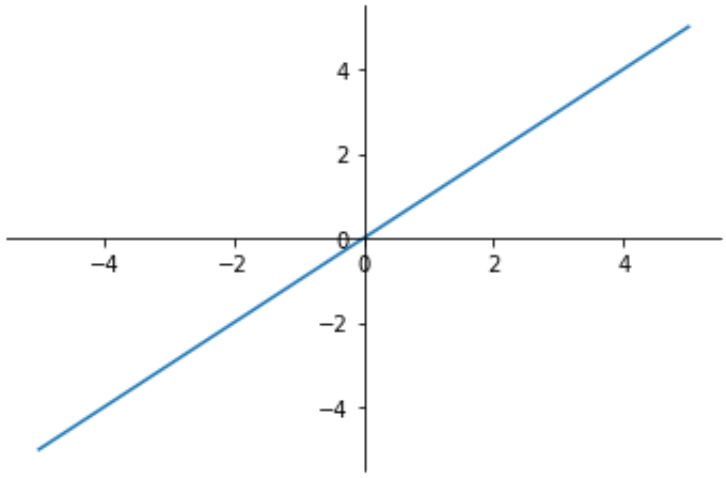
The antiderivative of a function f is a function F such that $F'(x) = f(x)$. For example, suppose that $f(x) = x$, as in the diagram below.

In [28]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = x

ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y)
```

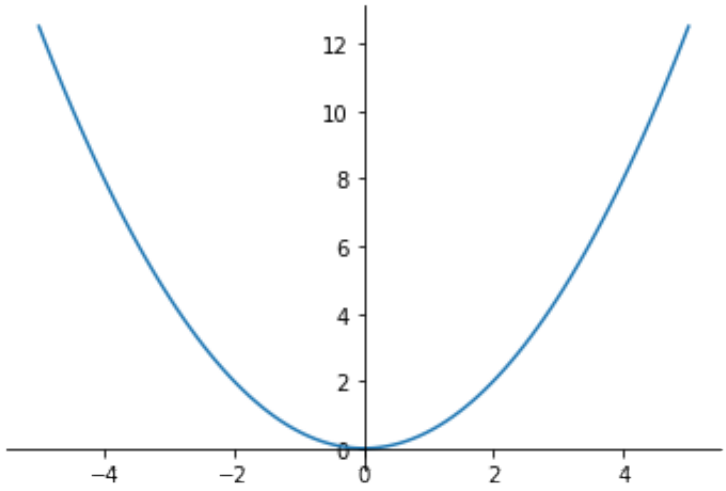
Out[28]:
```
[<matplotlib.lines.Line2D at 0x11cd1bf50>]
```



Then the anti-derivative of $f(x) = x$ is $F(x) = \frac{x^2}{2}$, with the graph displayed below.

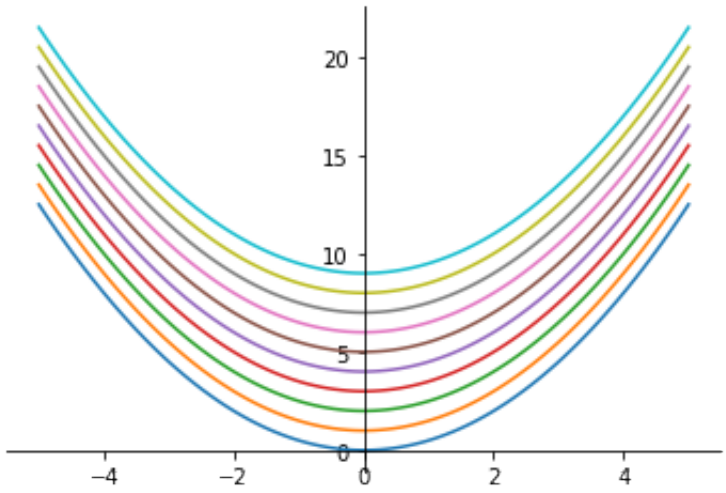
In [29]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = x**2/2

ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y)
```

Out[29]:
```
[<matplotlib.lines.Line2D at 0x11cdedc50>]
```



Is $F(x) = x^2/2$ the only anti-derivative of $f(x) = x$? No, because $F(x) = \frac{x^2}{2} + 100$ also does the job since $F'(x) = x + 0 = x$, where we have used the fact that the derivative of any constant is 0. More generally, for any constant C, $F(x) = \frac{x^2}{2} + C$ is an anti-derivative of $f(x) = x^2$. For visual clarity, we plot a few of these functions for different values of C.

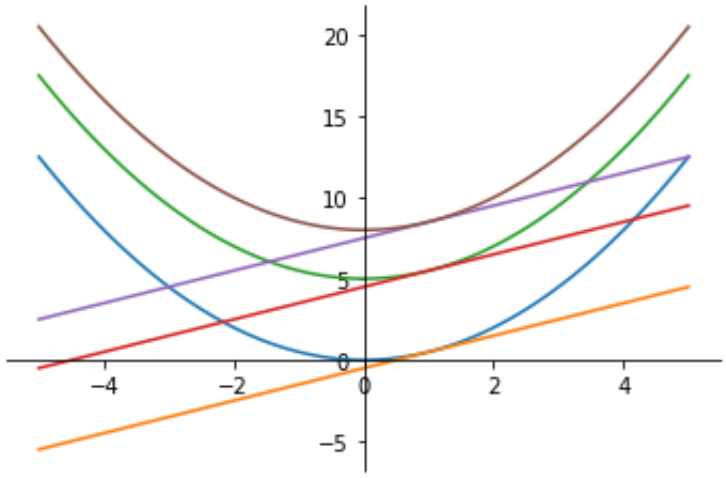
In [30]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = x**2/2
y1 = x**2/2+5
y2 = x**2/2+8

ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')

for i in range(10):
    plt.plot(x,y+i)
#plt.plot(x,y1)
#plt.plot(x,y2)
```



Looking at the picture of $F(x) = x^2/2 + C$ for three different values of C above, it is not so difficult to believe that the derivatives of these three functions are all the same. After all, the derivative is the slope of the tangent line. These slopes of the tangent lines of these functions at a given point are all the same. We plot these tangent lines below at $x = 0$.

In [31]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = x**2/2
z = x-1/2
y1 = x**2/2+5
z1 = x+4.5
y2 = x**2/2+8
z2 = x+7.5



ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y)
plt.plot(x,z)
plt.plot(x,y1)
plt.plot(x,z1)
plt.plot(x,z2)
plt.plot(x,y2)
```

Out[31]:
```
[<matplotlib.lines.Line2D at 0x11c16b1d0>]
```



As we can see, the slopes of all the tangent lines at $x = 1$ are the same as the lines are parallel.

Let us compute a few anti-derivatives. Let $f(x) = x^2$. Then the anti-derivative $F(x) = x^3/3 + C$, where C is an arbitrary constant. How did we know that? If n is an integer, the derivative of x^n with respect to x is

$$nx^{n-1}.$$

In other words, to take a derivative of x raised to a power, we write down this power and multiply it by x raised to that power less one. This gives us a hint for how to compute the anti-derivative in the case when $f(x) = x^n$. Since we are inverting the process, instead of reducing the power by 1, we increase it by 1. And instead of multiplying the answer by the original power, we divide it by the power plus 1. In this way, the anti-derivative of x^2 is

$$F(x) = \frac{1}{2+1}x^{2+1} + C = \frac{1}{3}x^3,$$

where C is an arbitrary constant.

Similarly, if $f(x) = 5x^7$, the anti-derivative is

$$F(x) = \frac{5}{7+1}x^{7+1} + C = \frac{5}{8}x^8 + C,$$

where C is an arbitrary constant.

The only time when this approach does not work is when x is raised to the power -1, i.e $f(x) = x^{-1} = \frac{1}{x}$. We cannot use the approach above because dividing by $-1 + 1$ would mean dividing by 0 and this is not allowed. Indeed, the anti-derivative of $f(x) = x^{-1}$ is rather special. It is given by

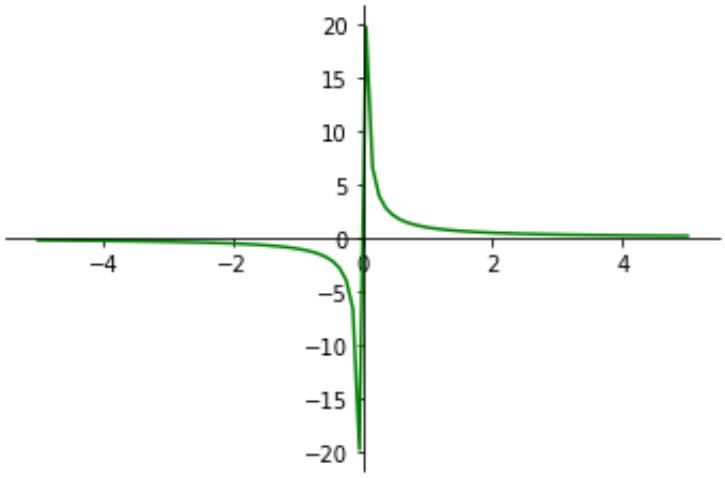
$$F(x) = \log(x) + C,$$

where $log(x)$ is the natural logarithm of x.

As a reminder, here is the graph of $f(x) = \frac{1}{x}$, followed by the graph of $log(x)$.

In [32]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = 1/x



ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y,'green')
```

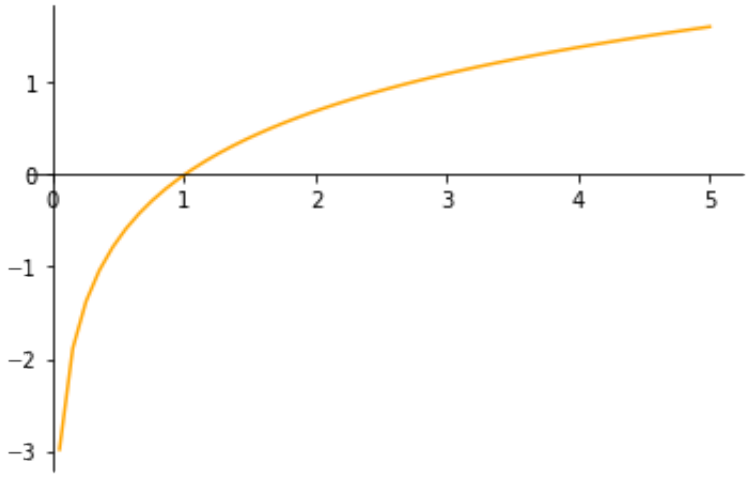
Out[32]:
```
[<matplotlib.lines.Line2D at 0x11cb67c50>]
```



In [33]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = np.log(x)



ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y,'orange')
```

```
/Users/iosevich/opt/anaconda3/lib/python3.7/site-packages/ipykernel_
launcher.py:5: RuntimeWarning: invalid value encountered in log
  """
```

Out[33]:
```
[<matplotlib.lines.Line2D at 0x11cbb3f90>]
```

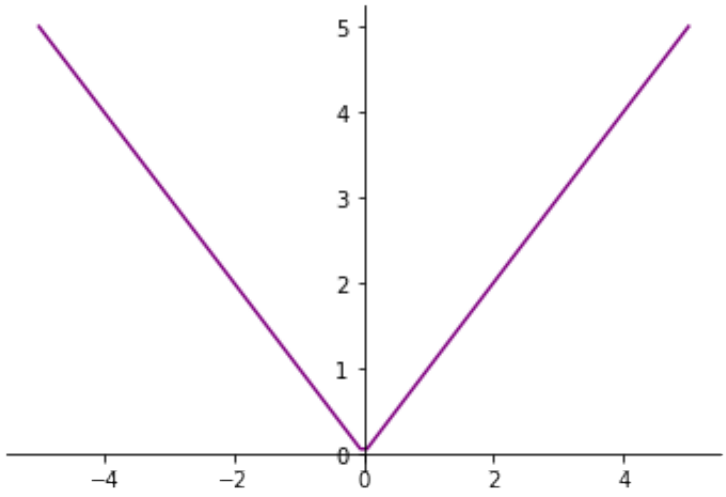


Please note that we did not need to graph any of these functions in order to make the calculations that we made. So why did I graph them anyways? Because in mathematics, the moment you start forgetting what graphs of the functions you work with look like, bad things start happening. So even when you are not working explicitly with the graphs, you should keep in your mind what the graphs look like because sooner or later this information is going to come in handy.

We learned in Calc 1 that the function $F(x) = |x|$ is not differentiable at $x = 0$.

In [34]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = abs(x)



ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y, 'purple')
```

Out[34]:
```
[<matplotlib.lines.Line2D at 0x11c351d10>]
```



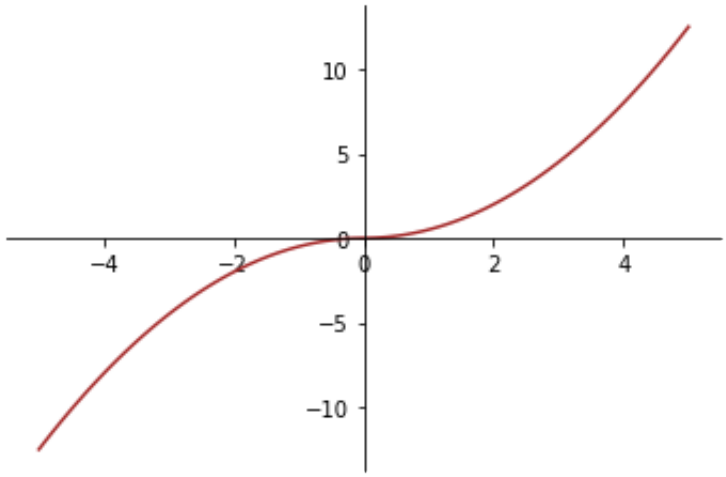
To remind ourselves how this works, as we approach 0 from the right, the slope of the tangent line is 1. But as we approach 0 from the left, the slope of the tangent line is -1. At $x = 0$, the derivative cannot decide whether it wants to be $+1$ or -1, so the derivative of $f(x)$ at 0 does not exist.

But the anti-derivative of $f(x) = |x|$ exists. What is it? It is the function $F(x)$ which equals $\frac{x^2}{2} + C$ for $x > 0$ and $-\frac{x^2}{2} + C$ if $x < 0$. We also set $F(0) = 0$.

The derivative of this function is equal to 1 if $x > 0$ and -1 if $x < 0$. The value of the derivative at $x = 0$ is 0 and this is precisely how $|x|$ is defined. Here is the graph of $F(x)$ with $C = 0$:

In [35]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-5, 5, 100)
y = np.sign(x)*x**2/2


ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y, 'brown')
```

Out[35]:
```
[<matplotlib.lines.Line2D at 0x11c7a6fd0>]
```



Notice that the graph of $F(x)$ looks a lot like the graph of x^3, but there are some sneaky differences that we are not going to worry about in this class.

What I would like you to notice is the graph of $|x|$ looks rougher than the graph of its anti-derivative above. By rougher I mean that there is a "corner" at the origin. But the graph of the anti-derivative does not have any corners. The idea is here is that taking anti-derivatives makes a function smoother. This is because an anti-derivative is a kind of an average of a function. We are going to understand this idea a bit better soon.

Let's do a few examples that are slightly more complicated. Let $f(x) = (x + \frac{1}{2}) \cdot (x^2 + x + 1)^7$. Let's find the most general form of a function $F(x)$ such that $F'(x) = f(x)$. The idea is to recall the chain rule. First, try

$$Try(x) = (x^2 + x + 1)^8.$$

Then, using chain rule,

$$\begin{aligned} Try'(x) &= 8(x^2 + x + 1)^7 \cdot (2x + 1) \\ &= 16(x^2 + x + 1)^7 \cdot (x + \frac{1}{2}) \end{aligned}$$

since

$$2(x + \frac{1}{2}) = 2x + 1.$$

It follows that $Try'(x)$ is equal to $16f(x)$. But we do not want $16f(x)$. We want $f(x)$!! What do we do? We just divide $Try(x)$ by 16. This tells us that our anti-derivative is equal to

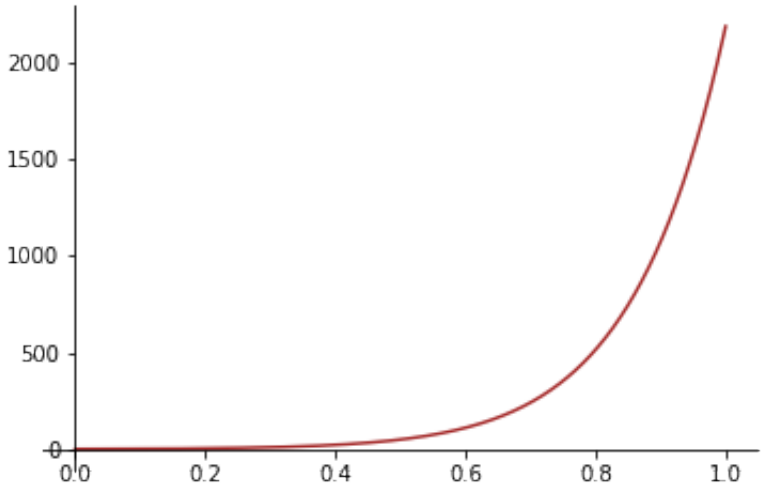
$$\frac{1}{16}(x^2 + x + 1)^8 + C,$$

where C is an arbitrary constant.

As usual, we graph $f(x)$ to make sure that we always keep on eye on what functions look like. Let us first see what the graph of $f(x)$ looks like when $0 \leq x \leq 1$.

In [36]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 1, 10000)
y = (x**2+x+1)**7


ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y, 'brown')
```

Out[36]:
```
[<matplotlib.lines.Line2D at 0x11c691fd0>]
```



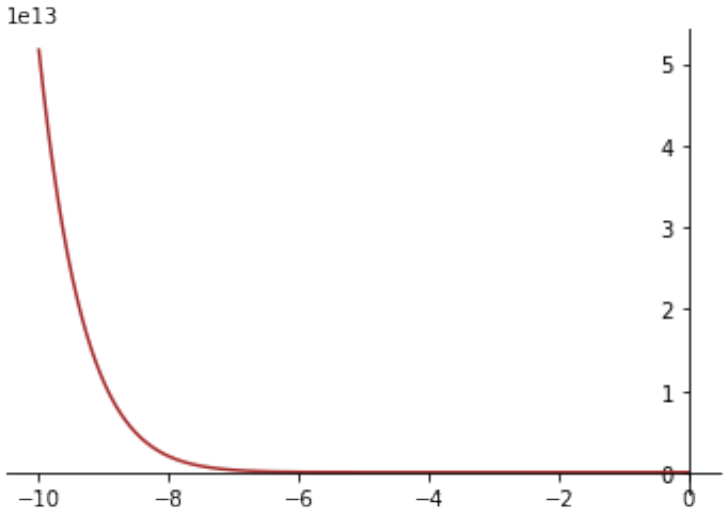
Notive that the graph is flat when x is close to 0, but then it takes off.

I am now going to add some extra python code that you may want to play around with if you enjoy that type of stuff. The code is thoroughly documented, so you should be able to follow it and modify it if you know a bit of python.

Let us now see what the graph of $f(x)$ looks like when x is negative.

In [37]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-10, 0, 10000)
y = (x**2+x+1)**7


ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y, 'brown')
```

Out[37]:
```
[<matplotlib.lines.Line2D at 0x11c9abb90>]
```

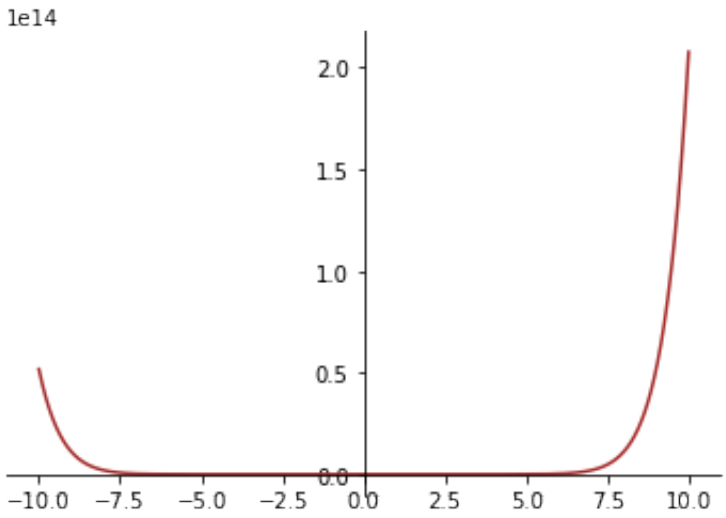


Note that the graph still takes off, but it takes longer because for a while, x draggs down x^2+1.

In the first graph of $f(x)$ above, we considered $0 \leq x \leq 1$. In the second graph, we considered $-10 \leq x \leq 0$. Now, let's consider $-10 \leq x \leq 10$.

In [38]:
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-10, 10, 10000)
y = (x**2+x+1)**7


ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
plt.plot(x,y, 'brown')
```

Out[38]:
```
[<matplotlib.lines.Line2D at 0x11c969c10>]
```



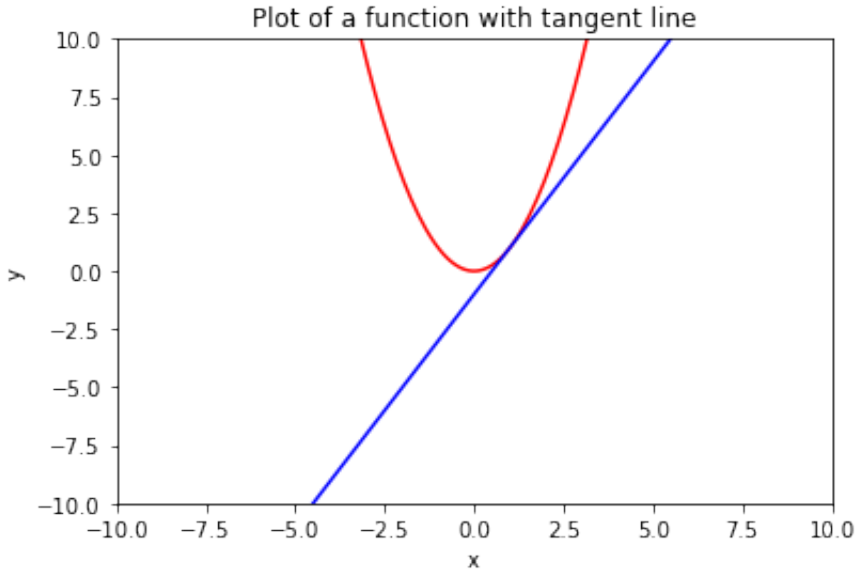
Do you see the problem? The graphing program is trying to "scale" the function in the same way on both sides of the origin. As a result, we are getting a rather misleading picture. When the function behaves differently in different ranges of x, one needs to zoom in on those ranges separately in order to get a better view. This is yet another illustration that using technology without understanding it is frought with danger.

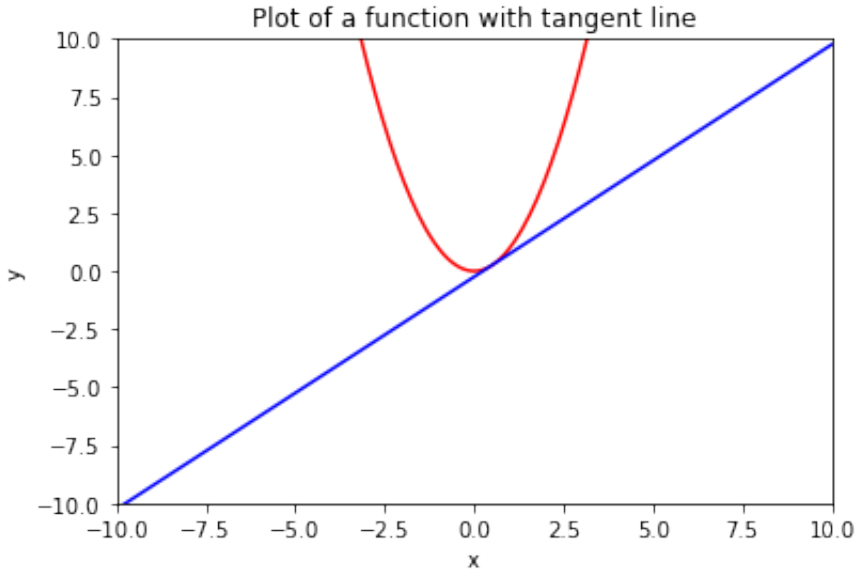
This ends the lecture, but I would like to throw in a bit of fun stuff below. It is python code for graphing a tangent line to the curve at a given point. If you would like to learn more about this in order to be able to modify the code, etc, please feel free to ask me questions.

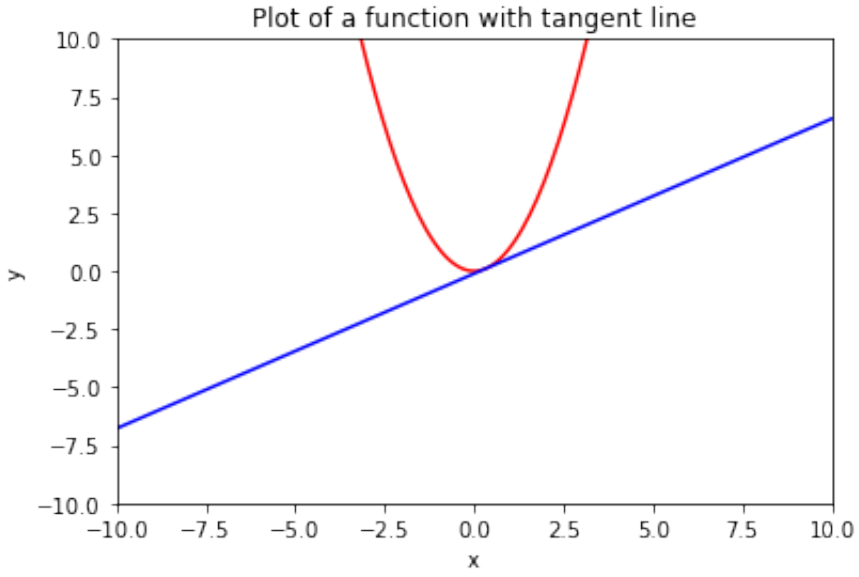
In [25]:
```
# defining a derivative function at x approximately. It is more than g
ood enough for graphing purposes.

def deriv(f,x,h):
    return (f(x+h) - f(x))/h        #definition of derivative. Note th
at as h gets smaller, we get close and close to
                                    # to the actual derivative.
```

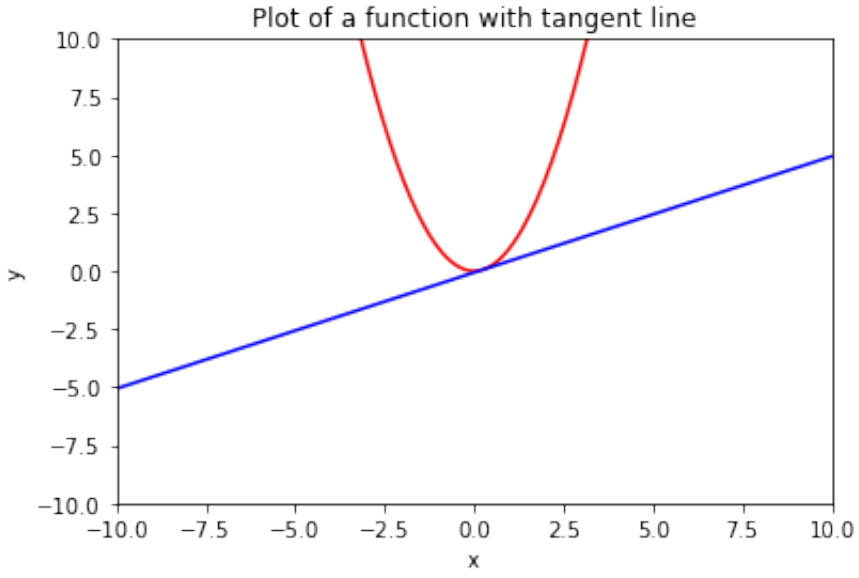
In [26]:
```
# defining a function that yields a tangent line to the graph of a giv
en function at a given point in a given range.

def tangent_line(f,x_0,h,a,b):
    x = np.linspace(a,b,200)
    y = f(x)
    y_0 = f(x_0)
    y_tan = deriv(f,x_0,h) * (x - x_0) + y_0

  #plotting
    plt.plot(x,y,'red')
    plt.plot(x,y_tan,'blue')
    plt.axis([a,b,a,b])
    plt.xlabel('x')
    plt.ylabel('y')
    plt.title('Plot of a function with tangent line')
    plt.show()
```

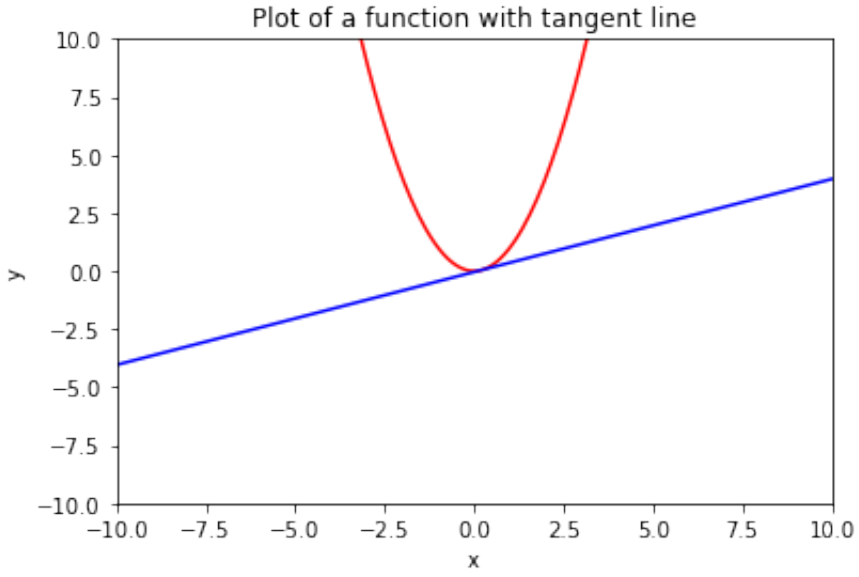
In [27]:
```
# defining a function to be used in conjunction with the above.

def f1(x):
    return x**2

# an illustration of how the tangent line changes as we graph at x=1/i
, i=1,2,...,10

for i in range(1,11):
    tangent_line(f1,1/i,.0000001,-10,10)
```

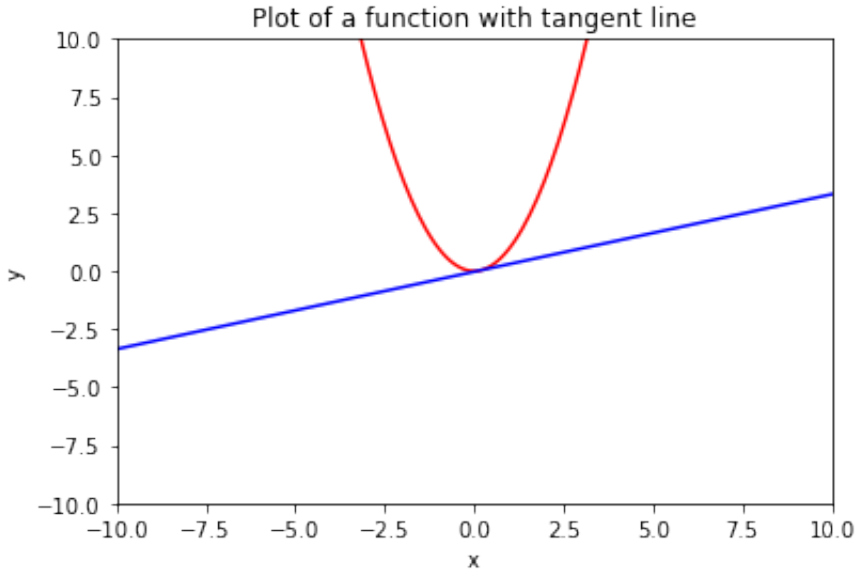

Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0
x

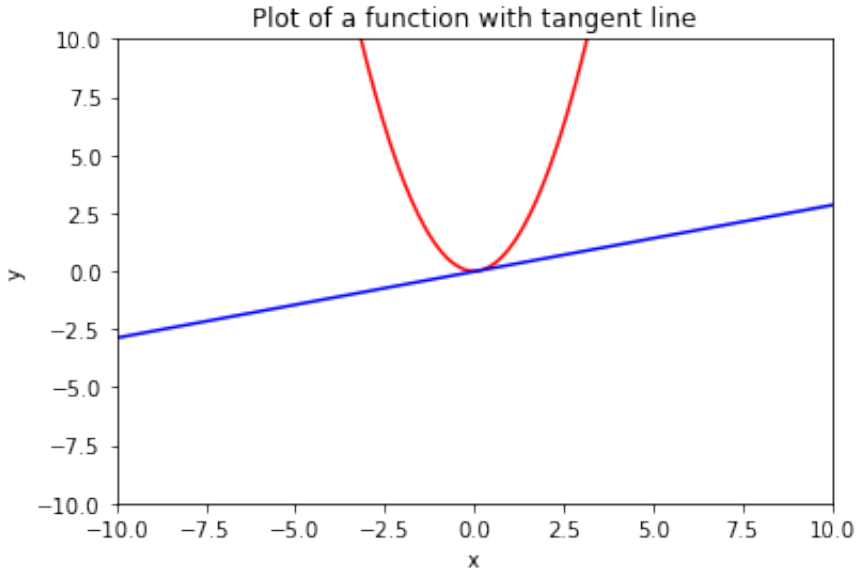

Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0
x

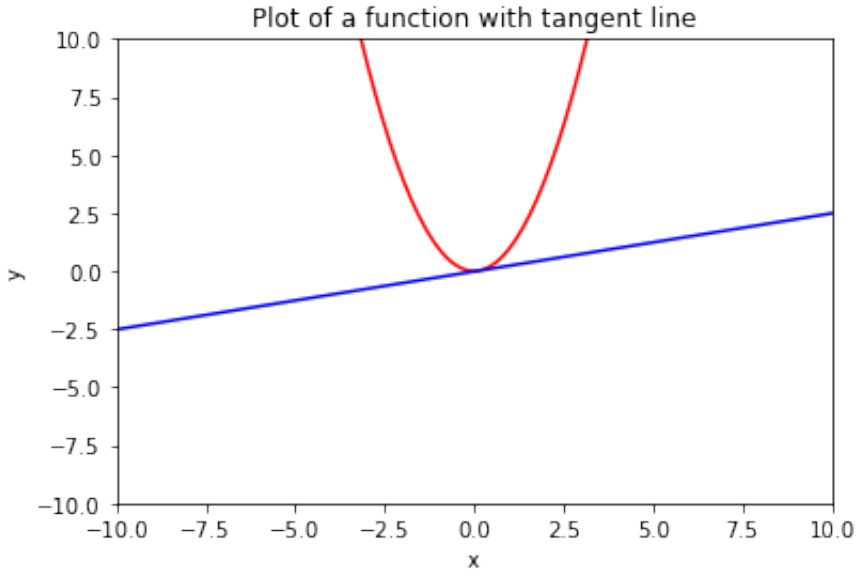

Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0
x

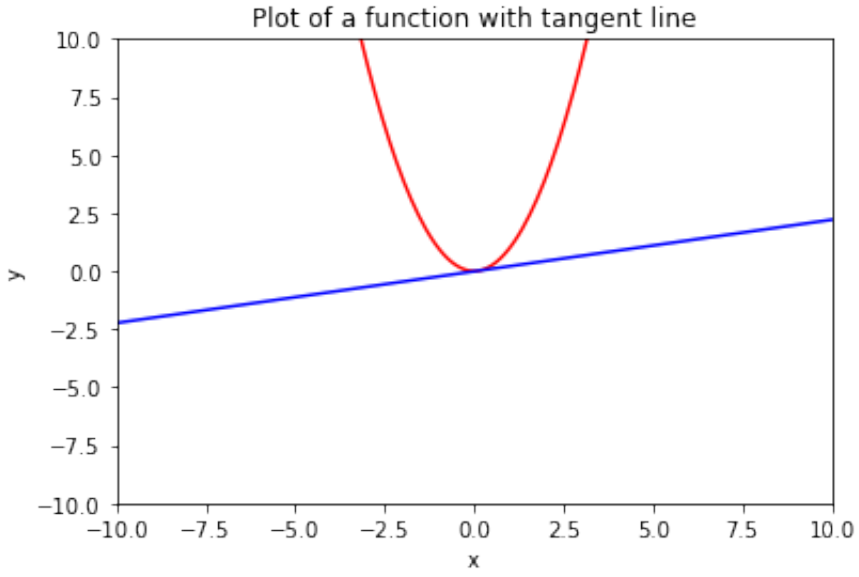

Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0
x

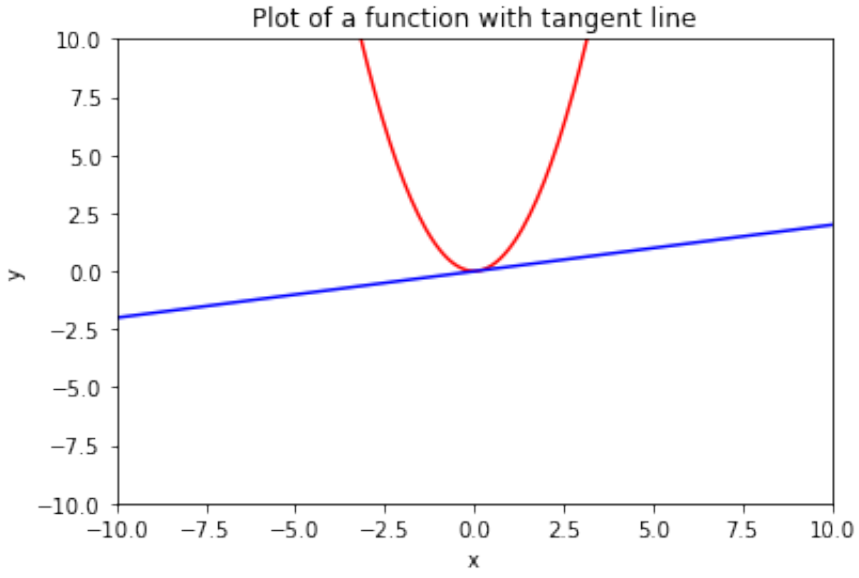

Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0
x


Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0
x


Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
x


Plot of a function with tangent line
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
x





In [41]:

Out[41]: 0

In [ ]: