Special Functions (SKP)

In [1]:
```
import numpy as np
from numpy import *
import matplotlib.pyplot as plt
import sympy as smp
from sympy import *
import scipy as sp
from scipy import *
```

In `sympy`

[https://docs.sympy.org/latest/modules/functions/special.html (https://docs.sympy.org/latest/modules/functions/special.html)](https://docs.sympy.org/latest/modules/functions/special.html)

```
# in scipy
import scipy
scipy.special?
```

Gamma function

$$\Gamma(n) = \int_0^\infty t^{n-1} e^{-t}\, \mathrm{d}t$$

$\Gamma(n) = (n-1)!$ when n is an integer.

In [2]:
```
from sympy import gamma
n = S(1)/2  # input a value
gamman = smp.gamma(n)
display(gamman, gamman.evalf())
```

$\sqrt{\pi}$

1.77245385090552

`gamma` : Gamma function.

In [ ]:
```

```

Beta function

$$\mathrm{B}(m,n) \int_0^1 t^{m-1}(1-t)^{n-1}\mathrm{d}t$$

$$\mathrm{B}(m,n) = \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n)}$$

In [3]:
```python
from sympy import beta
m, n = 3,4  # input values
betamn = smp.beta(m,n)
display(betamn, betamn.evalf())
```

$\mathrm{B}(3, 4)$

0.0166666666666667

`beta` : Beta function.

In [ ]:
```python

```

Error function and Complementary Error function

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\,\mathrm{d}t$$

In [4]:
```python
from sympy import erf, erfc
x = 1.1  # input a value
display(erf(x), erf(x).evalf())
display(erfc(x), erfc(x).evalf())
```

0.880205069574082

0.880205069574082

0.119794930425918

0.119794930425918

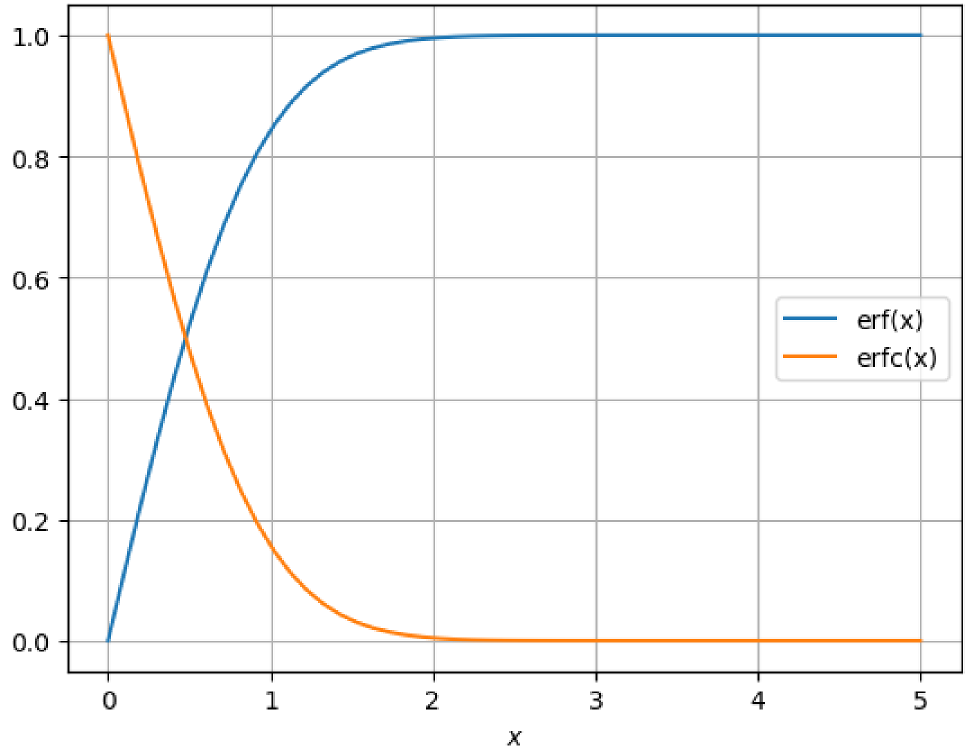
`scipy.special` functions

`erf` : Returns the error function of complex argument.

`erfc` : Complementary error function, `1 - erf(x)` .

`erf_zeros` : Compute nt complex zeros of error function erf(z).

In [5]:
```
from scipy.special import erf, erfc
x = np.linspace(0,5,50)
plt.plot(x, erf(x), label='erf(x)')
plt.plot(x, erfc(x), label='erfc(x)')
plt.xlabel('$x$')
plt.legend()
plt.grid()
plt.show()
```



Legendre Polynomials

$$(1-x^2)y'' - 2xy' + n(n+1)y = 0$$

$$P_n(x) = \sum_{k=0}^{m} (-1)^k \frac{(2n-2k)!}{2^n k!(n-k)!(n-2k)!} x^{n-2k}$$

Associated Legendre Polynomials:

$$P_v^m = (-1)^m (1-x^2)^{m/2} \frac{d^m}{dx^m} P_v(x)$$

where,

$$P_v = \sum_{k=0}^{\infty} \frac{(-v)_k (v+1)_k}{(k!)^2} \left(\frac{1-x}{2}\right)^k$$

In [6]:
```
from sympy import legendre, assoc_legendre
x = smp.symbols('x')
n, m = 3,2    # input the degree and order
print('degree, n =', n, '\t order, m =', m)
display('legendre polynomial', legendre(n,x))
display('associated legendre polynomial',assoc_legendre(n,m,x))
```

```
degree, n = 3    order, m = 2
'legendre polynomial'
```

$$\frac{5x^3}{2} - \frac{3x}{2}$$

```
'associated legendre polynomial'
```

$$15x\left(1 - x^2\right)$$

`scipy.special` functions

`legendre` : Legendre polynomial.

`lpn` : Legendre function of the first kind.

`lqn` : Legendre function of the second kind.

`lpmv` : Associated Legendre function of integer order and real degree.

`clpmn` : Associated Legendre function of the first kind for complex arguments.

`lpn` : Legendre function of the first kind.

`lqn` : Legendre function of the second kind.

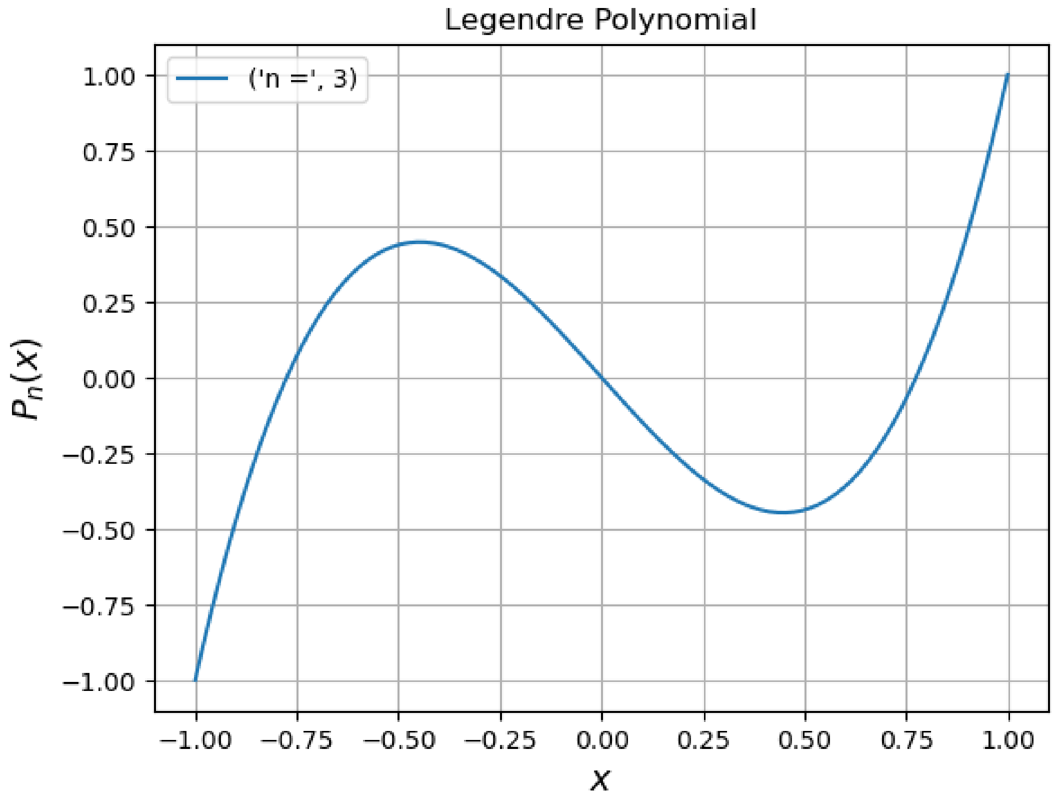
`lpmn` : Sequence of associated Legendre functions of the first kind.

`lqmn` : Sequence of associated Legendre functions of the second kind.

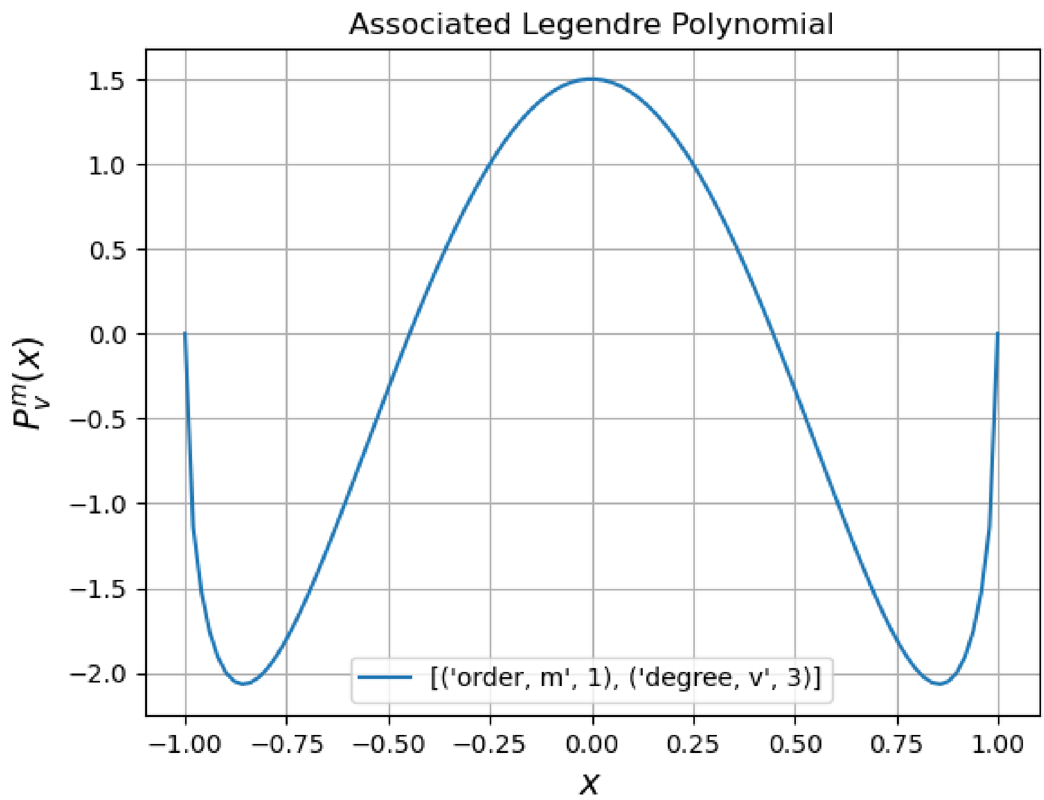
In [7]:
```
from scipy.special import legendre

n = 3      # value of degree
x = np.linspace(-1,1,100)
Pn = legendre(n)(x)

plt.plot(x,Pn, label=('n =',n))
plt.xlabel('$x$', fontsize=14)
plt.ylabel('$P_n(x)$', fontsize=14)
plt.title('Legendre Polynomial')
plt.legend()
plt.grid()
plt.show()
```



In [8]:
```
from scipy.special import lpmv
m, v = 1, 3   # order and degree
x = np.linspace(-1,1,100)
plt.plot(x,lpmv(m,v,x), label=[('order, m',m),('degree, v',v)])
plt.xlabel('$x$', fontsize=14)
plt.ylabel('$P^m_v(x)$', fontsize=14)
plt.title('Associated Legendre Polynomial')
plt.legend()
plt.grid()
plt.show()
```



Bessel Functions

In [9]:
```
from sympy import besselj
x = smp.symbols('x')
besselj(2,x)  # order
```

Out[9]: $J_2(x)$

$$x^2y'' + xy' + (x^2 - n^2)y = 0$$

$$J_n(x) = \sum_{m=0}^{\infty} \frac{(-1)^m}{m!\Gamma(m+n+1)} (\frac{x}{2})^{(2m+n)}$$

$$J_{-n}(x) = (-1)^n J_n(x)$$

`scipy.special` functions

`jv` : Bessel function of the first kind of real order and complex argument.

`yn` : Bessel function of the second kind of integer order and real argument.

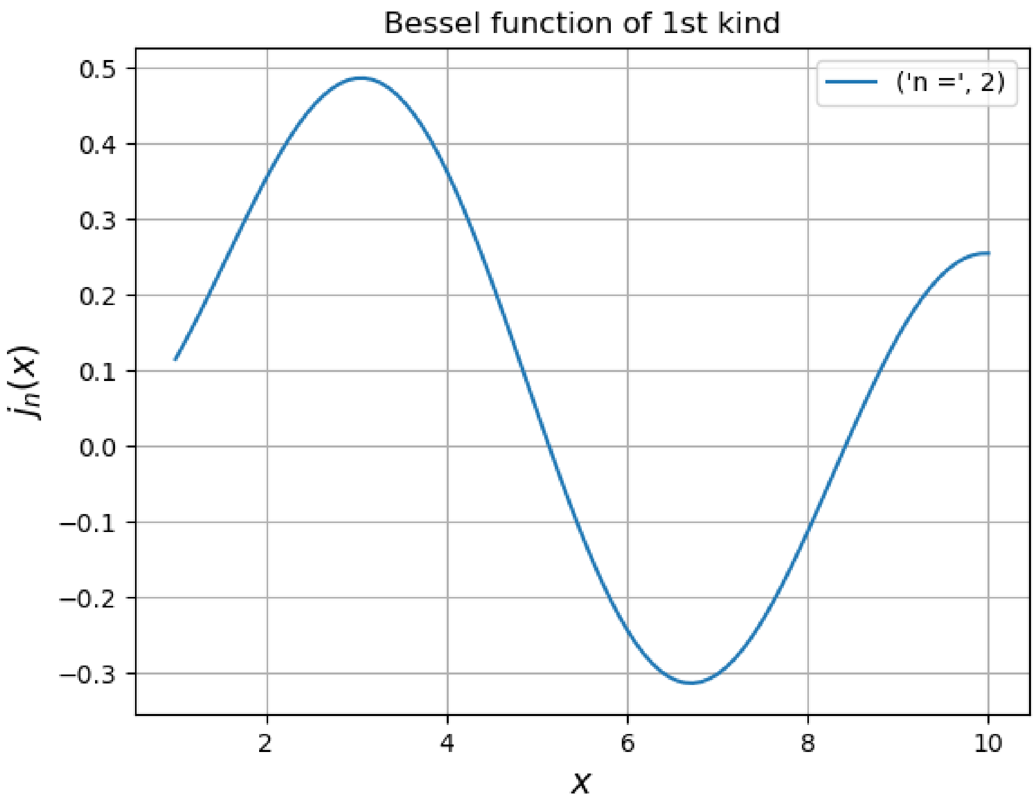
`yv` : Bessel function of the second kind of real order and complex argument.

jn_zeros : Compute zeros of integer-order Bessel function Jn(x).

In [10]:
```
from scipy.special import *
import matplotlib.pyplot as plt

n = 2   # value of n
x = np.linspace(1,10,100)
jn = jv(n,x)

plt.plot(x,jn, label=('n =',n))
plt.xlabel('$x$', fontsize=14)
plt.ylabel('$j_n(x)$', fontsize=14)
plt.title('Bessel function of 1st kind')
plt.legend()
plt.grid()
plt.show()
zeros = jn_zeros(n,10) # set the numbers of zeros required
print('zeros of the Bessel function are,', zeros)
```




```
zeros of the Bessel function are, [ 5.1356223   8.41724414 11.61984117 14.79595178
17.95981949 21.11699705
 24.27011231 27.42057355 30.5692045  33.71651951]
```

In [ ]:

Hermite Polynomials

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2}$$

In [11]:
```
from sympy import hermite
x = smp.symbols('x')
n = 2  # input value n
print('n =', n)
hermite(n,x)
```

```
n = 2
```

Out[11]: $4x^2 - 2$

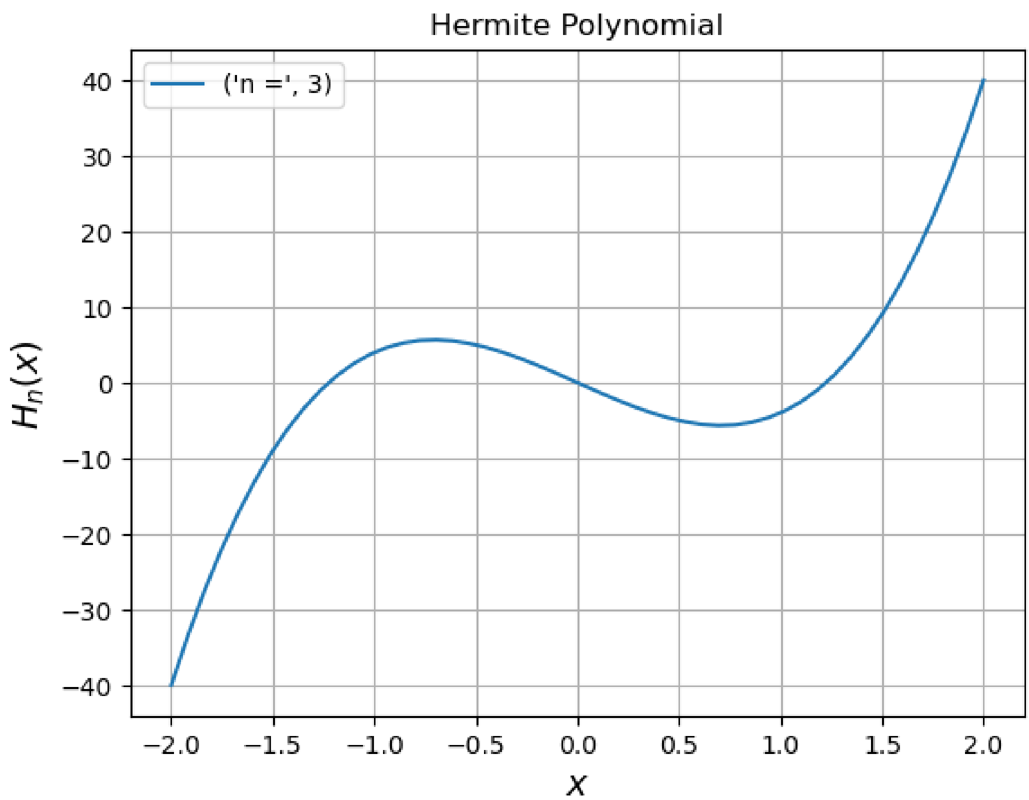
`scipy.special` functions

`hermite` : Physicist's Hermite polynomial.

In [12]:
```
from scipy.special import hermite

n = 3  # order of the polynomial
x = np.linspace(-2,2,50)
Hn = hermite(n)(x)

plt.plot(x,Hn, label=('n =',n))
plt.xlabel('$x$', fontsize=14)
plt.ylabel('$H_n(x)$', fontsize=14)
plt.title('Hermite Polynomial')
plt.legend()
plt.grid()
plt.show()
```



In [ ]:

Laguerre Polynomials

$$x\frac{d^2}{dx^2}L_n + (1-x)\frac{d}{dx}L_n + nL_n = 0$$

Solution: Laguerre polynomial of degree n in x, $L_n(x)$.

Associated Laguerre Polynomial:

$$x\frac{d^2}{dx^2}L_n^{(\alpha)} + (\alpha + 1 - x)\frac{d}{dx}L_n^{(\alpha)} + nL_n^{(\alpha)} = 0$$

Where, $\alpha > -1$; $L_n^{(\alpha)}$ is a polynomial of degree n.

In [13]:
```
from sympy import laguerre, assoc_laguerre
x, a = smp.symbols('x a')
n, a = 2, a  # input degree and a
print('n =', n, '\t a =', a)
display('laguerre polynomial', laguerre(n,x))
display('associated laguerre polynomial',assoc_laguerre(n,a,x))
```

```
n = 2    a = a
```

```
'laguerre polynomial'
```

$$\frac{x^2}{2} - 2x + 1$$

```
'associated laguerre polynomial'
```

$$\frac{a^2}{2} + \frac{3a}{2} + \frac{x^2}{2} + x(-a - 2) + 1$$

`scipy.special` functions

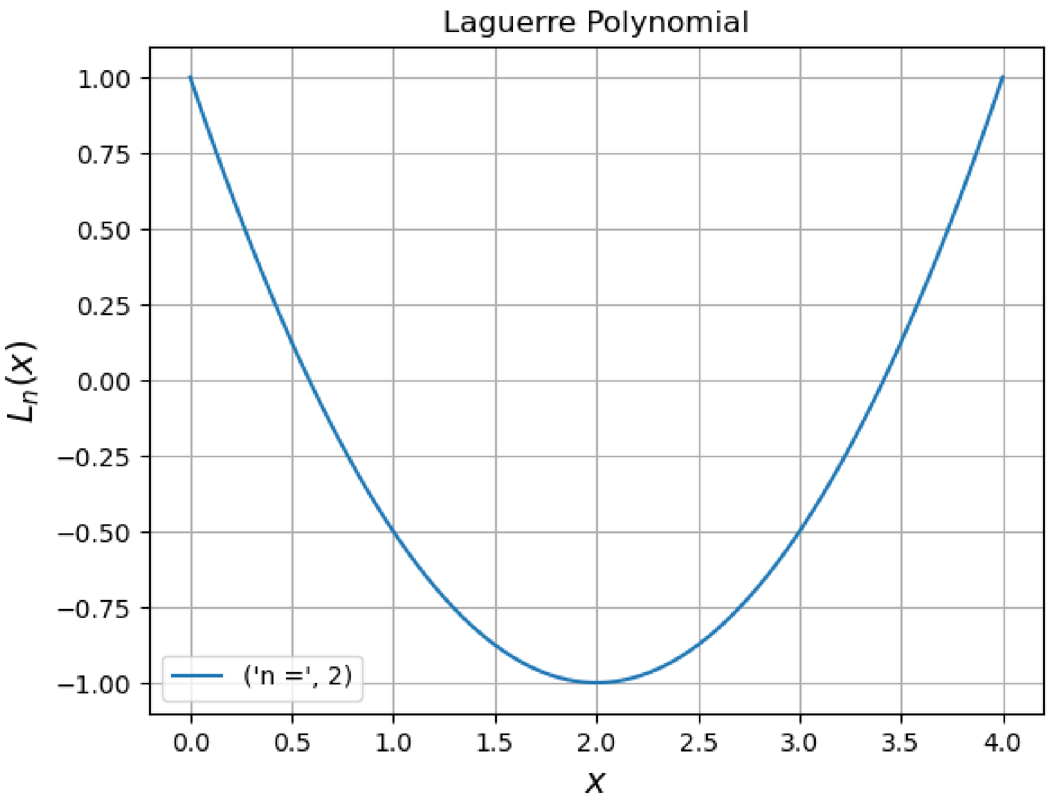
`laguerre` : Laguerre polynomial.

`genlaguerre` : Generalized (associated) Laguerre polynomial.

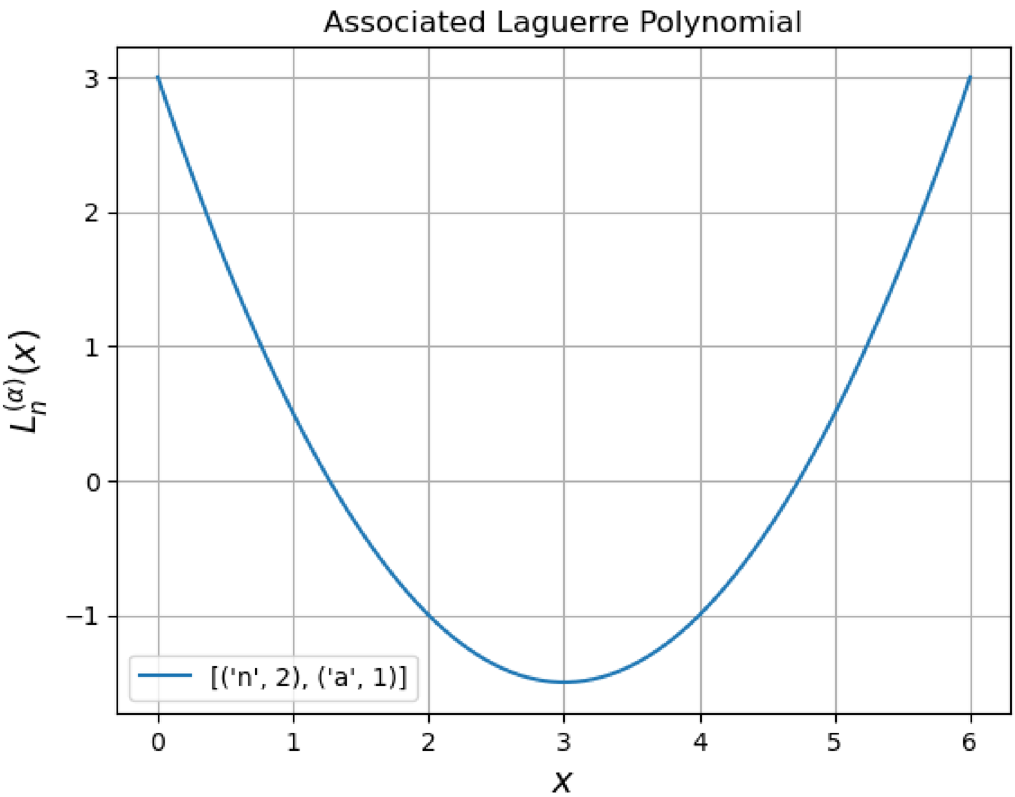
`assoc_laguerre` : Compute the generalized (associated) Laguerre polynomial of degree n and order k.

In [14]:

```
from scipy.special import laguerre
n = 2
x = np.linspace(0,4,100)
Lnx = laguerre(n)(x)
plt.plot(x,Lnx, label=('n =', n))
plt.xlabel('$x$', fontsize=14)
plt.ylabel('$L_n(x)$', fontsize=14)
plt.title('Laguerre Polynomial')
plt.legend()
plt.grid()
plt.show()
```



In [15]:
```
from scipy.special import genlaguerre
n, a = 2, 1
x = np.linspace(0,6,100)
Lnax = genlaguerre(n,a)(x)
plt.plot(x,Lnax, label=[('n', n),('a', a)])
plt.xlabel('$x$', fontsize=14)
plt.ylabel(r'$L_n^{(\alpha)}(x)$', fontsize=14)
plt.title('Associated Laguerre Polynomial')
plt.legend()
plt.grid()
plt.show()
```



Permutation and Combinations

`scipy.special` functions

`comb` : The number of combinations of N things taken k at a time.

`perm` : Permutations of N things taken k at a time, i.e., k-permutations of N.

In [ ]:
```
```

In [ ]:
```
```

Riemann zeta function and Riemann zeta function minus 1

`scipy.special` functions

`zeta` : Riemann zeta function.

`zetac` : Riemann zeta function minus 1.

```
In [ ]:
```

```
In [ ]:
```