el punto de máxima altura:

```
x1 = x /. maximo[[2]];
y1 = maximo[[1]];
puntoMaximo = {x1, y1}
{0.5, 12.25}
```

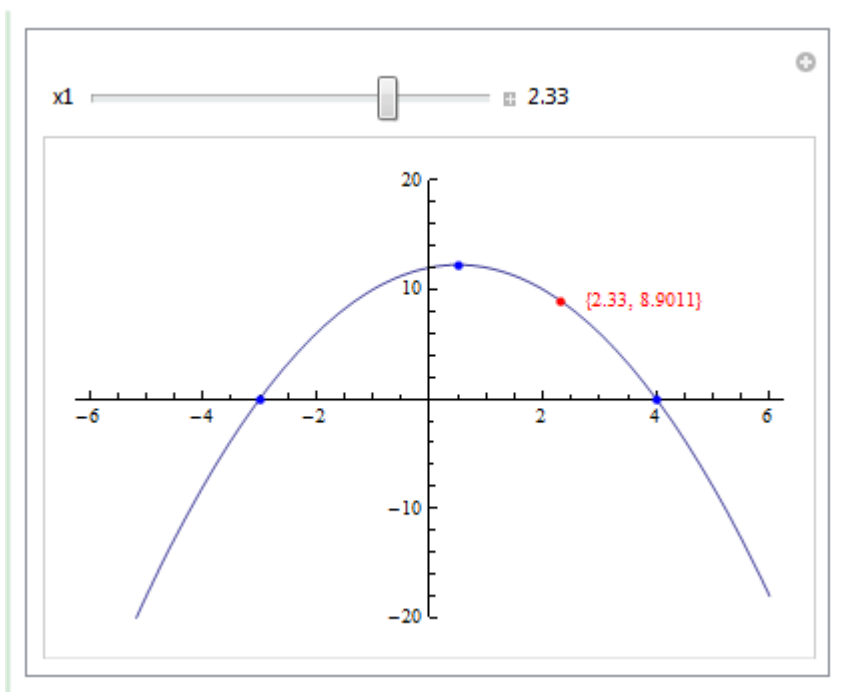
Graficar la función, los puntos de intersección en el eje-X, la altura máxima alcanzada y el segmento que define la distancia recorrida por la pelota:

```
Plot[f[x], {x, -6, 6}, PlotRange -> 20,
  Epilog -> {Blue, PointSize[Medium],
    Point[puntoA],
    Point[puntoB],
    Line[{puntoA, puntoB}],
    Text["Distancia " <> ToString[distancia], {1, -2}],
    Red,
    Point[puntoMaximo],
    Text["R1 " <> ToString[puntoA], {a - 1, 2}],
    Text["R2 " <> ToString[puntoB], {b + 1, 2}],
    Text["Máximo " <> ToString[puntoMaximo], {x1, y1} + 2]
  ]}]
```



Generalizar la solución

```
Manipulate[
Module[{fa, b, puntoA, PuntoB, maximo, puntoMaximo},
  f[x_] := -x^2 + x + 12;
  soluciones = NSolve[f[x] == 0, x];
  a = x /. soluciones[[1]];
  b = x /. soluciones[[2]];
  puntoA = {a, 0};
  puntoB = {b, 0};
  maximo = FindMaximum[f[x], {x, a, b}];
  puntoMaximo = {x /. maximo[[2]], maximo[[1]]};
  Plot[f[x], {x, -6, 6}, PlotRange -> 20,
  Epilog -> {Blue, PointSize[Medium],
    Point[puntoA],
    Point[puntoB],
    Point[puntoMaximo],
    Red,
    Point[{x1, f[x1]}],
    Text[{x1, f[x1]}, {x1 + 1.5, f[x1]}]
  }],
{x1, a, b, Appearance -> "Labeled"}]
```



## Ejemplo

Encontrar las raíces de la siguiente ecuación. Graficar la función y los puntos de intersección encontrados.

### Solución

Definir la función:

```
f = x^5 - 7 x^3 + 9 x;
```

Encontrar las raíces de la ecuación:

```
raices = NSolve[f == 0, x]
{{x -> -2.30278}, {x -> -1.30278},
 {x -> 0.}, {x -> 1.30278}, {x -> 2.30278}}
```

Aislar las soluciones encontradas:

```
raices = ReplaceAll[x, raices]
{-2.30278, -1.30278, 0., 1.30278, 2.30278}
```

Definir una función para calcular los puntos de intersección con el eje-X:

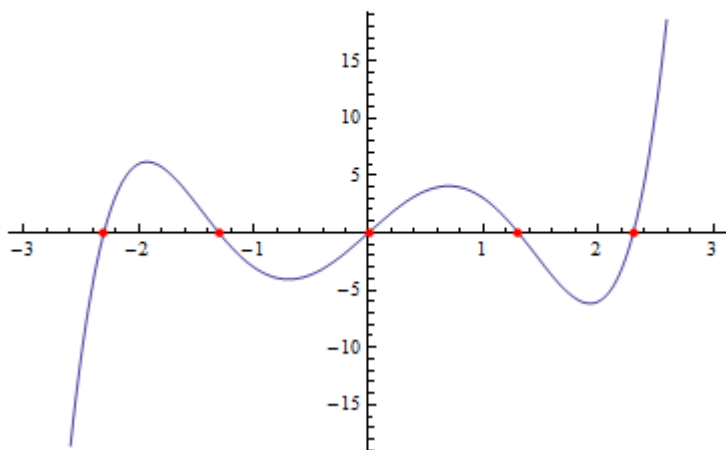
```
interseccion[a_] := {a, 0}
```

Calcular todos los puntos de intersección:

```
puntosInterseccion = Map[interseccion, raices]
{{-2.30278, 0}, {-1.30278, 0},
 {0., 0}, {1.30278, 0}, {2.30278, 0}}
```

Graficar la función y sus puntos de intersección con el eje-X:

```
Plot[f, {x, -3, 3},  
  Epilog -> {Red, PointSize[Medium],  
    Map[Point, puntosInterseccion]}]
```



## Ejemplo

Resolver el siguiente sistema de ecuaciones. Graficar la solución encontrada.

```
f = x^5 - 7 x^3 + 9 x;
g = 2 x;
```

## Solución

Solución numérica del par de ecuaciones:

```
soluciones = NSolve[f == g, x]
{{x → -2.40651}, {x → -1.09941},
 {x → 0.}, {x → 1.09941}, {x → 2.40651}}
```

Aislando los valores numéricos y almacenándolos en la misma variable:

```
soluciones = ReplaceAll[x, soluciones]
{-2.40651, -1.09941, 0., 1.09941, 2.40651}
```

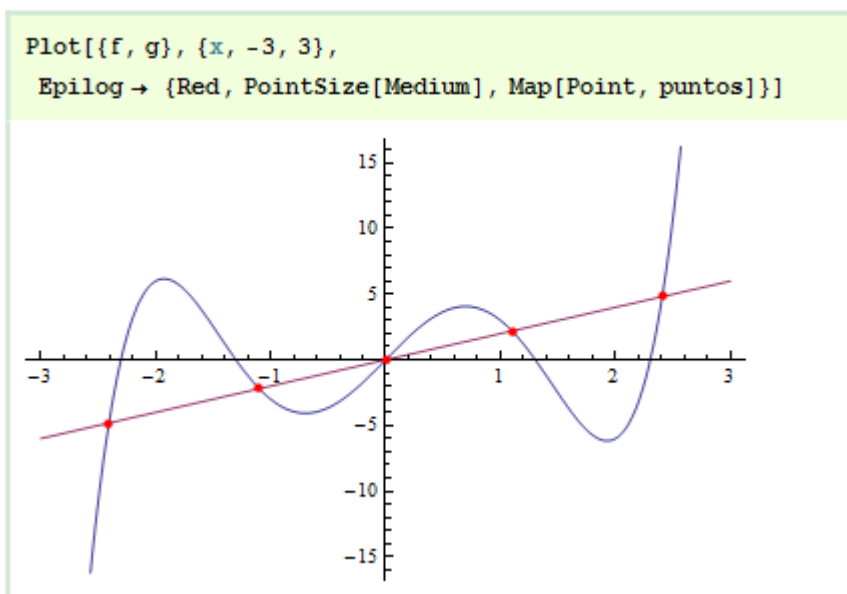
Definiendo una función para generar puntos para su graficación:

```
f3[a_] := {a, ReplaceAll[f, x → a]}
```

Generando la lista de puntos a graficar:

```
puntos = Map[f3, soluciones]
{{-2.40651, -4.81302}, {-1.09941, -2.19883},
 {0., 0.}, {1.09941, 2.19883}, {2.40651, 4.81302}}
```

Graficando las funciones y sus puntos de intersección:



## Ejemplo

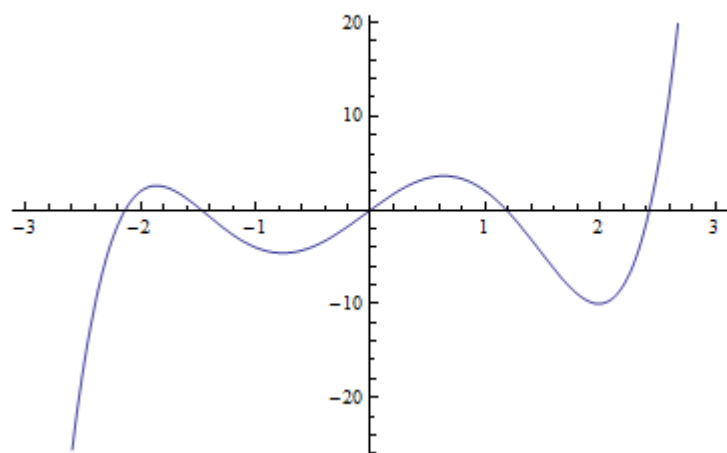
Mínimos de una función

Encontrar los puntos máximos de la siguiente función:

```
f1[x_] := x^5 - x^2 - 7 x^3 + 9 x;
```

Graficar la función:

```
Plot[f1[x], {x, -3, 3}]
```



Encontrar el primer máximo:

```
maximo1 = FindMaximum[f1[x], {x, -2, -1}]
```

```
{2.58244, {x → -1.86166}}
```

Formar el punto para su graficación:

```
x1 = ReplaceAll[x, maximo1[[2]]]
```

```
-1.86166
```

```
y1 = maximo1[[1]]
```

```
2.58244
```



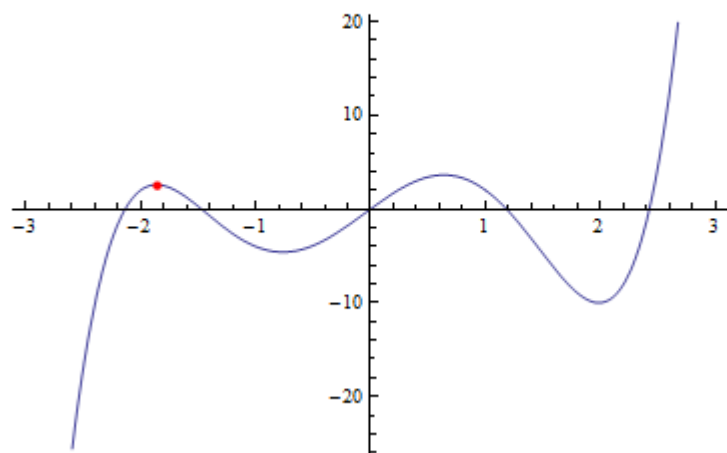
```
punto1 = {x1, y1}
```

```
{-1.86166, 2.58244}
```

Graficar la función y el primer máximo:

```
Plot[f1[x], {x, -3, 3},
```

```
Epilog -> {Red, PointSize[Medium], Point[punto1]}]
```



Encontrar el segundo máximo de la función:

```
maximo2 = FindMaximum[f1[x], {x, 0, 1}]
```

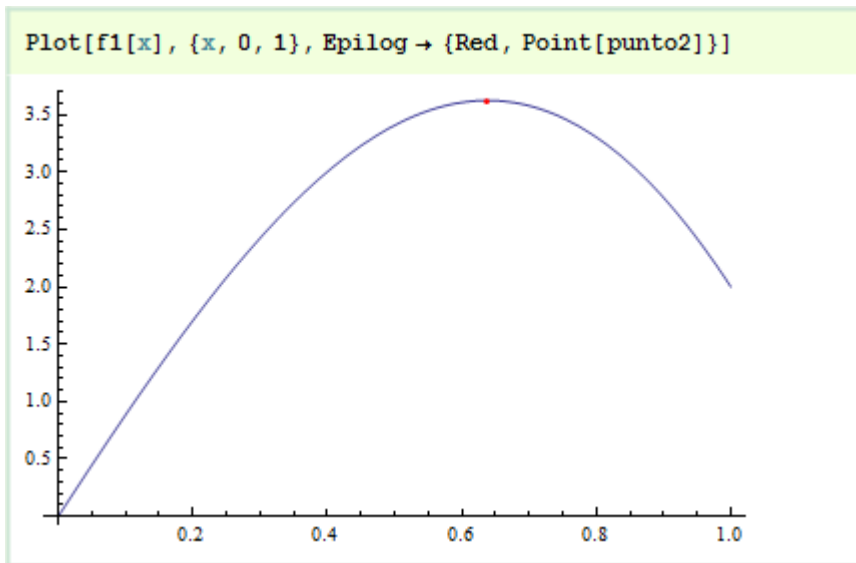
```
{3.6228, {x -> 0.638191}}
```

Formar el segundo punto:

```
punto2 = {ReplaceAll[x, maximo2[[2]]], maximo2[[1]]}
```

```
{0.638191, 3.6228}
```

Graficar la función y el segundo máximo:



Graficar la función y los dos puntos máximos:

