In [13]:

```
%autosave 0
```

```
Autosave disabled
```

Chapter 27: Power Series Solutions of Linear Differential Equations w Variable Coefficients.

Finding and expressing differential equation solutions in terms of power series.

Note: When transcribing text for entry into Wolfram Alpha the "fenceposts" !| ... |! are excluded from the text.

27.2 Find a recurrence formula for the power series solution around $x = 0$ for

$$y'' - xy' + 2y = 0$$

In [6]:

```
import sympy as sp
x = sp.Symbol('x')
y = sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp= sp.Derivative(y, x, x)
equation = sp.Eq(3*x + yp + 2*y, 0)
display(equation)

```

$$3x + 2y(x) + \frac{d}{dx}y(x) = 0$$

In [7]:

```
solution = sp.dsolve(equation, hint='1st_power_series')
display(solution)

```

$$y(x) = \frac{x^2 \cdot (4C_1 - 3)}{2} - \frac{x^3 \cdot (4C_1 - 3)}{3} + \frac{x^4 \cdot (4C_1 - 3)}{6} - \frac{x^5 \cdot (4C_1 - 3)}{15} + C_1 - 2C_1 x + O\left(x^6\right)$$

For a restricted set of equations, (above), sympy has an ability to go straight to a series solution.

Possibly it subverts the purpose of the problem set, but it might be possible to take a found solution and turn that into something in series form, all in Wolfram Alpha.

For the present example, the W|A solution is much uglier than the sympy version, so the sympy version will be used.

In [8]:

```
equation2 = sp.Eq(ydp -x*yp + 2*y, 0)
solution = sp.dsolve(equation2)
display(solution)

```

$$y(x) = C_2 \cdot \left(1 - x^2\right) + C_1 x \left(1 - \frac{x^2}{6}\right) + O\left(x^6\right)$$

(Above): Surprise! For this particular equation, sympy prefers to display it in series form, anticipating the intent of the problem. However, sympy does not give enough information to guess the recurrence relation. A new kid on the block, Symbolab, chips in with a more detailed solution:

$$y = c_1\left(1 - x^2\right) + c_2\left(x - \frac{x^3}{6} - \frac{x^5}{120} + \ldots + \frac{x^{2n+1}(-1)\,(1)\,(2n-3)}{(2)(3)(4)(5)\ldots 2n(2n+1)} + \ldots\right)$$

27.3 Find a recurrence formula for the power series solution around $x = 0$ for

$$y'' - xy' + 2y = 0$$

In [4]:
```
from sympy import *

x = symbols('x')
n = Symbol('n', positive=True, integer=True)
r = limit((-x**(2*n-1))/factorial(2*n - 1), x, oo)
if (r==0):
    print('nope')
else:
    print('converges to  ' + str(r))
print('limit of the given function  ' + str(r))

```

```
converges to  -oo
limit of the given function  -oo
```

In [5]:
```
from sympy import *

x = symbols('x')
n = Symbol('n', positive=True, integer=True)
r = limit((-x**(2*n+1)*(2*n - 3))/factorial(2*n + 1), x, oo)
if (r==0):
    print('nope')
else:
    print('converges to  ' + str(r))
print('limit of the given function  ' + str(r))

```

```
converges to  -oo*sign(2*n - 3)
limit of the given function  -oo*sign(2*n - 3)
```

The above two cells show that neither rendition of the Symbolab solution converges.

27.5 Find a recurrence formula for the power series $y'' + y' = 0$

If this differential equation is entered in Symbolab, the solution which is returned is:

$$y = c_1 \cos(t) + c_2 \sin(t)$$

The Maclaurin series for these two trig functions are equal to:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!}$$

$$\sin(x) = 1 - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)!}$$

No method for derivation of the recurrence formula based on the above information can be discerned. The derivation process, as developed by the text, is fairly involved.

27.9 Find a recurrence formula for the differential equation $y'' + (t-1)y' + (2t-3)y = 0$

Symbolab will not solve the problem equation. Maxima will not solve it either. Wolfram Alpha will solve it in a long, complicated solution containing the dreaded erfi function. The much more reasonable-looking solution of sympy can be seen below.

In [108]:
```
import sympy as sp

t = sp.Symbol('t')
y = sp.Function('y')(t)
yp = sp.Derivative(y, t)
ydp= sp.Derivative(y, t, t)
equation = sp.Eq(ydp + (t - 1)*yp + (2*t - 3)*y, 0)
display(equation)

```

$$(t-1)\,\frac{d}{dt}y(t)+(2t-3)\,y(t)+\frac{d^2}{dt^2}y(t)=0$$

In [109]:
```
solution = sp.dsolve(equation)
display(solution)

```

$$y(t)=C_2\left(\frac{t^4}{6}+\frac{t^3}{6}+\frac{3t^2}{2}+1\right)+C_1 t\left(\frac{t^2}{2}+\frac{t}{2}+1\right)+O\left(t^6\right)$$

Above: sympy chooses to couch its solution in terms of a series. Since that is the subject of this chapter, the choice is congenial. The answer does not comprise a recurrence formula, .

27.12 Find a recurrence formula for the differential equation $(1-x^2)\,y''-2xy'+n(n+1)\,y=0$

Considering the order and general structure of the problem equation, it seems unlikely that any except sympy will solve it.

In [117]:
```
import sympy as sp

n = symbols('n')
x = sp.Symbol('x')
y = sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp= sp.Derivative(y, x, x)
equation = sp.Eq((1 - x**2)*ydp - 2*x*yp + n*(n + 1)*y, 0)
display(equation)

```

$$n\,(n+1)\,y(x)-2x\frac{d}{dx}y(x)+\left(1-x^2\right)\frac{d^2}{dx^2}y(x)=0$$

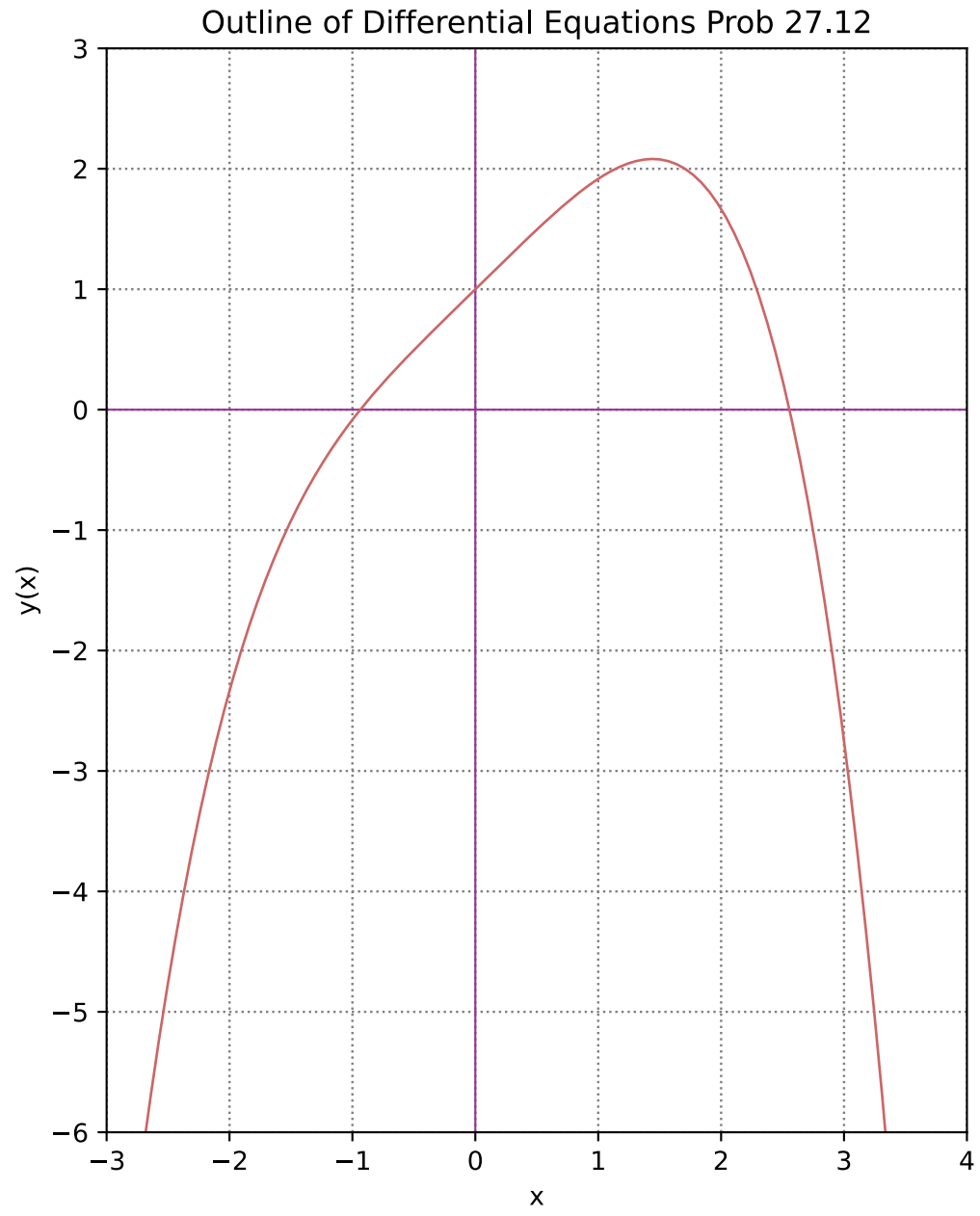
In [118]:
```
solution = sp.dsolve(equation)
```

$$y(x)=C_2\left(\frac{n^4x^4}{24}+\frac{n^3x^4}{12}-\frac{5n^2x^4}{24}-\frac{n^2x^2}{2}-\frac{nx^4}{4}-\frac{nx^2}{2}+1\right)+C_1 x\left(-\frac{n^2x^2}{6}-\frac{nx^2}{6}+\frac{x^2}{3}+1\right)+O\left(x^6\right)$$

A solution in series form, with 11 terms. It might be interesting to see what a plot would look like (with constants and n equal to 1).

In [135]:
```
import numpy as np
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']


x = np.linspace(-10,10,300)
y1 = (x**4/24) + (x**4/12) + (x**4/24) + (x**2/2) - (x**4/4) - \
    (x**2/2) + 1 + x*(-(x**2/6) - (x**2/6) + (x**2/3) + 1)


plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.12")
plt.rcParams['figure.figsize'] = [6, 9]


ax = plt.gca()
ax.axhline(y=0, color='#993399', linewidth=0.8)
ax.axvline(x=0, color='#993399', linewidth=0.8)
```

```
ratio = 0.98
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)


#plt.text(965, -2.5e43, "-np.log(abs((np.cos(x/2))**2 - (np.sin(x/2))**2)))
#plt.text(965, 2.25e43, "SOLN: y = 3*np.exp(x**2) + 1/2", size=10,\
        # bbox=dict(boxstyle="square", ec=('#8C564B'),fc=(1., 1., 1),))
plt.ylim(-6,3)
plt.xlim(-3, 4)
plt.plot(x, y1, linewidth = 1, color = '#CC6666')
plt.show()

```



27.14 Find a recurrence formula for the differential equation $(x^2 + 4)\, y'' + xy = x + 2$

A rather difficult problem, apparently. Symbolab will not solve it, sympy will not solve it, and Maxima will not solve it. Only Wolfram Alpha will solve this one, and it is a long solution. However, it does not contain erfi or any exotic elements like imaginaries.

The entry line for W|A is:

!| (x^2 + 4) y'' + x y = x + 2 |!

and the result is:

$$y(x) = c_1 + c_2\, x + \frac{x^2}{4} + \left(\frac{1}{24} - \frac{c_1}{24}\right) x^3 + \left(-\frac{c_2}{48} - \frac{1}{96}\right) x^4 + \left(\frac{c_1}{320} - \frac{1}{160}\right) x^5 +$$

$$\left(\frac{c_1}{2880} + \frac{c_2}{480} + \frac{1}{1440}\right) x^6 + \left(-\frac{c_1}{2688} + \frac{c_2}{8064} + \frac{13}{16128}\right) x7 +$$

$$\left(-\frac{13\, c_1}{215040} - \frac{c_2}{3584} - \frac{1}{15360}\right) x^8 + \left(\frac{11 c_1}{207360} - \frac{7\, c_2}{276480} - \frac{199}{1658880}\right) x^9 +$$

$$\left(\frac{101\, c_1}{9676800} + \frac{25\, c_2}{580608} + \frac{229}{29030400}\right) x^{10} + \left(-\frac{97\, c_1}{11354112} + \frac{113\, c_2}{23654400} + \frac{401}{20275200}\right) x^{11} +$$

$$\left(-\frac{2881\, c_1}{1532805120} - \frac{7451\, c_2}{1021870080} - \frac{6851}{6131220480}\right) x^{12} + O\left(x^{13}\right)$$

Of course this is only the solution. The recurrence formula would possibly be even more complex to calculate. In keeping with the chapter theme, W|A delivers its solution in series form.

27.16 Find the recurrence formula for the power series solution around $t = 0$ for the nonhomogeneous differential equation $(d^2y/dt^2) + ty = e^{t+1}$.

The sympy soln below contains Airy function terms. Though the name contains the letter 'i', this is not to be confused with an instance of the imaginary 'i'. This is something different. Symbolab will not solve it, Maxima will not solve it, and the Wolfram Alpha solution is not appetizing. Reliance falls on sympy.

In [8]:
```
import sympy as sp

n = sp.Symbol('n', positive=True, integer=True)
t = sp.Symbol('t')
y = sp.Function('y')(t)
yp = sp.Derivative(y, t)
ydp= sp.Derivative(y, t, t)
equation = sp.Eq(ydp + t*y, sp.exp(t + 1))
display(equation)

```

$$ty(t) + \frac{d^2}{dt^2}y(t) = e^{t+1}$$

In [9]:
```
solution = sp.dsolve(equation)
display(solution)

```

$$y(t) = C_1\,Ai\,(-t) + C_2\,Bi\,(-t)$$

The above is nice and compact, but there may be an insecure feeling about harboring unevaluated Airy functions in the solution. To correct that, assign 1 to the constants and try

In [13]:
```
from sympy import airyai
from sympy import airybi
from sympy import series
airyai(t)

```

Out[13]: $Ai\,(t)$

In [74]:
```
soln = series(airyai(t), t, 0, 1) + series(airybi(t), t, 0, 1)
soln

```

Out[74]: $$\frac{\sqrt[3]{3}\Gamma\left(\frac{1}{3}\right)}{2\pi} + \frac{3^{\frac{5}{6}}\Gamma\left(\frac{1}{3}\right)}{6\pi} + O\,(t)$$

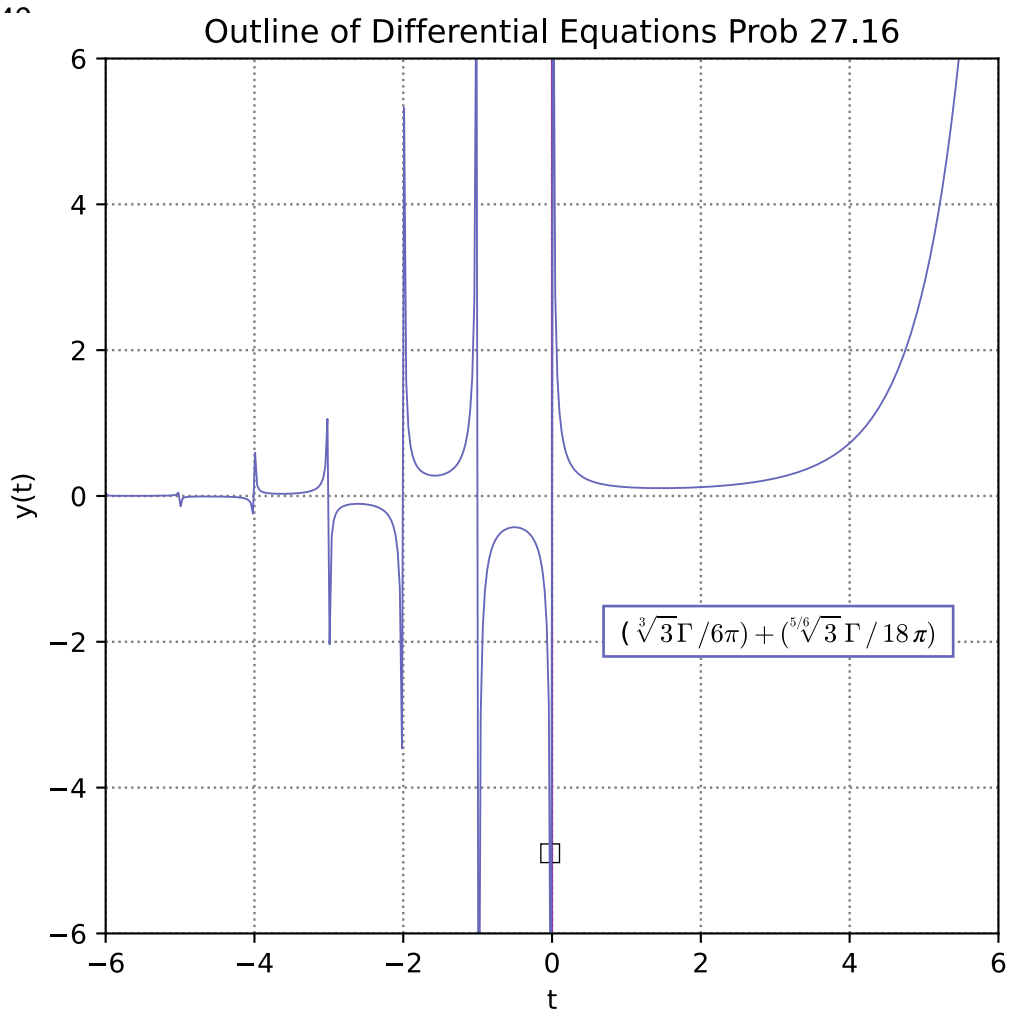
Trading Airy functions for gamma functions, which may or may not be more acceptable. The objective of the problem was to get an evaluation around $t = 0$, so it may be prudent to get a look at that area. (In the plot below a test coordinate is framed.)

In [243]:
```
soln = airyai(0).evalf(20) + airybi(0).evalf(20)
soln

```

Out[243]: 0.96995468133381797441

In [3]:
```
import numpy as np
from scipy.special import gamma as gm
import sympy as sp
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-10,10,700)

```

```
def f(x):
    return (3**(1/3)*gm(x)*(1/3))/(2*np.pi) + (3**(5/6)*gm(x)*(1/3))/(6*np.pi)
y = f(x)
yz = f(-0.025)
yw = f(0.025)
ys = f(0.003)

plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("t")
plt.ylabel("y(t)")
plt.title("Outline of Differential Equations Prob 27.16")
plt.rcParams['figure.figsize'] = [6, 9]
plt.rcParams['font.serif'] = ['Bookerly']
plt.rcParams['mathtext.fontset'] = 'cm'

ax = plt.gca()
#ax.axhline(y=0, color='#993399', linewidth=0.8)
ax.axvline(x=0, color='#993399', linewidth=0.8)
ratio = 0.98
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

plt.text(0.8, -2, " ( $\sqrt[3]{3}  \Gamma \, / 6 \pi) + (\sqrt[5/6]{3} \,\Gamma \,/ \,18 \,π) $ "
         bbox=dict(boxstyle="square", ec=('#6666BB'),fc=(1., 1., 1),))
xpts = np.array([-0.025])
ypts = np.array(yz)
wpts = np.array(yw)
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
#plt.plot(xpts, wpts, markersize=7, color='k', marker='s', \
         #mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.ylim(-6, 6)
plt.xlim(-6, 6)
plt.plot(x, y, linewidth = 0.7, color = '#6666BB')
plt.show()
print('f(-0.025) = '+str(yz)+'  [framed]')
print('f(0.003) = '+str(ys))

```




```
f(-0.025) = -4.9002753218079  [framed]
f(0.003) = 40.16033714044164
```

Above: The function is sensitive to being close to zero. Obviously, f(0) is not defined. Two small values are printed.

Technical note: since it is not feasible to chop a line in the middle of a Latex label, the PDF version may be cut off. The actual full line for the boxed label in the above plot is printed here for reference: " plt.text(0.8, -2, " ($\sqrt[3]{3}\Gamma /6\pi) + (\sqrt[5/6]{3}\,\Gamma / 18\,\pi)$ ", size=10,
bbox=dict(boxstyle="square", ec=('#6666BB'),fc=(1., 1., 1),)) " .

(Jupyter omits the line-breaking backslash.)

27.18 Find the general solution near $x = 2$ of $y'' - (x - 2)\,y' + 2y = 0$.

Another difficult problem to solve. Symbolab will not solve it, Maxima will not solve it, and the Wolfram Alpha solution is not alluring, containing the repellent erfi function. The sympy resource is sought again.

In [132]:
```
import sympy as sp

n = sp.Symbol('n', positive=True, integer=True)
x = sp.Symbol('x')
y = sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp= sp.Derivative(y, x, x)
eq = sp.Eq(ydp - (x - 2)*yp + 2*y, 0)
eq

```

Out[132]:
$$-(x-2)\frac{d}{dx}y(x) + 2y(x) + \frac{d^2}{dx^2}y(x) = 0$$

In [133]:
```
soln = sp.dsolve(eq)
soln

```

Out[133]:
$$y(x) = C_2\left(-\frac{x^4}{3} + \frac{2x^3}{3} - x^2 + 1\right) + C_1 x\left(-\frac{x^3}{4} + \frac{x^2}{2} - x + 1\right) + O\left(x^6\right)$$

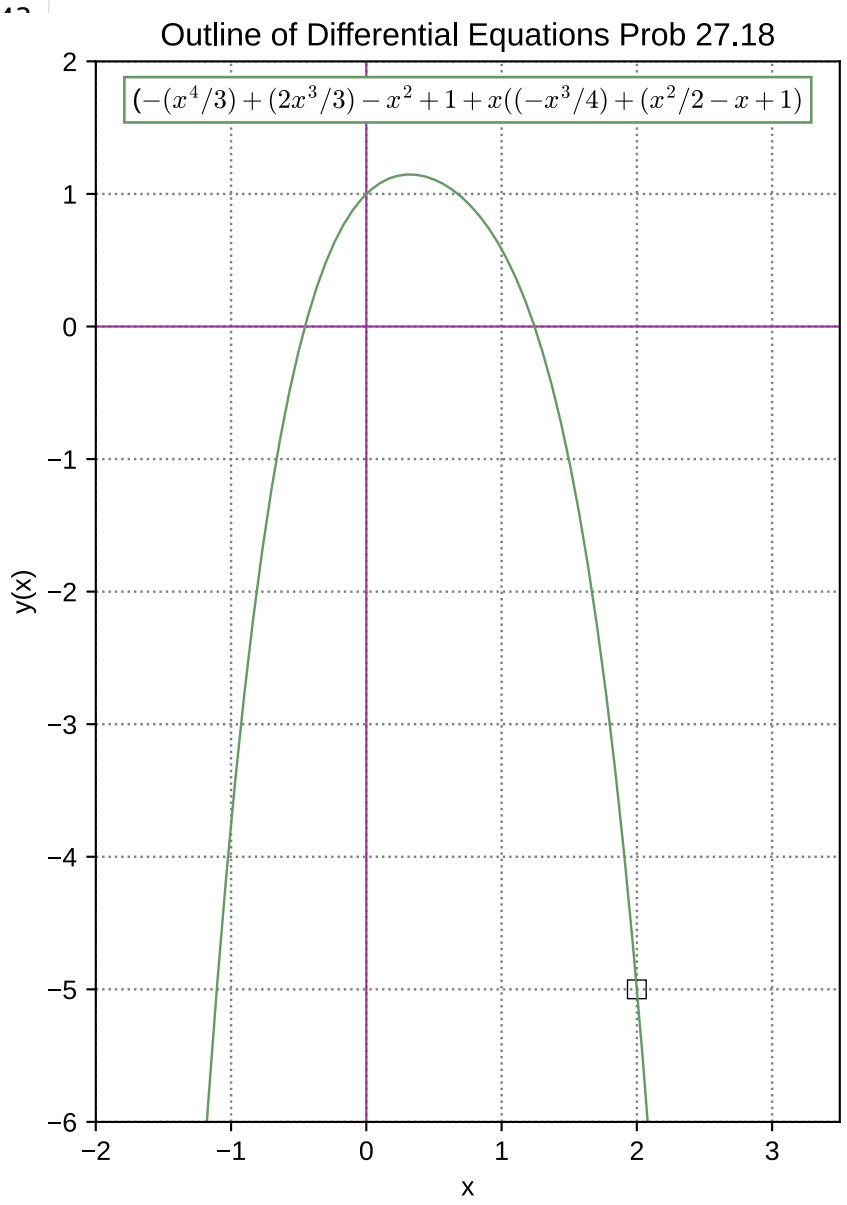
In [234]:
```
simplify( -(2**4/3) + (2*2**3/3) - (2**2) + 1) + 2*(-(2**3/4) + (2**2/2) - 2 + 1 )

```

Out[234]:
-5.0

To get a better understanding of the tasking for getting close to $x = 2$ a plot should be available.

In [82]:
```
import numpy as np
from scipy.special import gamma as gm
import sympy as sp
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-10,10,300)

#def f(x):
#    return (3**(1/3)*gm(x)*(1/3))/(2*np.pi) + (3**(5/6)*gm(x)*(1/3))/(6*np.pi)
#y = f(x)

y = -(x**4/3) + (2*x**3/3) - x**2 + 1 + x*(-(x**3/4) + (x**2/2) - x + 1)

plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.18")
plt.rcParams['figure.figsize'] = [5, 10]
plt.rcParams['font.sans-serif'] = ['Liberation Sans']
plt.rcParams['mathtext.fontset'] = 'cm'

ax = plt.gca()
ax.axhline(y=0, color='#993399', linewidth=0.8)
ax.axvline(x=0, color='#993399', linewidth=0.8)
ratio = 0.98
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

plt.text(-1.73, 1.65, "($-(x^4/3)+(2x^3/3)-x^2+1+x((-x^3/4)+(x^2/2-x+1)$",\
         size=10,bbox=dict(boxstyle="square",ec=('#669966'),fc=(1., 1., 1),))
xpts = np.array([2])
ypts = np.array([-5])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.ylim(-6, 2)
plt.xlim(-2, 3.5)
plt.plot(x, y, linewidth = 0.9, color = '#669966')
plt.show()

```

42



Nothing special seems to be happening around the point $x = 2$, so the party will pass on.

27.19 Find the general solution near $x = -1$ of $y'' + xy' + (2x - 1)\,y = 0$.

Another difficult problem to solve. Symbolab will not solve it, Maxima will not solve it, and the Wolfram Alpha solution is not inviting, containing the unfamiliar erfi function. Aid from sympy is needed again.

In [208]:
```
import sympy as sp

n = sp.Symbol('n', positive=True, integer=True)
x = sp.Symbol('x')
y = sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp= sp.Derivative(y, x, x)
eq = sp.Eq(ydp + x*yp + (2*x - 1)*y , 0)
eq

```

Out[208]:
$$x\frac{d}{dx}y(x) + (2x - 1)\,y(x) + \frac{d^2}{dx^2}y(x) = 0$$

In [209]:
```
soln = sp.dsolve(eq)
soln

```

Out[209]:
$$y(x) = C_2\left(-\frac{x^4}{24} - \frac{x^3}{3} + \frac{x^2}{2} + 1\right) + C_1 x\left(1 - \frac{x^3}{6}\right) + O\left(x^6\right)$$

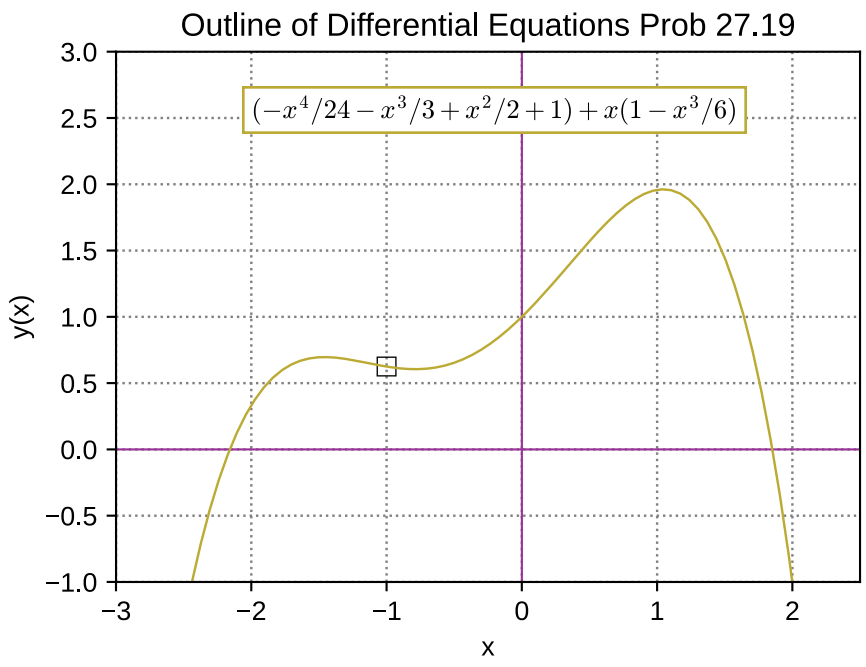
In [216]:
```
simplify((-1/24) + (1/3) + (1/2) + 1 + -1*(1 + 1/6))

```

Out[216]:
0.625

Above: The solution is evaluated at $x = -1$. To get a better understanding of the tasking for getting close to $x = -1$ a plot should be available, with the coordinate of interest framed.

In [218]:

```
import numpy as np
from scipy.special import gamma as gm
import sympy as sp
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-10,10,300)

#def f(x):
#    return (3**(1/3)*gm(x)*(1/3))/(2*np.pi) + (3**(5/6)*gm(x)*(1/3))/(6*np.pi)
#y = f(x)

y = -(x**4/24) - (x**3/3) + (x**2/2) + 1 + x*(1 - (x**3/6) )

plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.19")
plt.rcParams['figure.figsize'] = [7, 6]
plt.rcParams['font.sans-serif'] = ['Liberation Sans']
plt.rcParams['mathtext.fontset'] = 'cm'

ax = plt.gca()
ax.axhline(y=0, color='#993399', linewidth=0.8)
ax.axvline(x=0, color='#993399', linewidth=0.8)
ratio = 0.98
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

plt.text(-2,2.5,"$(-x^4/24-x^3/3+x^2/2+1) + x(1-x^3/6)$",size=10,\
         bbox=dict(boxstyle="square", ec=('#BBAA33'),fc=(1., 1., 1),))
xpts = np.array([-1])
ypts = np.array([0.625])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.ylim(-1, 3)
plt.xlim(-3, 2.5)
plt.plot(x, y, linewidth = 0.9, color = '#BBAA33')
plt.show()

```



The solution function appears uneventful around the point $x = -1$, so the problem will be concluded.

27.20 Find the general solution near $x = 1$ of $y'' + (x - 1)\, y = e^x$.

Neither Symbolab nor Maxima will solve the problem. Wolfram Alpha offers a solution with numerous Airy functions and ξ characters. Again, sympy becomes the solver of choice.

In [90]:
```
import sympy as sp
t = sp.symbols('t')
n = sp.Symbol('n', positive=True, integer=True)
x = sp.Symbol('x')
y = sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp= sp.Derivative(y, x, x)
eq = sp.Eq(ydp + (x - 1)*y , sp.exp(x))
eq

```

Out[90]:
$$(x - 1)\, y(x) + \frac{d^2}{dx^2} y(x) = e^x$$

In [91]:
```
soln = sp.dsolve(eq)
soln

```

Out[91]:
$$y(x) = C_1\, Ai\,(1 - x) + C_2\, Bi\,(1 - x)$$

In [92]:
```
from sympy import airyai
from sympy import airybi
from sympy import series
airyai(x)

```

Out[92]:
$$Ai\,(x)$$

The sympy solution is shown above, Airy functions including and consisting of Airy Ai and Bi. If the problem's concern is with the behavior around $x = -1$ that would relate to a function argument value of 2.

In [94]:
```
soln = series(airyai(1 - x), x, -1, 1) + series(airybi(1 - x), x, -1, 1)
soln

```

Out[94]:
$$Bi\,(2) + Ai\,(2) + O\,(x + 1; x \to -1)$$

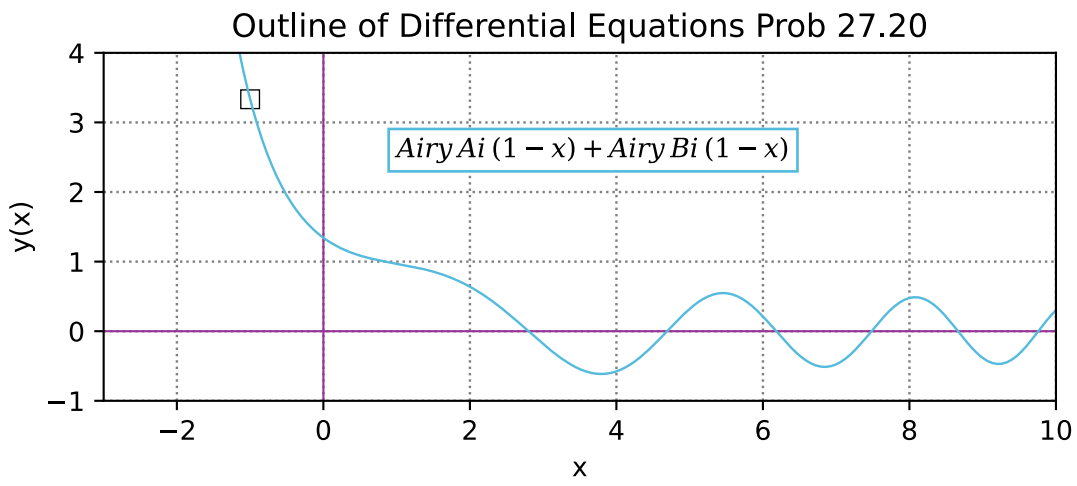
In [78]:
```
tval = airyai(2).evalf(20) + airybi(2).evalf(20)
tval

```

Out[78]:
$$3.3330191304014890894$$

The above cell suggests that there is no concern connected to the point $x = -1$.

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

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-10,10,300)
ary = []

def f(x):
    for k in x:
        k = sp.airyai(1 - k) + sp.airybi(1 - k)
        ary.append(k)

y = f(x)

plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.20")
plt.rcParams['figure.figsize'] = [11, 3]
#plt.rcParams['font.sans-serif'] = ['Liberation Sans']
plt.rcParams['mathtext.fontset'] = 'dejavuserif'

ax = plt.gca()
ax.axhline(y=0, color='#993399', linewidth=0.8)
ax.axvline(x=0, color='#993399', linewidth=0.8)
ratio = 0.95
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

plt.text(1,2.5,"$Airy\: Ai\:(1 - x) + Airy\: Bi\:(1 - x)$",size=10,\
```

```
        bbox=dict(boxstyle="square", ec=('#55BBDD'),fc=(1., 1., 1),))
xpts = np.array([-1])
ypts = np.array([3.333019])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
        mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.ylim(-1, 4)
plt.xlim(-3, 10)
plt.plot(x, ary, linewidth = 0.9, color = '#55BBDD')
plt.show()
#print(ary)

```



27.21 Solve the initial-value problem $y'' - (x - 2)\,y' + 2y = 0; \quad y(2) = 5,\ y'(2) = 60\,.$

Neither Symbolab nor Maxima will solve the problem. Wolfram Alpha offers a long solution including erfi functions. As before sympy remains the solver of choice.

In [16]:
```
import sympy as sp
t, c4, c5, c6 = sp.symbols('t c4, c5, c6')
n = sp.Symbol('n', positive=True, integer=True)
x = sp.Symbol('x')
y = sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp= sp.Derivative(y, x, x)
eq = sp.Eq(ydp - (x - 2)*yp + 2*y , 0)
eq

```

Out[16]:
$$-(x-2)\frac{d}{dx}y(x) + 2y(x) + \frac{d^2}{dx^2}y(x) = 0$$

In [17]:
```
soln = sp.dsolve(eq)
soln

```

Out[17]:
$$y(x) = C_2\left(-\frac{x^4}{3} + \frac{2x^3}{3} - x^2 + 1\right) + C_1 x\left(-\frac{x^3}{4} + \frac{x^2}{2} - x + 1\right) + O\left(x^6\right)$$

Above: a solution becomes available. And if a particular point in the domain caused curiosity, such as $x = 2$, it could be investigated as shown below.

In [18]:
```
solu = (-(x**4/3) + (2*x**3/3) - x**2 +1) + x*(-(x**3/4) + (x**2/2) - x + 1)
solu

```

Out[18]:
$$-\frac{x^4}{3} + \frac{2x^3}{3} - x^2 + x\left(-\frac{x^3}{4} + \frac{x^2}{2} - x + 1\right) + 1$$

In [19]:
```
solu2 = solu.subs(x , 2)
solu2

```

Out[19]:
$$-5$$

But the actual purpose of the problem was to handle the IVP part.

In [21]:
```
ics = {y.subs(x, 2): 5, yp.subs(x, 2): 60}
ics
```

Out[21]:
```
{y(2): 5, Subs(Derivative(y(x), x), x, 2): 60}
```

In [ ]:
```
ivp = dsolve(eq, ics=ics)
ivp
```

When the above cell is run, it throws an exception. The message says that it is not possible to solve the ODE with the set of initial conditions which is present. This is something of a setback, because it forces the initiative to Wolfram Alpha, which, at least, is able to solve the IVP. It is now necessary to see if it is possbile to become accustomed to the Wolfram Alpha style of IVP solution, complete with erfi function.

The entry line is:

!| solve[[y'' - (x - 2) y' + 2 y = 0], {y(2) = 5, y'(2) = 60}] |!

and the solution is:

$$y(x) = 5\left(-3\sqrt{2\pi}\,(x^2 - 4x + 3)\,\text{erfi}\left(\frac{x-2}{\sqrt{2}}\right) - x^2 + 4x + 6\,e^{1/2\,(x-2)^2}\,(x-2) - 3\right)$$

Unfortunately it is not possible to check the answer against the text, since the text answer ends with an ellipsis, and the recurrence pattern is not readily discernible.

The W|A entry line to get the derivative of the solution is:

!| D[5 * (-3 * sqrt(2 * pi) * (x^2 - 4 * x + 3) * erfi((x-2)/sqrt(2)) - x^2 + 4 * x + 6 * exp((1/2) * (x-2)^2) * (x - 2) - 3)] |!

Pursuing the Wolfram Alpha side of the solution process consists in only a couple of steps. On reconsideration, a route to a solution in sympy is glimpsed. A sequence of steps is documented below.

In [22]:
```
c1, c2 = sp.symbols('c1 c2')
jaz = c2*(-(x**4/3) + (2*x**3/3) - x**2 + 1) + c1*x*(-(x**3/4) + (x**2/2) - x +1)
jaz
```

Out[22]:
$$c_1 x\left(-\frac{x^3}{4} + \frac{x^2}{2} - x + 1\right) + c_2\left(-\frac{x^4}{3} + \frac{2x^3}{3} - x^2 + 1\right)$$

In the above cell the solution equation is copied by hand into a new instance, with a new name. It may have been possible to use the memory-resident version.

In [23]:
```
jazs = jaz.subs(x, 2)
jazs
```

Out[23]:
$$-2c_1 - 3c_2$$

In the above cell is an expression giving the values of c1 and c2 in terms of one another. It may be useful later.

In [24]:
```
jaz2 = jaz.diff(x)
jaz2
```

Out[24]:
$$c_1 x\left(-\frac{3x^2}{4} + x - 1\right) + c_1\left(-\frac{x^3}{4} + \frac{x^2}{2} - x + 1\right) + c_2\left(-\frac{4x^3}{3} + 2x^2 - 2x\right)$$

In the above cell is the differentiated version of the solution equation.

In [38]:
```
jaz2f = sp.simplify(2*c1*(-(3*2**2/4) +2 -1) + c1*(-(2**3/4) + (2**2/2) -2 + 1) + c2*(-(4*2**3/3)
jaz2f
```

Out[38]: $-5.0c_1 - 6.66666666666667c_2$

In the above cell is an expression evaluating the differentiated solution equation, with c1 and c2 set to 2, to capture the constants at the point $x = 2$. It is not possible to use the sympy substitution command in this case because, for some reason, it introduces new constants.

In [39]:
```
eq1 = sp.Eq(-2*c1 - 3*c2 ,5)
eq2 = sp.Eq(-5*c1 - 6.6666666*c2, 60)
res = sp.solve([eq1, eq2], (c1, c2))
res
```

Out[39]:
```
{c1: -88.0000000000000, c2: 57.0000000000000}
```

There is now enough information to solve for the constant values, which is what happens in the cell above.

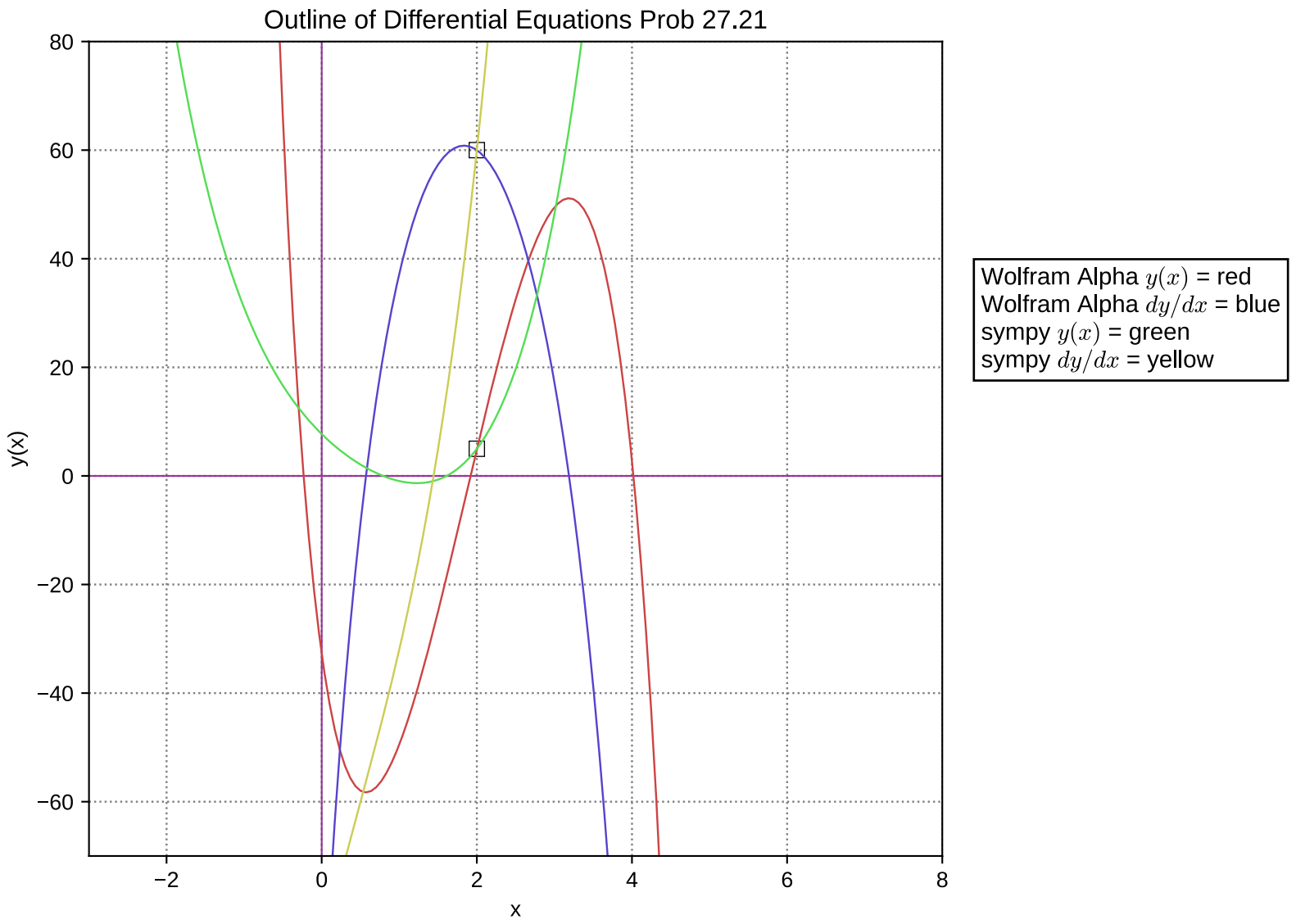
In [32]:
```
jazfin = jaz.subs({c1:-14.0540534, c2: 7.7027022})
jazfin
```

Out[32]: $-2.5675674x^4 + 5.1351348x^3 - 7.7027022x^2 - 14.0540534x\left(-\frac{x^3}{4} + \frac{x^2}{2} - x + 1\right) + 7.7027022$

Above: Installing the constants into the solution equation.

In [87]:
```
import numpy as np
import sympy as sp
from scipy import special
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-10,10,300)
y = 5*(-3*np.sqrt(2*np.pi)*(x**2 - 4*x + 3)*special.erfi((x-2)/\
    np.sqrt(2)) - x**2 + 4*x + 6*np.exp((1/2)*(x-2)**2)*(x - 2) - 3)
y1 = 10*(-3*np.sqrt(2*np.pi)*(x - 2)*special.erfi((x-2)/np.sqrt(2)) - \
         x + 6*np.exp((1/2)*(x-2)**2) + 2)
y2 = -2.5675674*x**4 + 5.1351348*x**3 -7.7027022*x**2 -14.0540534*x*\
    (-(x**3/4) + (x**2/2) -x +1) + 7.7027022
y3 = -88*x*(-(3*x**2/4) + x - 1) -88*(-(x**3/4) +(x**2/2) -x + 1) + 57*\
    (-(4*x**3/3) + 2*x**2 - 2*x)


plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.21")
plt.rcParams['figure.figsize'] = [7, 7]
plt.rcParams['font.sans-serif'] = ['Liberation Sans']
plt.rcParams['mathtext.fontset'] = 'cm'

ax = plt.gca()
ax.axhline(y=0, color='#993399', linewidth=0.8)
ax.axvline(x=0, color='#993399', linewidth=0.8)
ratio = 0.07
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

textstr = '\n'.join((
    r'Wolfram Alpha $y(x)$ = red',
    r'Wolfram Alpha $dy/dx$ = blue',
    r'sympy $y(x)$ = green',
    r'sympy $dy/dx$ = yellow'))

plt.text(8.5, 20, textstr,size=11,\
         bbox=dict(boxstyle="square", ec=('k'),fc=(1., 1., 1),))
xpts = np.array([2, 2])
ypts = np.array([5, 60])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.ylim(-70, 80)
plt.xlim(-3, 8)
```

```
49
50 plt.plot(x, y, linewidth=0.9, color='#CC4444')
51 plt.plot(x, y1, linewidth=0.9, color='#5544CC')
52 plt.plot(x, y2, linewidth=0.9, color='#55DD55')
53 plt.plot(x, y3, linewidth=0.9, color='#CCCC55')
54 plt.show()
55
```



Solving an IVP is supposed to focus in on a specific member of a solution family. How Wolfram Alpha and sympy came up with different solutions to the IVP, both of which, superficially at least, seem to qualify, is a condition currently without explanation. Though unrelated to the larger mystery, it should be remembered in the case of sympy, there was a series remainder that was discarded.

27.22 Solve $y'' + xy' + (2x - 1)\,y = 0; \quad y(-1) = 2,\ y'(-1) = -2\,.$

The current problem is not solvable by sympy, nor by Maxima. Wolfram Alpha can solve it, though the solution is somewhat long. The entry for the solve looks like:

!| solve[[y'' + x y' + (2 x -1) y = 0],{y(-1) = 2, y'(-1) = -2}] |!

and the result looks like:

$$y(x) = e^{-1/2(x-4)x\,-\,29/2}\left(-43e^{9/2}\sqrt{2\pi}\,(x^2 - 8x + 15)\,\text{erfi}\left(\frac{x-4}{\sqrt{2}}\right) - 43e^{9/2}\sqrt{2\,\pi}\,\text{erfi}\left(\frac{5}{\sqrt{2}}\right) + (x^2 - 8x + 15) + 18\,e^{17}\,(x^2 - 8x + 15) + 86\,e^{(1/2)(x-5)^2+x}\,(x-4)\right)$$

One thing that would seem sensible would be to get a numerical evaluation of the function at the first initial value coordinate. Unfortunately, this cannot be done in Wolfram Alpha because the number of characters to enter the expression would exceed the number allowed. What can be done is to cut the primary expression into three parts, evaluate each, and then join the numerical results.

① !| [e^(-(1/2) ((-1) - 4) (-1) - (29/2)) (-43 e^(9/2) sqrt(2 pi) ((-1)^2 - 8 (-1) + 15) erfi(((-1) - 4)/sqrt(2)))] |!

substituting $x = -1$ in the above expression equals 431.88324285146157.

② !| e^(-(1/2) ((-1) - 4) (-1) - (29/2)) (-43 e^(9/2) sqrt(2 pi) erfi(5/sqrt(2)) ((-1)^2 -8 (-1) +15)) |!

substituting $x = -1$ in the above expression equals -431.88324285146157.

③ !| e^(-(1/2) ((-1) - 4) (-1) - (29/2)) (18 e^17 ((-1)^2 - 8 (-1) +15) + 86e^((1/2) ((-1) - 5)^2 + (-1)) ((-1) - 4)) |!

substituting $x = -1$ in the above expression equals 2.

Therefore the Wolfram Alpha solution meets the first initial condition. Doing the same for the first derivative is not possible in Wolfram Alpha, which declines to show the first derivative. This part of the task can be accomplished by Maxima, as shown below.

The main equation for $y(x)$ is broken into three parts by Maxima. The fuchsia coloring in the cell below indicates the factor in common among the three which needs to be distributed over all three parts. At the bottom is the float value for the first section. Details of the second and third sections are similar.

```
/* The following two cells find the derivative of the 1st part of the expression for f(x).*/
r1(x):= %e^(-(1/2)·(x - 4)·x - (29/2))·(-43·%e^(9/2)·sqrt(2·%pi)·(x^2 - 8·x + 15)·erfi((x- 4)/sqrt(2)))
```

$$r1(x) := \%e^{\left(-\frac{1}{2}\right)(x-4)x-\frac{29}{2}}\left(-43\right)\%e^{9/2}\sqrt{2\,\pi}\left(x^2-8\,x+15\right)\operatorname{erfi}\left(\frac{x-4}{\sqrt{2}}\right)$$

```
r1p: diff(r1(x), x, 1)
```

$$-86\left(x^2-8\,x+15\right)\%e^{-\frac{(x-4)x}{2}+\frac{(x-4)^2}{2}-10}-43\sqrt{2}\sqrt{\pi}\operatorname{erfi}\left(\frac{x-4}{\sqrt{2}}\right)\left(-\frac{x}{2}-\frac{x-4}{2}\right)\left(x^2-8\,x+15\right)\%e^{-\frac{(x-4)x}{2}-10}-43\sqrt{2}\sqrt{\pi}\operatorname{erfi}\left(\frac{x-4}{\sqrt{2}}\right)$$

$$(2\,x-8)\,\%e^{-\frac{(x-4)x}{2}-10}$$

```
/* The following cell evaluates the derivative of the 1st part at the point x = -1. */
at (r1p, [x=-1])
```

$$387\,2^{7/2}\,\%e^{-\frac{25}{2}}\sqrt{\pi}\operatorname{erfi}\left(\frac{5}{\sqrt{2}}\right)-215\,2^{3/2}\,\%e^{-\frac{25}{2}}\sqrt{\pi}\operatorname{erfi}\left(\frac{5}{\sqrt{2}}\right)-2064$$

```
float(%)
```

```
-948.3016226337268
```

The sum of the three section values for the total value of the derivative at $x = -1$ is:

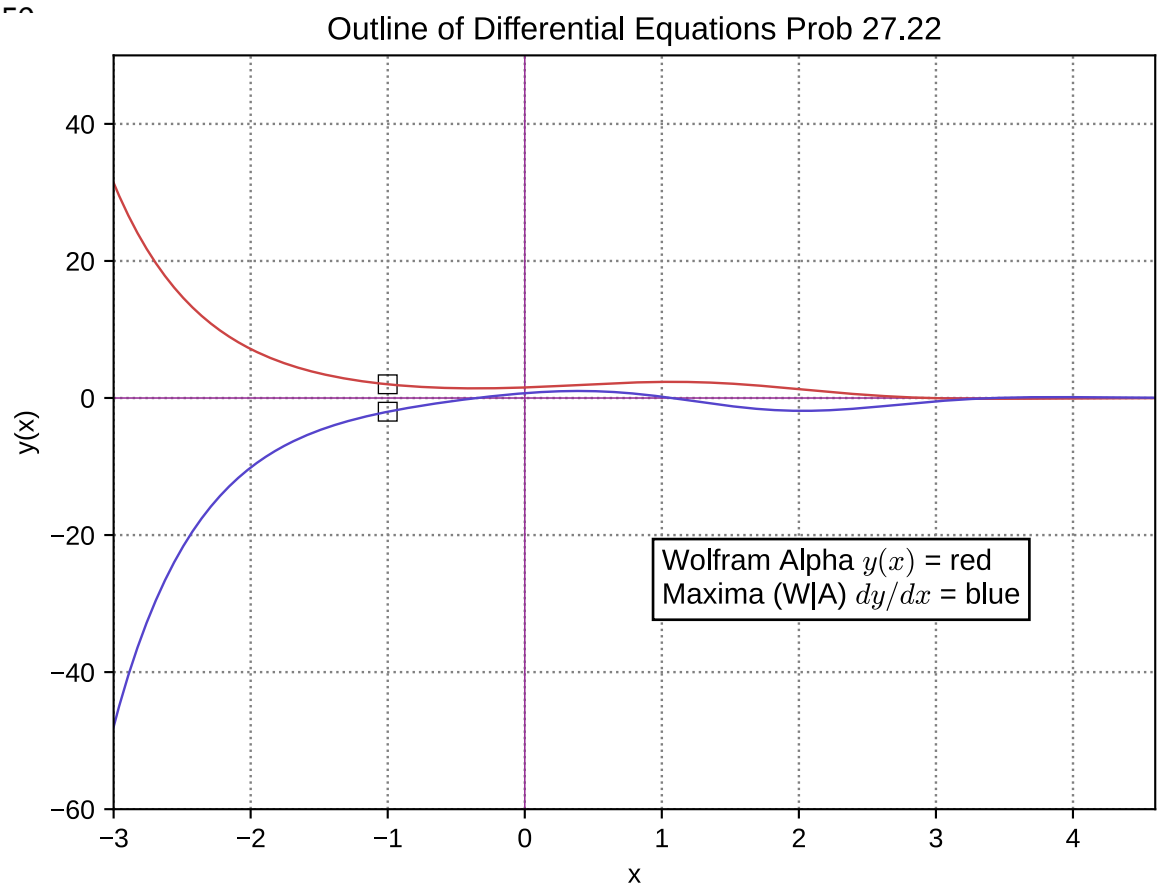
-948.3016226337268 + -1115.698377366273 + 2062 = -1.999999999999773

very close to the -2 value expected.

In [10]:

```
import numpy as np
import sympy as sp
from scipy import special
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-5,5,300)
y = np.exp(-(1/2)*(x - 4)*x - (29/2))*(-43*np.exp(9/2)*np.sqrt(2*np.pi)\
            *(x**2 - 8*x + 15)*special.erfi((x-4)/np.sqrt(2)) - 43*np.exp(9/2)*\
            np.sqrt(2*np.pi)*special.erfi(5/np.sqrt(2))*(x**2 - 8*x + 15) + 18*\
            np.exp(17)*(x**2 - 8*x + 15)+ 86*np.exp(((1/2)*(x - 5)**2) + x) *
            (x -4))
y1 = (-86*np.exp(9/2)*(x**2 - 8*x + 15)*np.exp((x - 4)**2/2) - 43*np.sqrt(2*np.pi)\
            *np.exp(9/2)*special.erfi((x - 4)/np.sqrt(2)) * (2*x - 8) + 18*\
            np.exp(17)*(2*x - 8) - 43*np.sqrt(2)*np.exp(9/2)*np.sqrt(np.pi)*special\
            .erfi(5/np.sqrt(2))*(2*x - 8) + 86*np.exp(x + ((x - 5)**2)/2)\
            *np.log(np.e)*(x - 4)**2 + 86*np.exp(x + (x - 5)**2/2))*np.exp(-((x - 4)\
            *x)/2 - 29/2) + (-x/2 - (x - 4)/2)*(-43*np.sqrt(2*np.pi)*np.\
            exp(9/2)*special.erfi((x - 4)/np.sqrt(2))*(x**2 - 8*x + 15) + 18*\
            np.exp(17)*(x**2 - 8*x + 15) - 43*np.sqrt(2)*np.exp(9/2)*np.sqrt\
            (np.pi)*special.erfi(5/np.sqrt(2))*(x**2 - 8*x + 15) + 86*np.exp\
            (x +((x - 5)**2)/2)*(x - 4))*np.exp(-((x - 4)*x)/2 - 29/2)

plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.22")
plt.rcParams['figure.figsize'] = [7, 7]
plt.rcParams['font.sans-serif'] = ['Liberation Sans']
plt.rcParams['mathtext.fontset'] = 'cm'

ax = plt.gca()
ax.axhline(y=0, color='#993399', linewidth=0.6)
ax.axvline(x=0, color='#993399', linewidth=0.6)
ratio = 0.05
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

textstr = '\n'.join((
    r'Wolfram Alpha $y(x)$ = red',
    r'Maxima (W|A) $dy/dx$ = blue',
    ))
```

```
45
46 plt.text(1, -30, textstr,size=11,\
47          bbox=dict(boxstyle="square", ec=('k'),fc=(1., 1., 1),))
48 xpts = np.array([-1, -1])
49 ypts = np.array([-2, 2])
50 plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
51          mfc='none', linestyle = 'none', markeredgewidth=0.5)
52 plt.ylim(-60, 50)
53 plt.xlim(-3, 4.6)
54
55 plt.plot(x, y, linewidth=0.9, color='#CC4444')
56 plt.plot(x, y1, linewidth=0.9, color='#5544CC')
57 plt.show()
58
```



27.24 Use the method outlined in Problem 27.23 to solve $y'' - 2xy = 0; \quad y(2) = 1,\ y'(2) = 0$.

This problem is a step full into Airyland. The entry line in Wolfram Alpha is:

!| solve[[y'' - 2 x y = 0],{y(2) = 1, y'(2) = 0}] |!

and the return is:

$$y(x) = \frac{\text{Ai}'(2\sqrt[3]{2})\,\text{Bi}(\sqrt[3]{2}\,x) - \text{Bi}'(2\sqrt[3]{2})\,\text{Ai}(\sqrt[3]{2}\,x)}{\text{Ai}'(2\sqrt[3]{2})\,\text{Bi}(2\sqrt[3]{2}) - \text{Ai}(2\sqrt[3]{2})\,\text{Bi}'(2\sqrt[3]{2})}$$

On substituting the value $x = 2$, W|A compliantly produces the value 1, in agreement with the first initial condition.

Wolfram Alpha is also ready to provide a derivative for the above equation (if care is taken in putting in the entry):

!| D[(Ai' (2 2^(1/3)) Bi (2^(1/3) x) - Bi' (2 2^(1/3)) Ai (2^(1/3) x))/(Ai' (2 2^(1/3)) Bi (2 2^(1/3)) - Ai (2 2^(1/3)) Bi' (2 2^(1/3)))] |!

for which the answer is:

$$y'(x) = \sqrt[3]{2}\,\pi\left(\text{Bi}'\left(2\sqrt[3]{2}\right)\text{Ai}'\left(\sqrt[3]{2}x\right) - \text{Ai}'\left(2\sqrt[3]{2}\right)\text{Bi}'\left(\sqrt[3]{2}x\right)\right)$$

The above is listed as an alternate form of the answer.

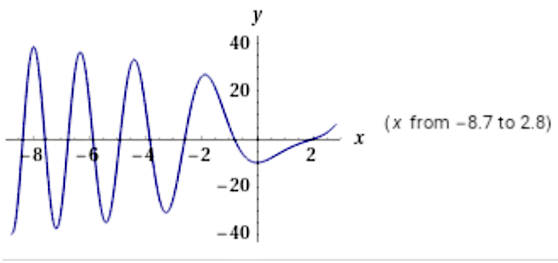
The small plot for the derivative $y'(x)$ does not look like anything which is easily achievable in matplotlib or Maxima.

In [ ]:

```
1
2
3
4
5
6
7
8
```

```
9
```

Plots



Following are a number of cells which illustrate various attempts to work the problem, with varying degrees of success, none completely successful. To see a succinct performance, based on the green cell above, drop down to the next yellow cell.

Maxima note: The trial is made to differentiate the W|A solution in Maxima, but the effort is not successful. An error condition is triggered, with the tag:

expt: undefined: 0 to a negative exponent.

Review of development posts shows that this error was previously associated with a bug in Maxima, but whether it is a bug in the present circumstance or is a legitimate result of faulty input, is not known.

Now back to the Wolfram Alpha channel. It turns out that W|A is willing to substitute the first initial condition into the solution to verify it. The entry for verification is:

!| ReplaceAll[[(Ai'(2 2^(1/3)) Bi(2^(1/3) x) - Bi'(2 2^(1/3)) Ai(2^(1/3) x))/(Ai'(2 2^(1/3)) Bi(2 2^(1/3)) - Ai(2 2^(1/3)) Bi'(2 2^(1/3)))],{x -> 2}] |!

and the result is the desired: 1.

Hitting the initial condition algebraically is a good thing, surely, but the problem solution submitted by Wolfram Alpha comes to a dead end when it turns out that it cannot be plotted. Neither matplotlib nor Maxima can find enough substance in the equation to grasp firmly enough to make a trace. This is where sympy asserts authority. The next cells show a path for determining the exact solution.

In [32]:
```
import sympy as sp
from sympy import airyai
from sympy import airybi
x, c1, c2 =sp.symbols('x c1 c2')
y =sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp = sp.Derivative(y, x, x)
ode = sp.Eq(ydp - 2*x*y , 0)
ode

```

Out[32]:
$$-2xy(x) + \frac{d^2}{dx^2}y(x) = 0$$

In [33]:
```
sol = sp.dsolve(ode, y)
sol

```

Out[33]:
$$y(x) = C_1 Ai\left(\sqrt[3]{2}x\right) + C_2 Bi\left(\sqrt[3]{2}x\right)$$

Exhibiting sympy coming up with its version of the solution, a much simpler one than Wolfram Alpha found. But all that sympy can find, unfortunately, is the general solution. A crash occurs when attempting to force a particular solution using initial conditions. This does not mean that the particular solution cannot be found; it merely requires a slightly different course.

In [35]:
```
tval = airyai(2**(1/3)*2).evalf(15)
tval

```

Out[35]:
0.0152127253988192

In [36]:
```
tval = airybi(2**(1/3)*2).evalf(15)
tval

```

Out[36]:
6.67183091212593

0.0152127253988192*C_1* + *6.67183091212593*C_2 = 1

In [38]:
```
from sympy import *

c1, c2 = symbols('c1 c2')
eq1 = Eq(0.0152127253988192*c1 + 6.67183091212593*c2, 1)
eq2 = Eq(-0.06688969289497138*c1 + 5.166535633636507*c2, 0)
res = solve([eq1, eq2], (c1, c2))
print(res)

```

```
{c1: 9.84338625554559, c2: 0.127439570801255}
```

Above are the determination cells for the constant values, which make the particular solution possible. But there is a slight gap in the effort to get all bases covered, because whereas the exact form of the solution equation has been revealed, the derivative has not.

Some of the information for determination of the constant values comes from Maxima. The cell below shows the applicable section where one input value for c_2 is reported.

```
p(x):= 2^(1/3)·airy_bi(x)
```

$$p(x) := 2^{1/3}\,\text{airy\_bi}(x)$$

```
pp: diff(p(x),x)
```

$$2^{1/3}\,\text{airy\_dbi}(x)$$

```
at(%, [x=2])
```

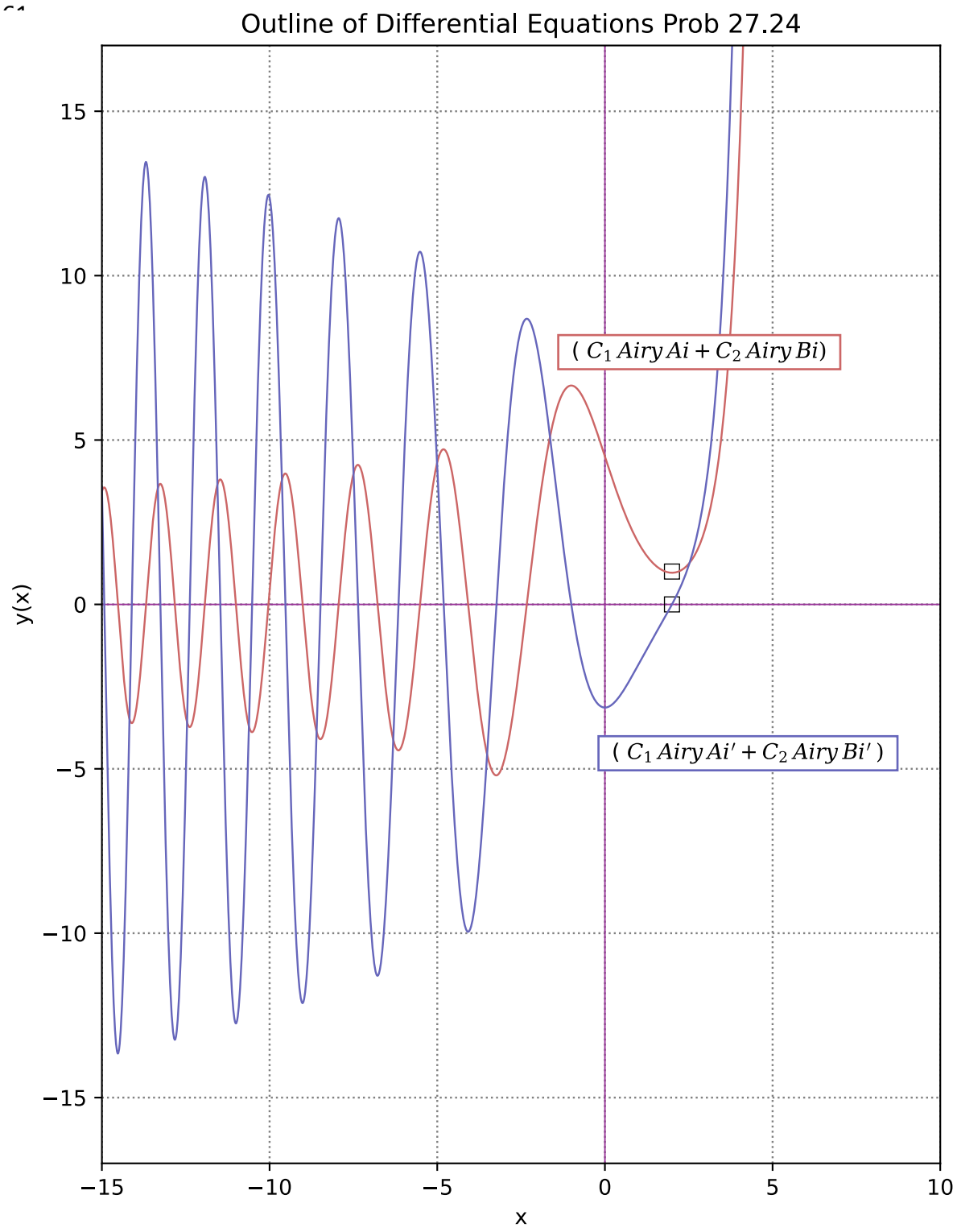
$$2^{1/3}\,\text{airy\_dbi}(2)$$

```
float(%)
```

5.166535633636507

In [25]:
```
import numpy as np
import sympy as sp
from sympy import airyai
from sympy import airybi
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = sp.symbols('x')
x2 = 9.84338625554559*2**(1/3)*airyai(x) + 0.127439570801255*2**(1/3)*airybi(x)
x4 = 9.84338625554558*2**(1/3)*sp.diff(airyai(x)) + 0.127439570801255*2**(1/3)*sp.diff(airybi(x))

xt = np.arange(-15, 10, 0.01)
yt2 = np.zeros(len(xt))
yt4 = np.zeros(len(xt))

for n in range(0,len(xt)):
    yt2[n] = sp.N(x2.subs(x, xt[n]))
    yt4[n] = sp.N(x4.subs(x, xt[n]))

plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.24")
plt.rcParams['figure.figsize'] = [7, 7]
plt.rcParams['font.serif'] = ['Bookerly']
plt.rcParams['mathtext.fontset'] = 'dejavuserif'

ax = plt.gca()
ax.axhline(y=0, color='#993399', linewidth=0.7)
ax.axvline(x=0, color='#993399', linewidth=0.7)
ratio = 0.98
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

plt.text(0, -4.7, " ( $C_1\:Airy \: Ai' + C_2 \: Airy \: Bi'\:) $ ", size=10,\
         bbox=dict(boxstyle="square", ec=('#6666BB'),fc=(1., 1., 1),))
plt.text(-1.2, 7.5, " ( $C_1\: Airy \:Ai + C_2 \:Airy \:Bi) $ ", size=10,\
         bbox=dict(boxstyle="square", ec=('#CC6666'),fc=(1., 1., 1),))
xpts = np.array([2, 2])
ypts = np.array([1, 0])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
```

```
plt.ylim(-17, 17)
plt.xlim(-15, 10)
plt.plot(xt, yt2, linewidth = 0.9, color = '#CC6666')
plt.plot(xt, yt4, linewidth = 0.9, color = '#6666BB')
plt.show()
```



In the plot above the failure to nail the first initial condition is clear. Maxima says that instead of 1, the function has a value of only 0.9626798668262899. This is disappointing especially when considering that the Wolfram Alpha solution equation hit the same initial condition exactly.

However, in its favor, the plot does resemble the one from Wolfram Alpha.

Marking the start of the short version of the problem solution.

In [36]:

```
import sympy as sp
x = sp.Symbol('x')
y = sp.Function('y')(x)
yp = sp.Derivative(y, x)
ydp= sp.Derivative(y, x, x)
equation = sp.Eq(ydp - 2*x*y, 0)
display(equation)
```

$$-2xy(x) + \frac{d^2}{dx^2}y(x) = 0$$

In [37]:
```
ics = {y.subs(x, 2): 1, yp.subs(x, 2): 0}
ics
```

Out[37]:
```
{y(2): 1, Subs(Derivative(y(x), x), x, 2): 0}
```

In [38]:
```
ivp = sp.dsolve(equation, ics=ics)
ivp
```

Out[38]:

$$y(x) = \frac{Ai\left(\sqrt[3]{2}x\right) Bi'\left(2\cdot\sqrt[3]{2}\right)}{Ai\left(2\cdot\sqrt[3]{2}\right) Bi'\left(2\cdot\sqrt[3]{2}\right) - Ai'\left(2\cdot\sqrt[3]{2}\right) Bi\left(2\cdot\sqrt[3]{2}\right)} - \frac{Ai'\left(2\cdot\sqrt[3]{2}\right) Bi\left(\sqrt[3]{2}x\right)}{Ai\left(2\cdot\sqrt[3]{2}\right) Bi'\left(2\cdot\sqrt[3]{2}\right) - Ai'\left(2\cdot\sqrt[3]{2}\right) Bi\left(2\cdot\sqrt[3]{2}\right)}$$

In [39]:
```
sp.checkodesol(equation, ivp)
```

Out[39]:
```
(True, 0)
```

The answer above agrees with what Wolfram Alpha came up with for $y(x)$.

The derivative of the solution function may be a train wreck anyway you look at it, but the output to the three cells below is still disappointing. In view of sympy's non-participation, the Wolfram Alpha version of the derivative will remain the referenced one.

In [2]:
```
from sympy import *
from sympy import besselj, jn
from sympy import bessely, yn
from sympy import airyai
from sympy import airybi
x = Symbol('x')
y = Function('y')(x)
yp = Derivative(y, x)
ydp= Derivative(y, x, x)

```

In [11]:
```
equation = ((airyai(x)*diff(airybi(x), x))-(airybi(x)*diff(airyai(x), x))*2**(5/3))/((airyai(x)*di
                                                    (airybi(x)*diff(airyai(x), x)))*2**(8/3))
equation
```

Out[11]:

$$\frac{Ai\left(x\right) Bi'\left(x\right) - 3.1748021039364 Ai'\left(x\right) Bi\left(x\right)}{6.3496042078728 Ai\left(x\right) Bi'\left(x\right) - 6.3496042078728 Ai'\left(x\right) Bi\left(x\right)}$$

In [13]:
```
equationp = (Derivative(equation, x))
```

Out[13]:

$$\frac{d}{dx}\frac{Ai\left(x\right) Bi'\left(x\right) - 3.1748021039364 Ai'\left(x\right) Bi\left(x\right)}{6.3496042078728 Ai\left(x\right) Bi'\left(x\right) - 6.3496042078728 Ai'\left(x\right) Bi\left(x\right)}$$

Wolfram Mathworld has the series expressions for the Airies, so, in keeping with the theme of the chapter (series form solutions), the answer for $y(x)$ is:

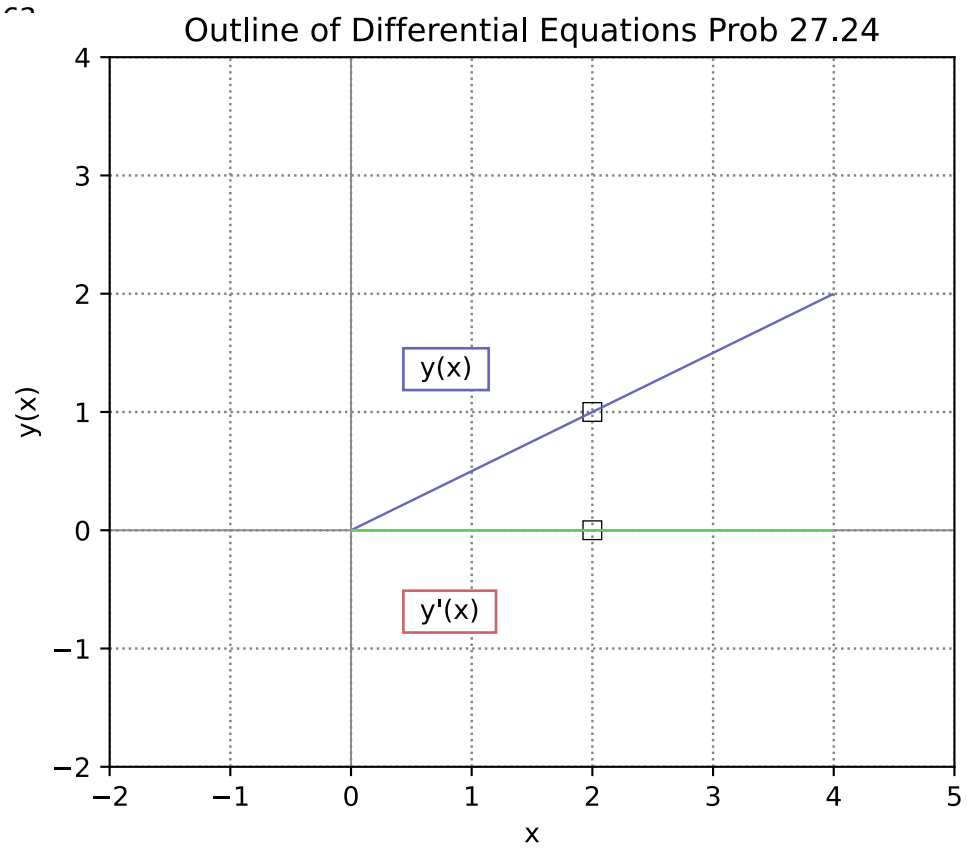
$$y(x) = C_1 \sqrt[3]{2x}\left(\frac{1}{3^{2/3}\pi}\sum_{n=0}^{\infty}\frac{\Gamma\left(\frac{1}{3}(n+1)\right)}{n!}\right)\left(3^{1/3}\,z\right)^n \sin\left[\frac{2\,(n+1)\,\pi}{3}\right] +$$

$$C_2 \sqrt[3]{2x}\left(\frac{1}{3^{1/6}\pi}\sum_{n=0}^{\infty}\frac{\Gamma\left(\frac{1}{3}(n+1)\right)}{n!}\right)\left(3^{1/3}\,z\right)^n \left|\sin\left[\frac{2\,(n+1)\,\pi}{3}\right]\right|$$

The opportunity now presents itself to check the initial conditions as plotted using the abbreviated procedure.

In [5]:
```
import numpy as np
import sympy as sp
from sympy import airyai
from sympy import airybi
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = sp.symbols('x')
x5 = ((2**(5/3)/2**(8/3))*(sp.diff(sp.airyai(x),x)*sp.airybi(x)*x -\
```

```
              sp.diff(sp.airybi(x),x)*sp.airyai(x)*x))/\
           (sp.diff(sp.airyai(x),x)*sp.airybi(x) - (sp.airyai(x))*sp.diff(sp.airybi(x),x))

x6 = (4*(sp.diff(sp.airyai(x),x)*(sp.diff(sp.airybi(x),x))*x -\
              sp.diff(sp.airybi(x),x)*sp.diff(sp.airyai(x),x)*x))/\
           8*(sp.diff(sp.airyai(x),x)*sp.airybi(x) - (sp.airyai(x))*sp.diff(sp.airybi(x),x))

x7 = (2)**(1/3)*sp.pi*((sp.diff(sp.airybi(x),x)*2**(4/3)*sp.diff(sp.airyai(x),x)*2**(1/3)*x)-\
           (sp.diff(sp.airyai(x),x)*2**(4/3)*sp.diff(sp.airybi(x),x)*2**(1/3)*x))

xt = np.arange(0, 4, 0.01)
yt5 = np.zeros(len(xt))
yt6 = np.zeros(len(xt))
yt7 = np.zeros(len(xt))

for n in range(0,len(xt)):
    yt5[n] = sp.N(x5.subs(x, xt[n]))
    yt6[n] = sp.N(x6.subs(x, xt[n]))
    yt7[n] = sp.N(x7.subs(x, xt[n]))

plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 27.24")
plt.rcParams['figure.figsize'] = [5,5]
plt.rcParams['font.serif'] = ['Bookerly']
plt.rcParams['mathtext.fontset'] = 'dejavuserif'

ax = plt.gca()
ax.axhline(y=0, color='#888888', linewidth=0.7)
ax.axvline(x=0, color='#888888', linewidth=0.7)
ratio = 0.98
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)

plt.text(0.5, 1.3, " y(x) ", size=10,\
         bbox=dict(boxstyle="square", ec=('#6666BB'),fc=(1., 1., 1),))
plt.text(0.5, -0.75, " y'(x) ", size=10,\
         bbox=dict(boxstyle="square", ec=('#CC6666'),fc=(1., 1., 1),))
xpts = np.array([2, 2])
ypts = np.array([1, 0])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.ylim(-2, 4)
plt.xlim(-2, 5)
plt.plot(xt, yt5, linewidth = 0.9, color = '#6666BB')
plt.plot(xt, yt6, linewidth = 0.9, color = '#CC6666')
plt.plot(xt, yt7, linewidth = 0.9, color = '#66CC66')
plt.show()
```



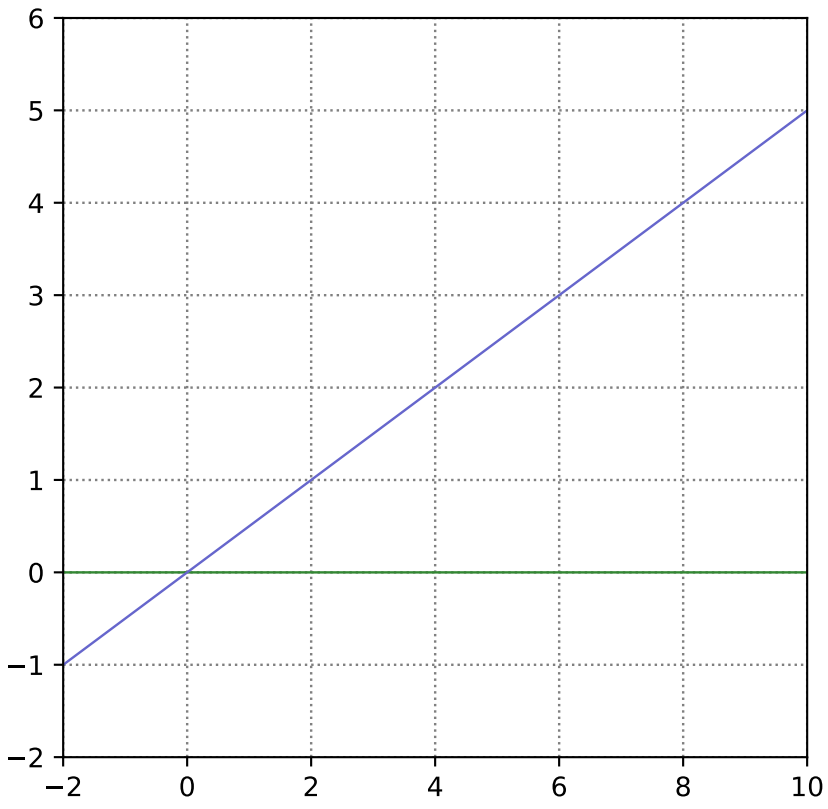
Above: Quite the boring-looking plot, but the initial values are delivered exactly.

Below: To check the integrity of the airy forms resident in sympy, the scipy versions are tried.

In [23]:

```
import numpy as np
from scipy import special
import matplotlib.pyplot as plt
%config InlineBackend.figure_formats = ['svg']

```

```
x = np.linspace(-15, 1000, 2000)

ai, aip, bi, bip = special.airy(x)

y2 = (2)**(1/3)*np.pi*((bip*2**(4/3)*aip*2**(1/3)*x)-\
            (aip*2**(4/3)*bip*2**(1/3)*x))
y3 = (2**(5/3)/2**(8/3))*(aip*bi*x -\
              bip*ai*x)/(aip*bi - ai*bip)
plt.rcParams['figure.figsize'] = [5,5]
plt.ylim(-2, 6)
plt.xlim(-2, 10)
plt.plot(x, y2, linewidth = 0.9, color = '#338833')
plt.plot(x, y3, linewidth = 0.9, color = '#6666CC')
plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)

plt.show()


```



Above: Apparently the sympy Airies are sound. However, something is still fishy about these last plots. It would be nice to see something along the line of the Wolfram Alpha plot.