Taller 03: series de Taylor y polinomios de Lagrange

Nombre: David Alejandro Díaz Pineda

Grafique las curvas de las series de Taylor de varios órdenes para los siguientes casos:

In [1]:
```
import numpy as np
from numpy import cos, sin
import matplotlib.pyplot as plt
from math import factorial
```

1.

$$f(x) = ln(x)$$

$$x_0 = 1$$

$$orden = 5$$

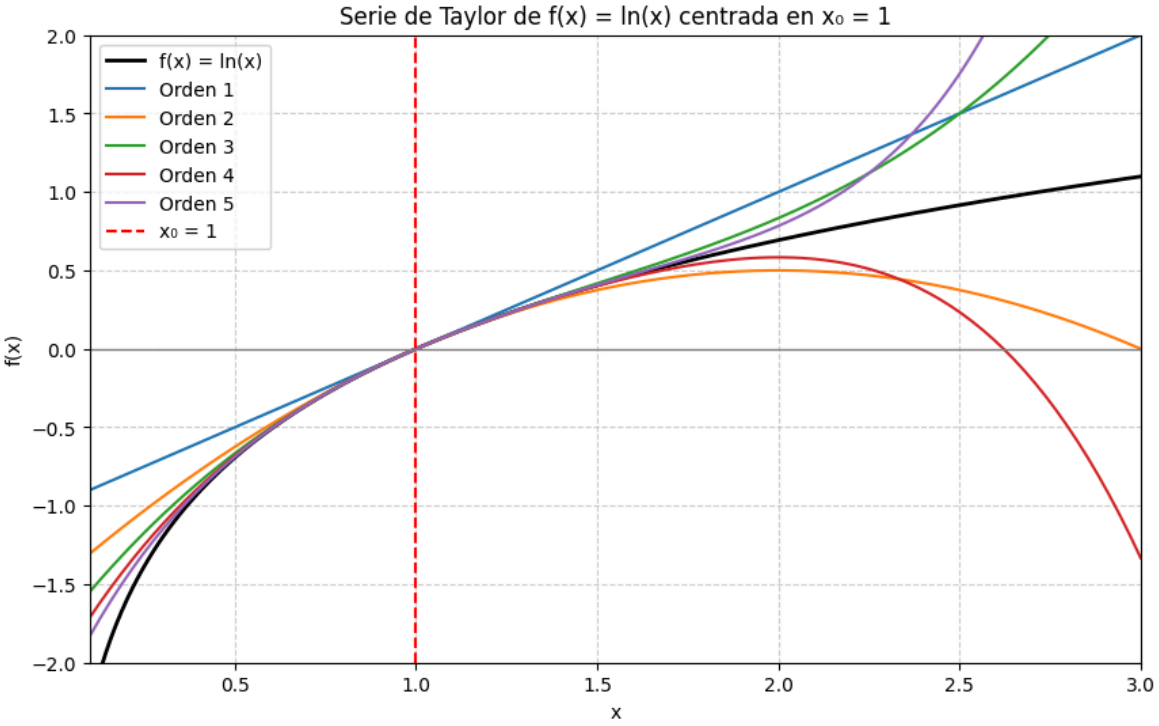
$f'(x) = x^{-1}$
$f''(x) = -x^{-2}$
$f^{(3)}(x) = 2x^{-3}$
$f^{(4)}(x) = -6x^{-4}$
$f^{(5)}(x) = 24x^{-5}$

Quedándonos

$$f(x) \approx ln(1) + (x-1) - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \frac{(x-1)^4}{4} + \frac{(x-1)^5}{5}$$

In [2]:
```
x1 = np.linspace(0.1, 3, 200)

y_ln = np.log(x1)

taylor_ln_1 = (x1 - 1)
taylor_ln_2 = taylor_ln_1 - (x1 - 1)**2 / 2
taylor_ln_3 = taylor_ln_2 + (x1 - 1)**3 / 3
taylor_ln_4 = taylor_ln_3 - (x1 - 1)**4 / 4
taylor_ln_5 = taylor_ln_4 + (x1 - 1)**5 / 5

plt.figure(figsize=(10, 6))
plt.plot(x1, y_ln, 'k-', linewidth=2, label='f(x) = ln(x)')
plt.plot(x1, taylor_ln_1, label='Orden 1')
plt.plot(x1, taylor_ln_2, label='Orden 2')
plt.plot(x1, taylor_ln_3, label='Orden 3')
plt.plot(x1, taylor_ln_4, label='Orden 4')
plt.plot(x1, taylor_ln_5, label='Orden 5')

plt.axhline(0, color='gray', linewidth=1)
```

```
plt.axvline(1, color='red', linestyle='--', linewidth=1.5, label='x₀ = 1')

plt.grid(True, linestyle='--', alpha=0.6)
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Serie de Taylor de f(x) = ln(x) centrada en x₀ = 1')
plt.legend()
plt.xlim(0.1, 3)
plt.ylim(-2, 2)
plt.show()
```



2.

$$f(x) = ln(x)$$

$$x_0 = 10$$

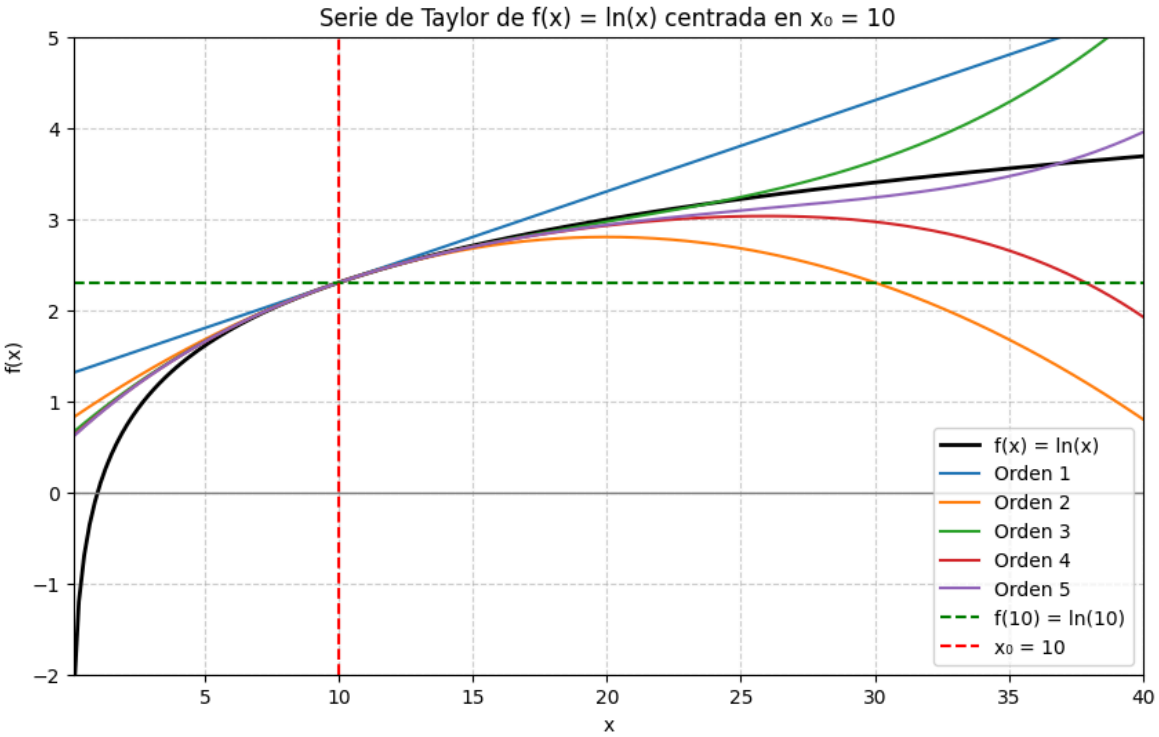
$$orden = 5$$

Quedándonos

$$f(x) \approx ln(10) + \frac{(x-10)}{10} - \frac{(x-10)^2}{10^2 * 2!} + \frac{2(x-10)^3}{10^3 * 3!} - \frac{6(x-10)^4}{10^4 * 4!} + \frac{24(x-10)^5}{10^5 * 5!}$$

In [3]:
```
x1 = np.linspace(0.1, 40, 200)

y_ln = np.log(x1)
f_x = np.log(10)
taylor_ln_1 = f_x + (x1 - 10)/10
taylor_ln_2 = taylor_ln_1 - (x1 - 10)**2 / (10**2 * factorial(2))
taylor_ln_3 = taylor_ln_2 + (x1 - 10)**3 / (10**3 * factorial(3))
taylor_ln_4 = taylor_ln_3 - (x1 - 10)**4 / (10**4 * factorial(4))
taylor_ln_5 = taylor_ln_4 + (x1 - 10)**5 / (10**5 * factorial(5))
```

```python
plt.figure(figsize=(10, 6))
plt.plot(x1, y_ln, 'k-', linewidth=2, label='f(x) = ln(x)')
plt.plot(x1, taylor_ln_1, label='Orden 1')
plt.plot(x1, taylor_ln_2, label='Orden 2')
plt.plot(x1, taylor_ln_3, label='Orden 3')
plt.plot(x1, taylor_ln_4, label='Orden 4')
plt.plot(x1, taylor_ln_5, label='Orden 5')
plt.plot(x1, np.full_like(x1, f_x), 'g--', label='f(10) = ln(10)')

plt.axhline(0, color='gray', linewidth=1)

plt.axvline(10, color='red', linestyle='--', linewidth=1.5, label='x₀ = 10')


plt.grid(True, linestyle='--', alpha=0.6)
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Serie de Taylor de f(x) = ln(x) centrada en x₀ = 10')
plt.legend()
plt.xlim(0.1, 40)
plt.ylim(-2, 5)
plt.show()
```



3.

$$f(x) = cos(x)$$

$$x_0 = 0$$

$$orden = 5$$

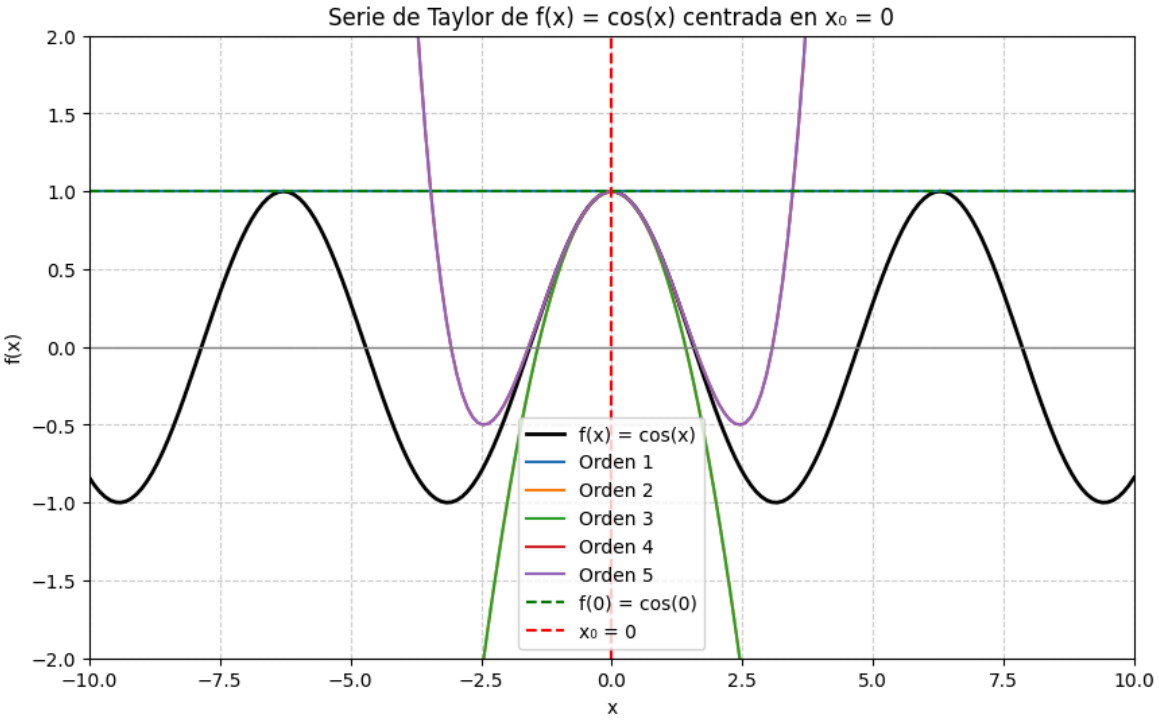
$f'(x) = -sen(x)$
$f''(x) = -cos(x)$
$f^{(3)}(x) = sen(x)$

$f^{(4)}(x) = cos(x)$
$f^{(5)}(x) = -sen(x)$

Quedándonos como expresión final

$$f(x) \approx cos(0) - \frac{cos(0)(x-0)^2}{2!} + \frac{cos(0)(x-0)^4}{4!}$$

In [4]:
```
x1 = np.linspace(-10, 10, 200)

y_cos = np.cos(x1)
f_x = cos(0)
taylor_cos_1 = f_x - sin(0)*(x1 - 0)
taylor_cos_2 = taylor_cos_1 - cos(0)*(x1 - 0)**2 / factorial(2)
taylor_cos_3 = taylor_cos_2 + sin(0)*(x1 - 0)**3 / factorial(3)
taylor_cos_4 = taylor_cos_3 + cos(0)*(x1 - 0)**4 / factorial(4)
taylor_cos_5 = taylor_cos_4 - sin(0)*(x1 - 0)**5 / factorial(5)

plt.figure(figsize=(10, 6))
plt.plot(x1, y_cos, 'k-', linewidth=2, label='f(x) = cos(x)')
plt.plot(x1, taylor_cos_1, label='Orden 1')
plt.plot(x1, taylor_cos_2, label='Orden 2')
plt.plot(x1, taylor_cos_3, label='Orden 3')
plt.plot(x1, taylor_cos_4, label='Orden 4')
plt.plot(x1, taylor_cos_5, label='Orden 5')
plt.plot(x1, np.full_like(x1, f_x), 'g--', label='f(0) = cos(0)')

plt.axhline(0, color='gray', linewidth=1)

plt.axvline(0, color='red', linestyle='--', linewidth=1.5, label='x₀ = 0')

plt.grid(True, linestyle='--', alpha=0.6)
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Serie de Taylor de f(x) = cos(x) centrada en x₀ = 0')
plt.legend()
plt.xlim(-10, 10)
plt.ylim(-2, 2)
plt.show()
```



4.

$$f(x) = \frac{1}{1-x}$$

$$x_0 = 0.5$$

$$orden = 5$$

$f'(x) = \frac{1}{(1-x)^2}$
$f''(x) = 2(1-x)^{-3}$
$f^{(3)}(x) = 6(1-x)^{-4}$
$f^{(4)}(x) = 24(1-x)^{-5}$
$f^{(5)}(x) = 120(1-x)^{-6}$

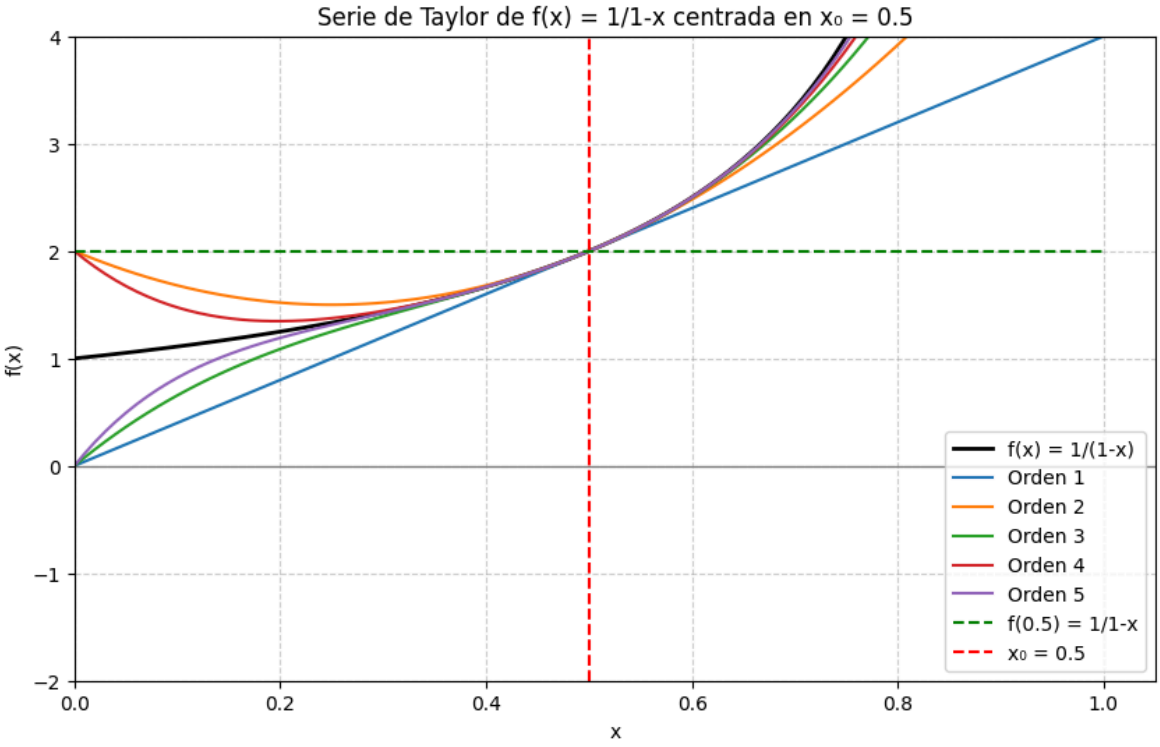
Quedándonos como expresión final

$$f(x) \approx 2 + 4(x-0.5) + 8(x-0.5)^2 + 16(x-0.5)^3 + 32(x-0.5)^4 + 64(x-0.5)^5$$

In [5]:
```
x1 = np.linspace(0, 1, 100)

y_f = 1/ (1-(x1))
f_x = 1/ (1-0.5)
taylor_f_1 = f_x + 4*(x1 - 0.5)
taylor_f_2 = taylor_f_1 + 2/(0.5**3 * factorial(2)) *(x1 - 0.5)**2
taylor_f_3 = taylor_f_2 + 6/(0.5**4 * factorial(3)) *(x1 - 0.5)**3
taylor_f_4 = taylor_f_3 + 24/(0.5**5 *  factorial(4)) *(x1 - 0.5)**4
taylor_f_5 = taylor_f_4 + 120/(0.5**6 *  factorial(5)) *(x1 - 0.5)**5

plt.figure(figsize=(10, 6))
plt.plot(x1, y_f, 'k-', linewidth=2, label='f(x) = 1/(1-x)')
plt.plot(x1, taylor_f_1, label='Orden 1')
```

```
plt.plot(x1, taylor_f_2, label='Orden 2')
plt.plot(x1, taylor_f_3, label='Orden 3')
plt.plot(x1, taylor_f_4, label='Orden 4')
plt.plot(x1, taylor_f_5, label='Orden 5')
plt.plot(x1, np.full_like(x1, f_x), 'g--', label='f(0.5) = 1/1-x')

plt.axhline(0, color='gray', linewidth=1)

plt.axvline(0.5, color='red', linestyle='--', linewidth=1.5, label='x₀ = 0.5')


plt.grid(True, linestyle='--', alpha=0.6)
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Serie de Taylor de f(x) = 1/1-x centrada en x₀ = 0.5')
plt.legend()
plt.xlim(0, )
plt.ylim(-2, 4)
plt.show()
```

```
C:\Users\dalz2\AppData\Local\Temp\ipykernel_22672\1009377270.py:3: RuntimeWarnin
g: divide by zero encountered in divide
  y_f = 1/ (1-(x1))
```



Encuentre el polinomio de Lagrange para los siguientes datos y grafique

1.

$$(0,0),(30,0.5),(60,\frac{\sqrt{3}}{2}),(90,1)$$

$$L_1(x) = \frac{(x)(x-60)(x-90)}{54000}$$

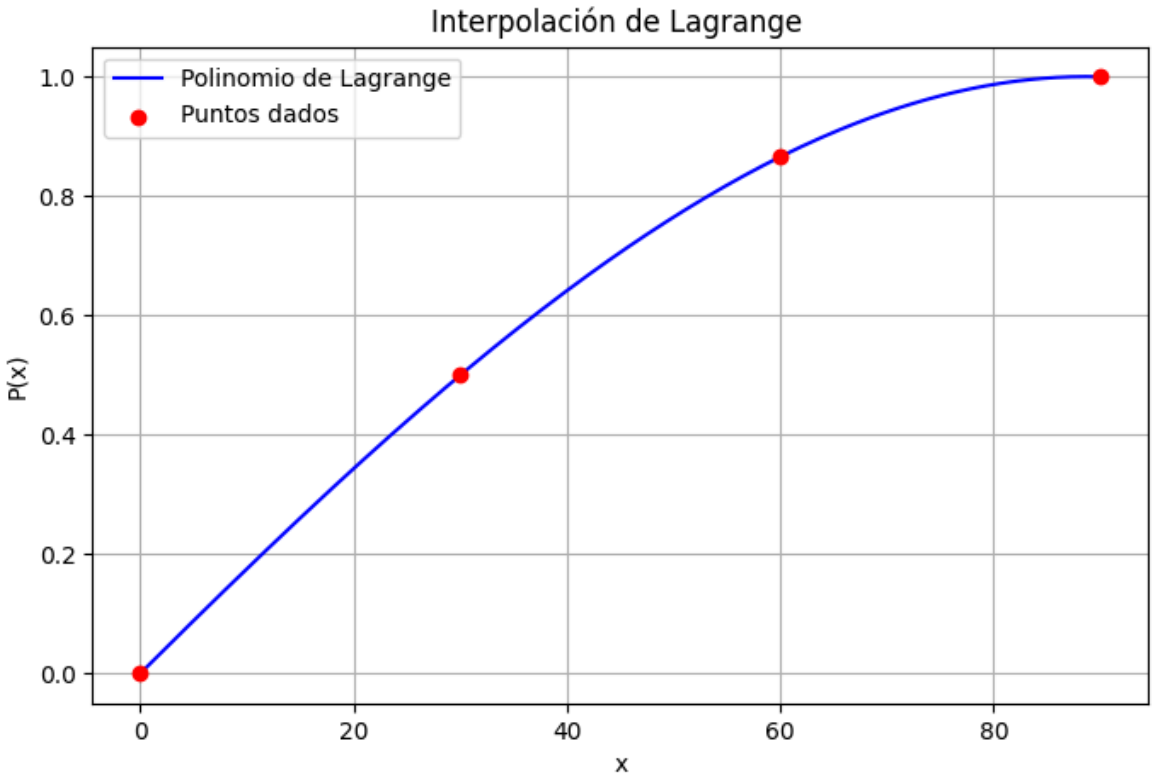
$$L_2(x) = -\frac{(x)(x-30)(x-90)}{54000}$$

$$L_3(x) = \frac{x(x-30)(x-60)}{162000}$$

$$P(x) = \frac{1}{2}\left(\frac{(x)(x-60)(x-90)}{54000}\right) - \frac{\sqrt{3}}{2}\left(\frac{(x)(x-30)(x-90)}{54000}\right) + \frac{x(x-30)(x-60)}{162000}$$

$$P(x) \approx (-6.05 * 10^{-7})x^3 - (1.9943 * 10^{-5})x^2 + 0.017x$$

In [6]:
```
def P(x):
    return ((1/64800 - np.sqrt(3)/108000)*x**3 +
            (-7/3600 + np.sqrt(3)/900)*x**2 +
            (11/180 - np.sqrt(3)/40)*x)

x_points = np.array([0, 30, 60, 90])
y_points = np.array([0, 0.5, np.sqrt(3)/2, 1])

x_vals = np.linspace(0, 90, 200)

plt.figure(figsize=(8,5))
plt.plot(x_vals, P(x_vals), label='Polinomio de Lagrange', color='blue')
plt.scatter(x_points, y_points, color='red', zorder=5, label='Puntos dados')
plt.title('Interpolación de Lagrange ')
plt.xlabel('x')
plt.ylabel('P(x)')
plt.legend()
plt.grid(True)
plt.show()
```



2.

$$(1,1), (2,2), (3,2)$$

$$L_0(x) = \frac{(x-2)(x-3)}{2}$$

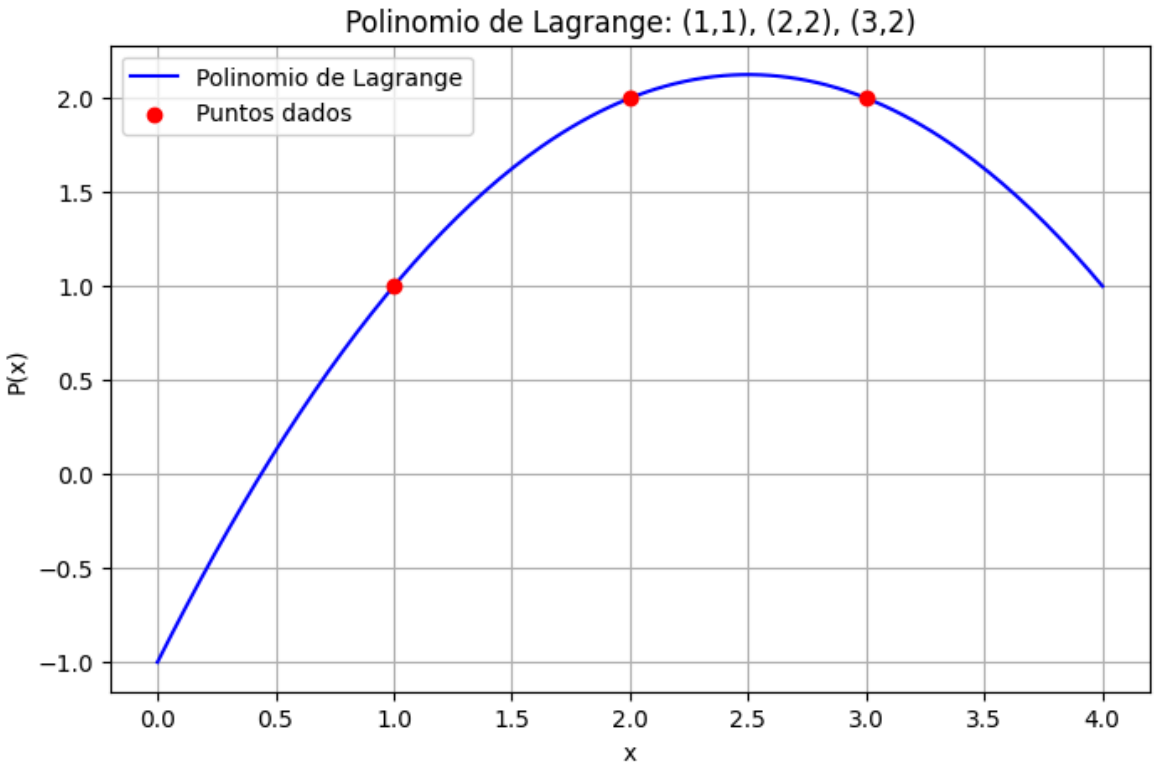
$$L_1(x) = (x-1)(x-3)$$

$$L_2(x) = \frac{(x-1)(x-2)}{2}$$

$$P(x) = 1\frac{(x-2)(x-3)}{2} - 2(x-1)(x-3) + 2\frac{(x-1)(x-2)}{2}$$

$$P(x) = -\frac{1}{2}x^2 + \frac{5}{2}x - 1$$

In [7]:
```
x_points = np.array([1, 2, 3])
y_points = np.array([1, 2, 2])

def p(x):
    return -0.5*x**2 + 2.5*x - 1
x_vals = np.linspace(0, 4, 200)

plt.figure(figsize=(8,5))
plt.plot(x_vals, p(x_vals), label='Polinomio de Lagrange', color='blue')
plt.scatter(x_points, y_points, color='red', zorder=5, label='Puntos dados')
plt.title('Polinomio de Lagrange: (1,1), (2,2), (3,2)')
plt.xlabel('x')
plt.ylabel('P(x)')
plt.legend()
plt.grid(True)
plt.show()
```



3.

$$(-2,5), (1,7), (3,11), (7,4)$$

$$L_0(x) = -\frac{(x-1(x-3)(x-7))}{135}$$

$$L_1(x) = \frac{(x+2)(x-3)(x-7)}{36}$$

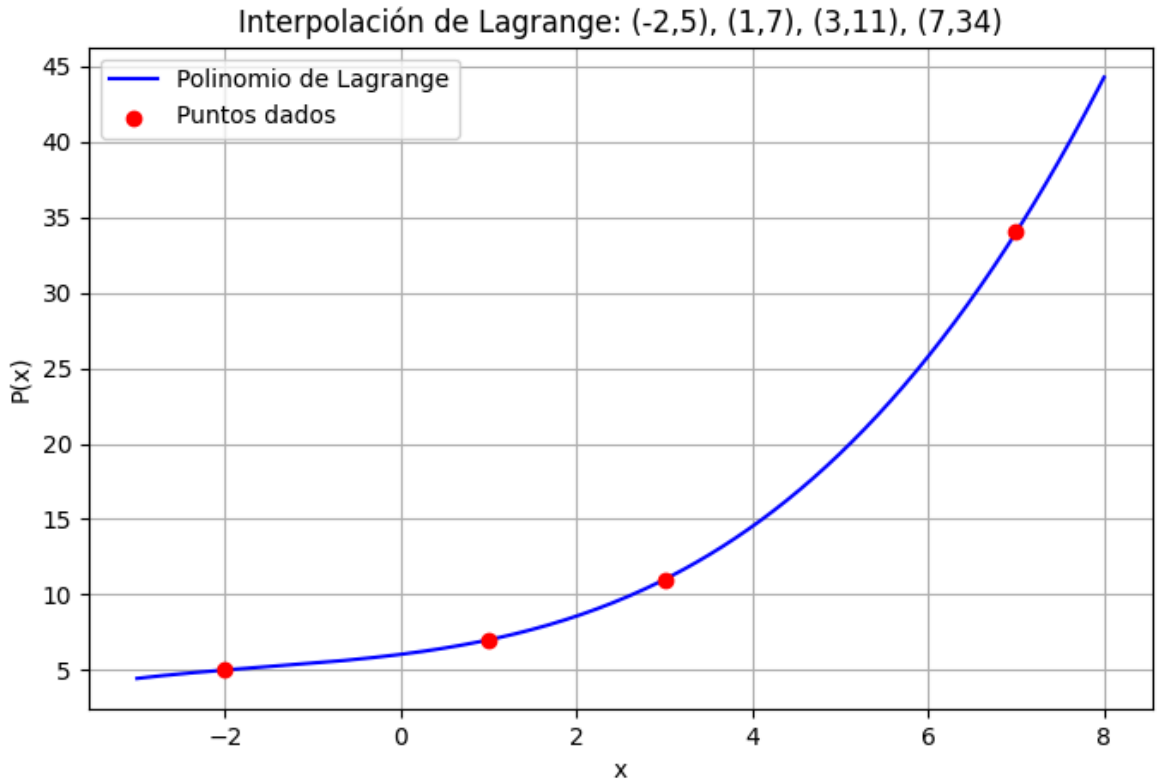
$$L_2(x) = -\frac{(x+2)(x-1(x-7))}{40}$$

$$L_3(x) = \frac{(x+2)(x-1)(x-3)}{216}$$

$$P(x) = 5(\frac{(x-1(x-3)(x-7))}{135}) + 7(\frac{(x+2)(x-3)(x-7)}{36}) - 11(\frac{(x+2)(x-1(x-7))}{40}) + 34(\frac{(x+2)(x-1)(x-3}{216}$$

$$P(x) = \frac{43x3+202x^2+793x+6522}{1087}$$

In [8]:
```
x_points = np.array([-2, 1, 3, 7])
y_points = np.array([5, 7, 11, 34])

def P(x):
    return (43*x**3 + 202*x**2 + 793*x + 6522) / 1080

x_vals = np.linspace(-3, 8, 300)
y_vals = P(x_vals)

plt.figure(figsize=(8,5))
plt.plot(x_vals, y_vals, color='blue', label='Polinomio de Lagrange')
plt.scatter(x_points, y_points, color='red', zorder=5, label='Puntos dados')
plt.title('Interpolación de Lagrange: (-2,5), (1,7), (3,11), (7,34)')
plt.xlabel('x')
plt.ylabel('P(x)')
plt.grid(True)
plt.legend()
plt.show()
```


Interpolación de Lagrange: (-2,5), (1,7), (3,11), (7,34)
Polinomio de Lagrange
Puntos dados
P(x)
x
45
40
35
30
25
20
15
10
5
−2
0
2
4
6
8