Taylor Polynomial Approximations of a Function

Introduction

OBJECTIVE: Observe an animated demonstration of the convergence of Taylor polynomials to a function that has derivatives of all orders over some interval of its domain.

Part II of this module contains a specially designed function that can be used for demonstrations and explorations. The **taylorpolydemo[ ]** command demonstrates the convergence of Taylor polynomials to a function over some interval of its domain, provided the function has derivatives of all orders. The command generates a series of plots showing the function and the Taylor polynomial, as the degree of the polynomial increases. The series of plots that is generated can be animated to demonstrate the convergence of the Taylor polynomials. Several functions are included in the module and others can easily be added. Part I leads you through the construction of the **taylorpolydemo[ ]** command and explains how it works. If you are only interested in using **taylorpolydemo[ ]** for demonstrations and explorations, you can skip Part I and the related You Try It section, and go directly to Part II.

Technology Guidelines

NOTE: If you have just finished a module, restart *Mathematica* or close the *Kernel* before executing a new module.
TO OPEN CELLS, put your cursor on the right cell bracket and double click.
INITIALIZATION CELLS
When asked if you want to ". . . automatically evaluate all the initialization cells in the notebook . . . ," respond by pressing the "Yes" button.
TO STOP AN EXECUTION
Select the *Kernel* pull-down menu and click on *Abort Evaluation.*
ORDER OF EXECUTION
Execute cells in the order given. Do not skip any Input cells within a given notebook.
SAVING NOTEBOOKS
You can save anytime to any directory you choose, and it is wise to save often.

However, before you do your final save, delete all your output by selecting the *Delete All Output* selection under the *Kernel* pull-down menu.

EXPERIENCING MAJOR PROBLEMS

Save if appropriate, then shut down *Mathematica* and start it up again.

Part I: Computing and Visualizing Taylor Polynomials

We can use *Mathematica* to find Taylor polynomial approximations for functions in the vicinity of a point whose *x*-coordinate equals *a*. The third-degree polynomial that approximates **Exp[x]** near x = 0 is:

In[49]:=

```
Clear[x]

Series[Exp[x], {x, 0, 3}]
```

Out[50]=

$$1 + x + \frac{x^2}{2} + \frac{x^3}{6} + 0[x]^4$$

The last term is called the error term. It tells us that the cubic polynomial gives a fourth-order approximation of **Exp[x]** near x = 0.

To evaluate a Taylor polynomial for specific values of *x*, we must obtain a form of the polynomial that does not include the error term. This can be done as follows.

In[51]:=

```
Normal[Series[Exp[x], {x, 0, 5}]]
```

Out[51]=

$$1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120}$$

Next, we build a *Mathematica* command that will give an n^{th} degree Taylor polynomial for approximating **f** near x = a.

In[52]:=

```
taylor[f_, a_, n_] := Normal[Series[f, {x, a, n
```

The next command shows how **taylor[ ]** works.

In[53]:=

```
taylor[Exp[x], 0, 7]
```

Out[53]=

$$1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040}$$

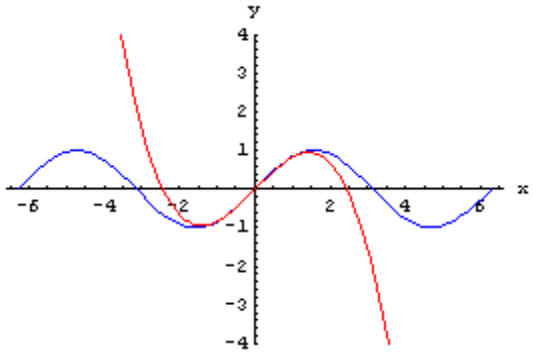
Now we use our new **taylor[ ]** command to demonstrate how Taylor polynomials converge on the **Sin[x]** as *n*, the degree of the polynomial, increases.

```
"> About Mathematica
```

In[54]:=

```
f = Sin[x];

Print[
 "The third degree Taylor polynomial to appı
   Sin(x) is ", g = taylor[f, 0, 3]]

graph1 = Plot[f, {x, -2 π, 2 π}, PlotStyle → {R
  AxesLabel → {"x", "y"}, DisplayFunction -> I

graph2 = Plot[g, {x, -2 π, 2 π}, PlotStyle → RG
  AxesLabel → {"x", "y"}, DisplayFunction -> I

Show[graph1, graph2, PlotRange → {-4, 4},
 DisplayFunction -> $DisplayFunction];

Print[
 "The function is plotted in blue; its Taylı
   is in red."]
```

```
The third degree Taylor polynomial
   to approximate Sin(x) is x - x^3/6
```




```
The function is plotted in blue;
  its Taylor polynomial is in red.
```

We can put this all together and form a new command, which we will call **compare[ ]**. When a *Mathematica* command consists of a list of instructions, we use the **Block[ ]** command to form the function.

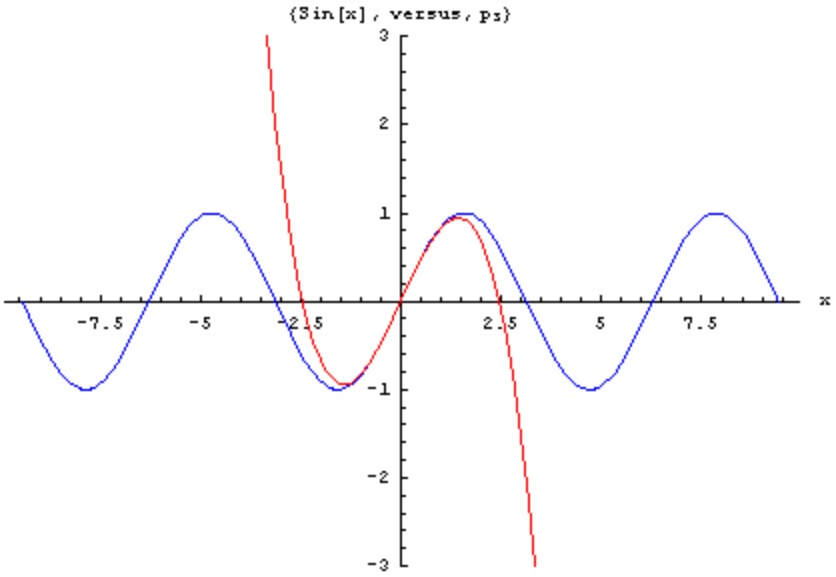
In[60]:=

```
compare[f_, a_, n_, xlimits_, yyylimits_] :=
 Block[{}, g = taylor[f, a, n]; size = 6;
        graph1 = Plot[f, {x, xlimits[[1]], xli
    PlotStyle → {RGBColor[0, 0, 1]}, AxesLabel
    ImageSize -> {72 * size, 72 * size / 3 * 2}, As
    DisplayFunction → Identity];
        graph2 = Plot[g, {x, xlimits[[1]], xli
    PlotStyle -> {RGBColor[1, 0, 0]}, AxesLabe
    ImageSize -> {72 * size, 72 * size / 3 * 2}, As
    DisplayFunction → Identity];
        Show[{graph1, graph2},
   DisplayFunction → $DisplayFunction,
   PlotRange -> {yyylimits[[1]], yyylimits[[2
   PlotLabel -> {f , "versus",  pn}];     ];
```

Let's try it on sin(x) near x = 0.

In[61]:=

```
compare[Sin[x], 0, 3, {-3 * Pi, 3 * Pi}, {-3, 3}]
```

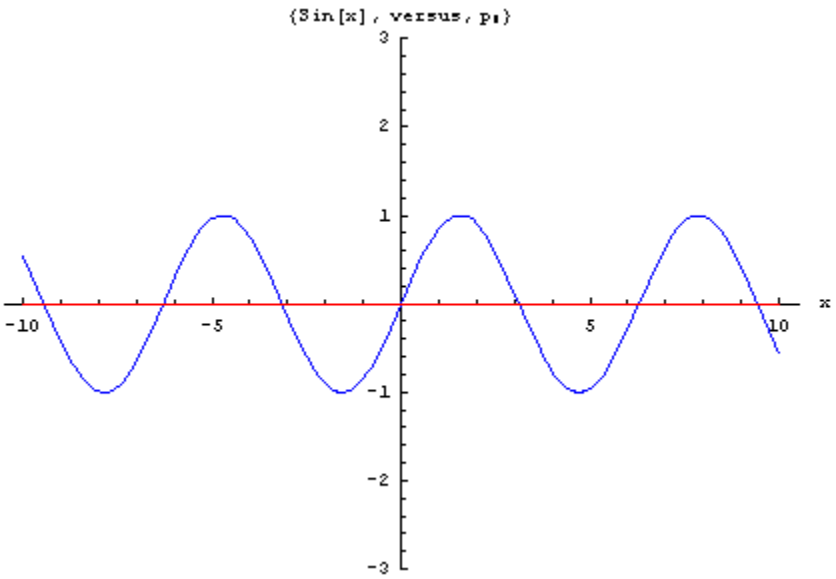


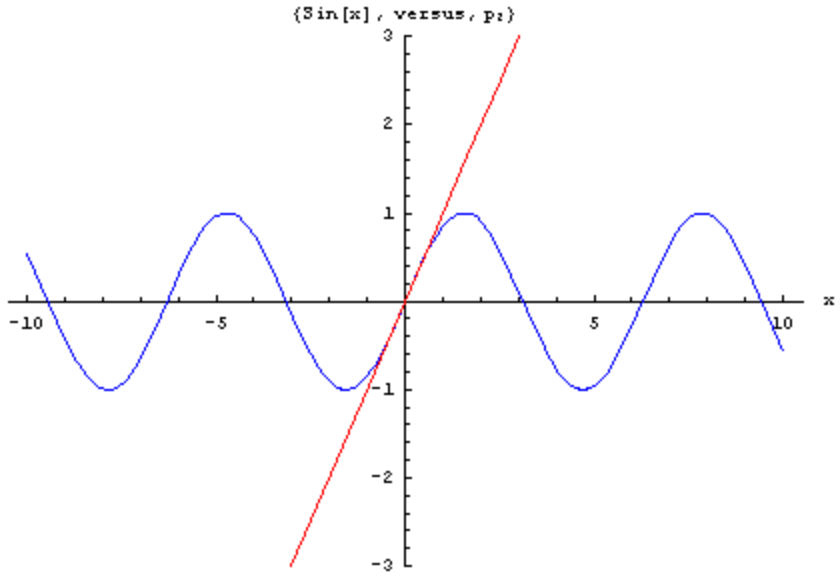
The **Do[ ]** command will generate a sequence of graphs showing the convergence of the Taylor polynomials on *f* near x = a. The value of *n* will go from 0 to 20 in increments of 2. This means that the sine function will be approximated by polynomials of degree 0 through 20. Notice how each successive approximation does a better job of fitting the function as you move away from x=0.

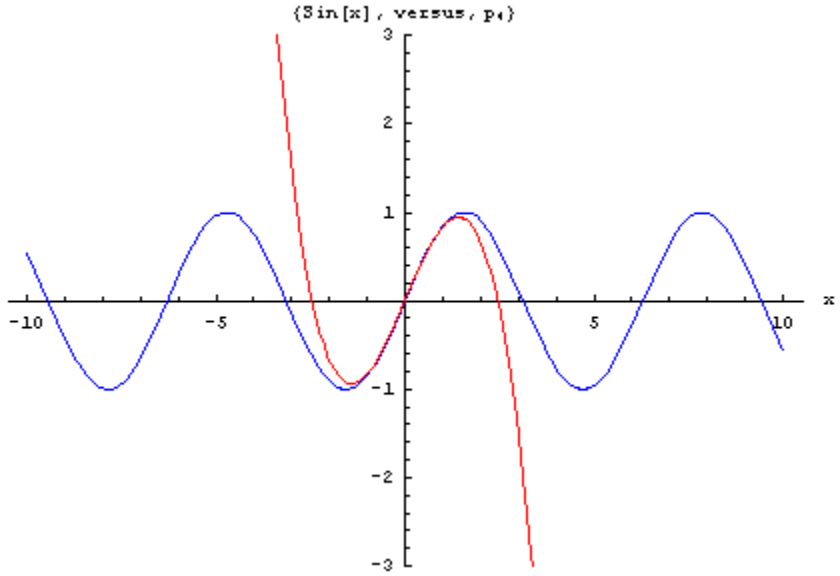
You can animate the graphs by double clicking on any one of the graphs in the sequence.

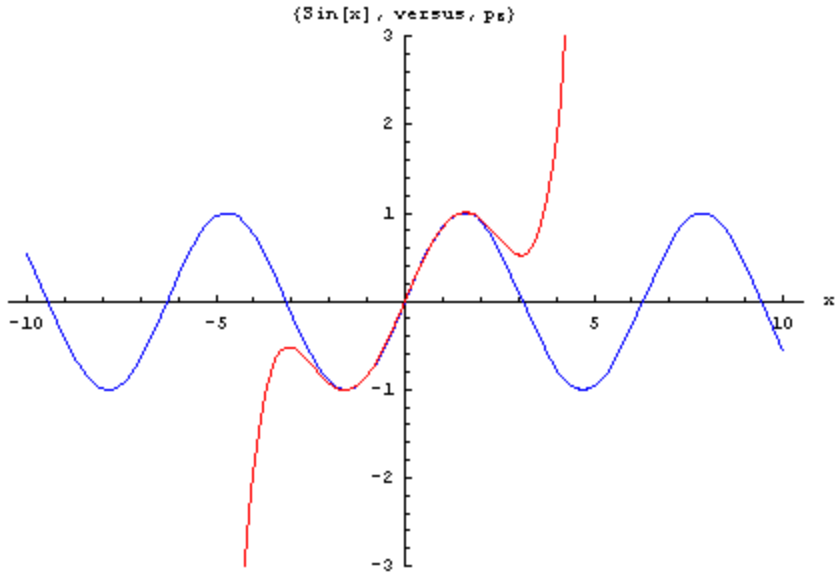
In[62]:=

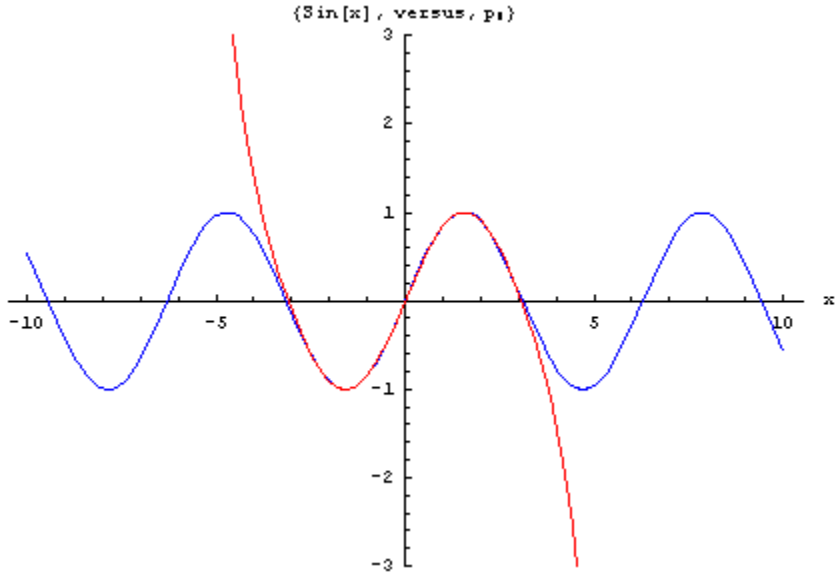
```
Do[compare[Sin[x], 0, n, {-10, 10}, {-3, 3}],
```

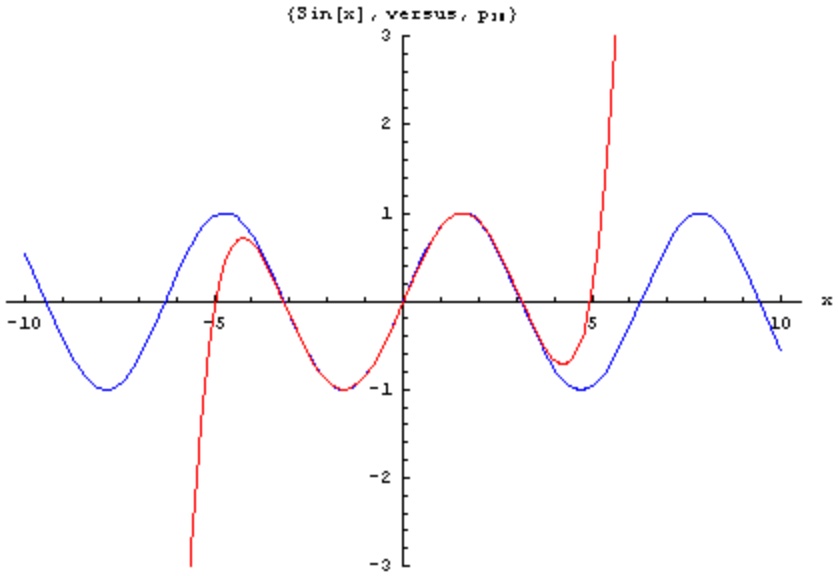


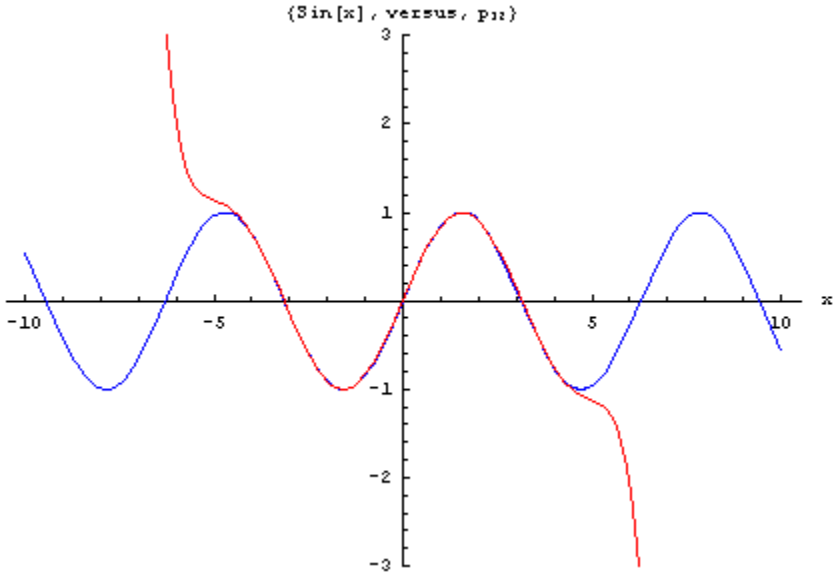

{Sin[x], versus, p2}
3
2
1
-1
-2
-3
-10
-5
5
10
x

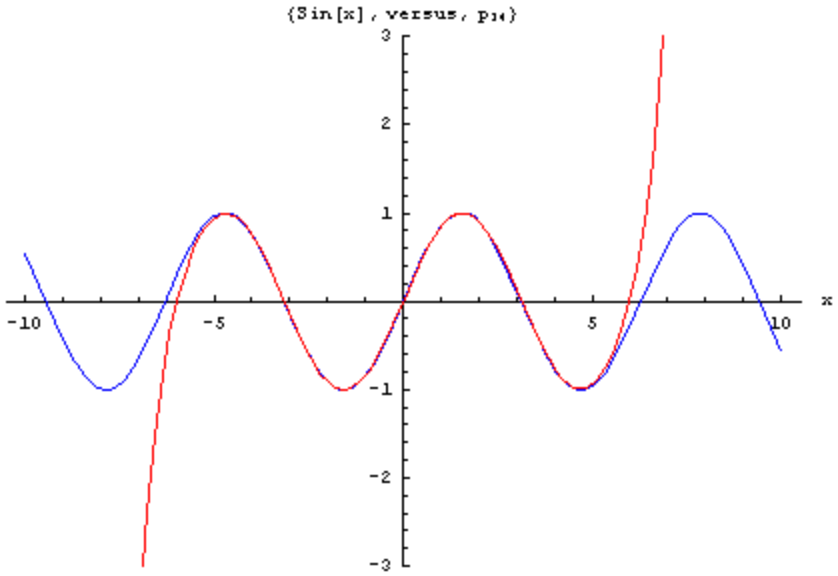

{Sin[x], versus, p4}
3
2
1
-1
-2
-3
-10
-5
5
10
x

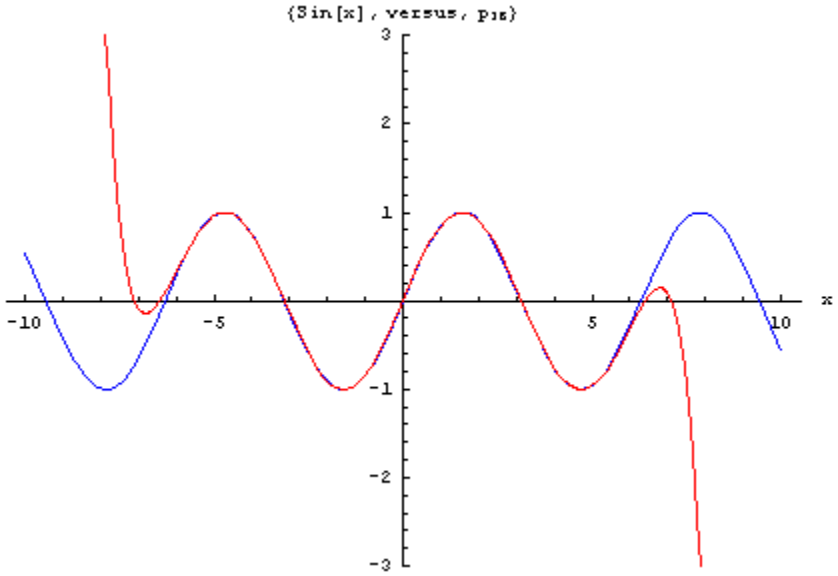

{Sin[x], versus, p6}
3
2
1
-1
-2
-3
-10
-5
5
10
x

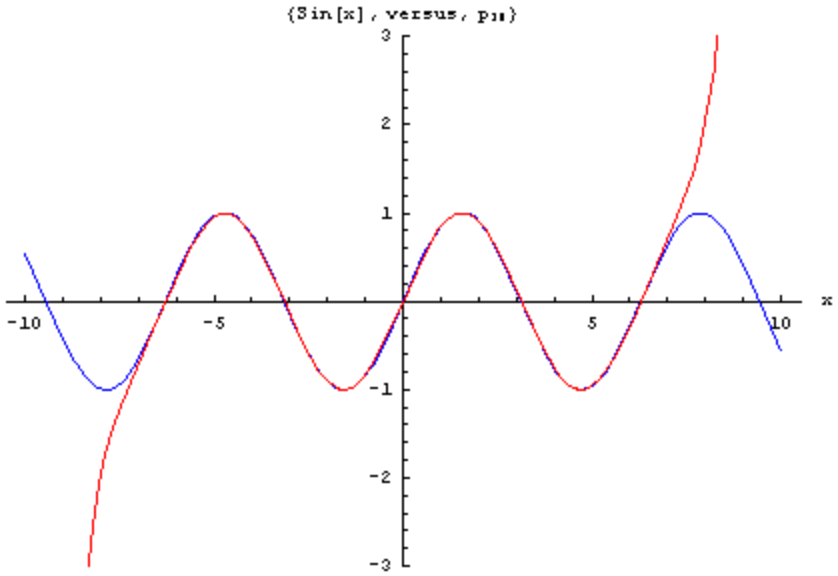

{Sin[x], versus, p7}
3
2
1
-1
-2
-3
-10
-5
5
10
x

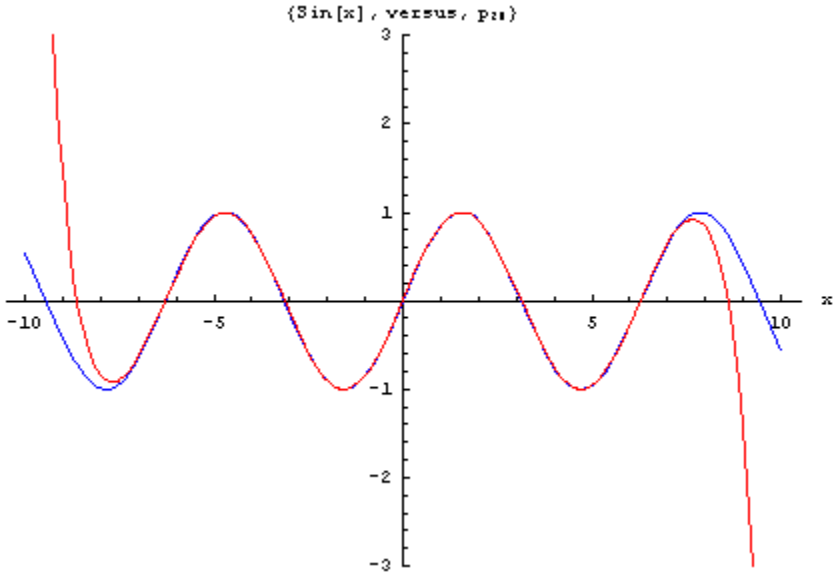

{Sin[x], versus, p1}
3
2
1
-1
-2
-3
-10
-5
5
10
x


{Sin[x], versus, p2}
3
2
1
-1
-2
-3
-10
-5
5
10
x


{Sin[x], versus, p14}
3
2
1
-1
-2
-3
-10
-5
5
10
x


{Sin[x], versus, p15}
3
2
1
-1
-2
-3
-10
-5
5
10
x





You Try It: Part I

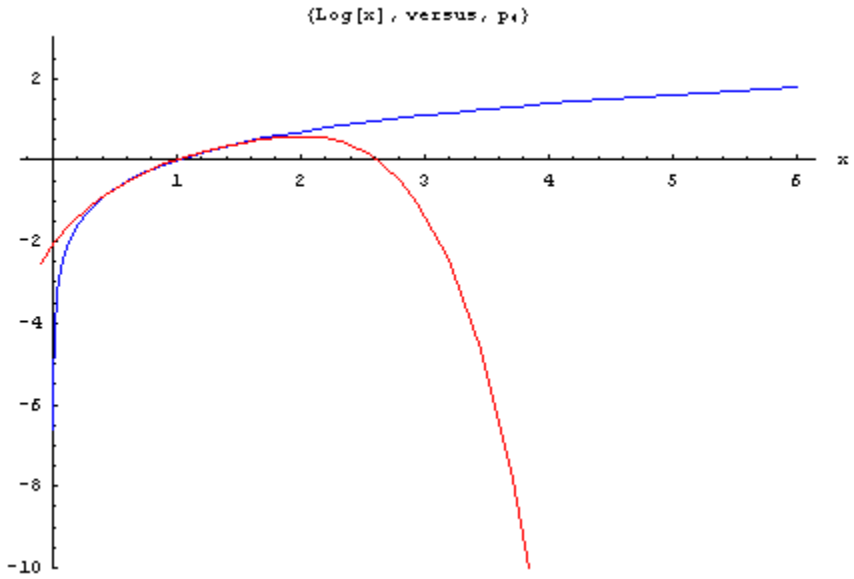
Try some of your own functions. Some suggestions are: **Sin[x]** at a=0, **Cos[x]** at a=0, **Exp[x]** at a=0, **1/(1+x)** at a=0, **Log[x]** (the natural log) at a=1,and **Sqrt[x]** at a=4. Change any of the terms in red that you wish. The first cell creates only one Taylor polynomial and graph. The red terms in the next cell will carry out the Taylor approximation for the natural log function around a=1.

In[63]:=

```
f = Log[x];
a = 1;
degree = 4;
xlimits = {-.1, 6};
ylimits = {-10, 3};
Print["The Taylor polynomial to approximate
 " around ", a, " is ", taylor[f, a, degree]]

compare[f, a, degree, xlimits, ylimits];

Print[
 "The function is plotted in blue; its Tayl
   is in red."]
```

The Taylor polynomial to approximate
 Log[x] around 1 is

$$-1-\frac{1}{2}(-1+x)^2+\frac{1}{3}(-1+x)^3-\frac{1}{4}(-1+x)^4+x$$

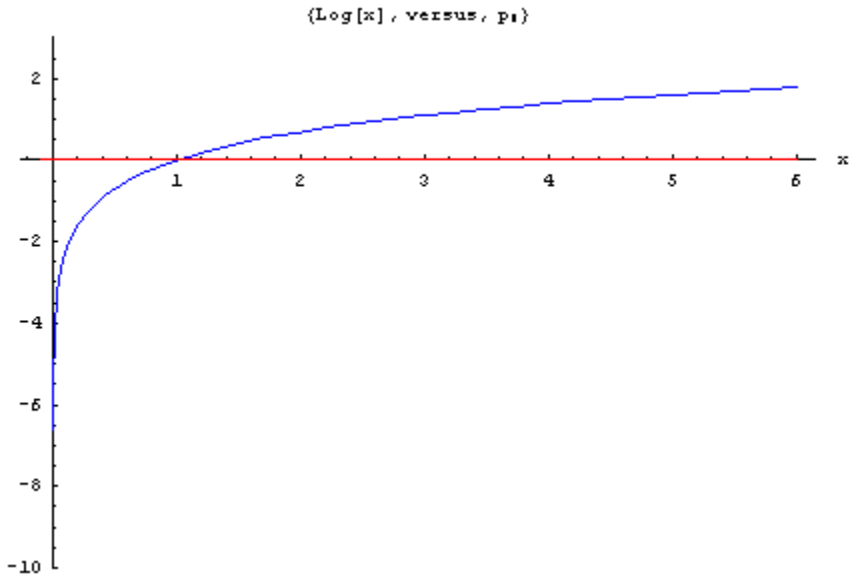


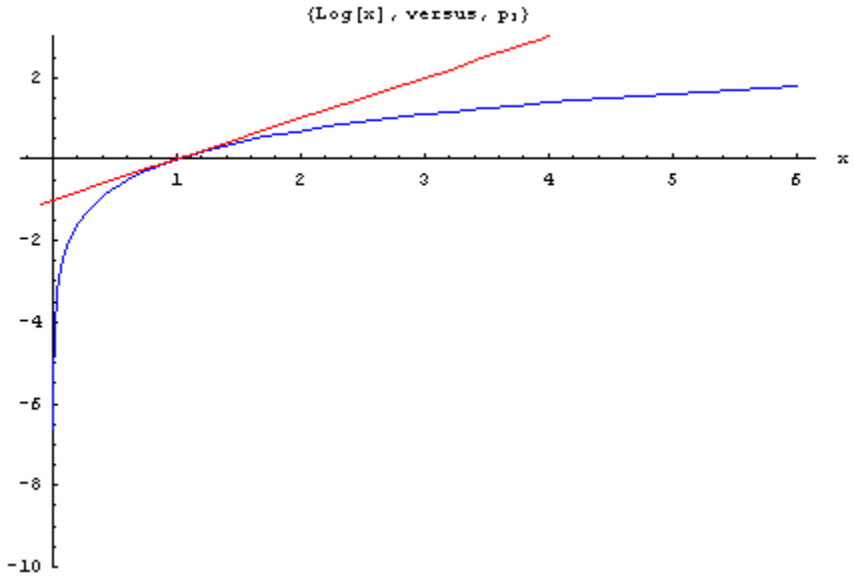
The function is plotted in blue;
  its Taylor polynomial is in red.

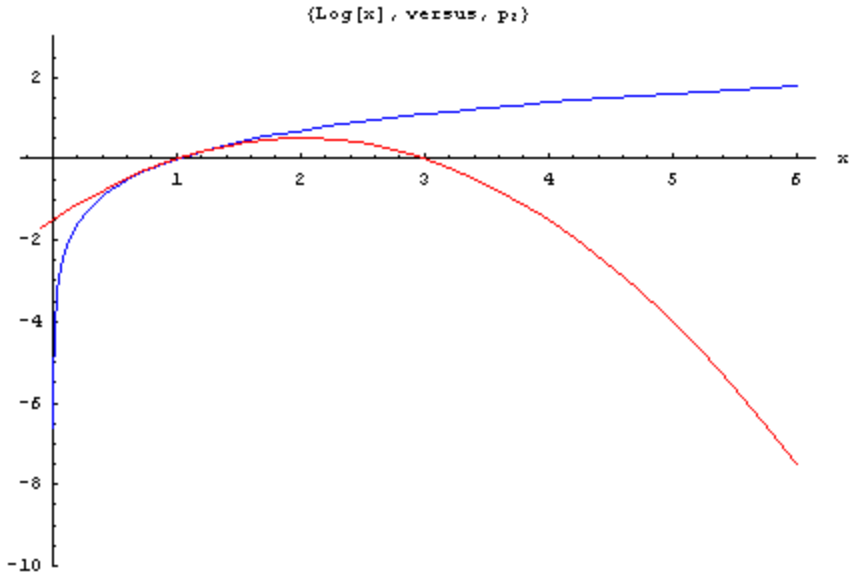
To create a series of plots with the function you have chosen, execute the following command.

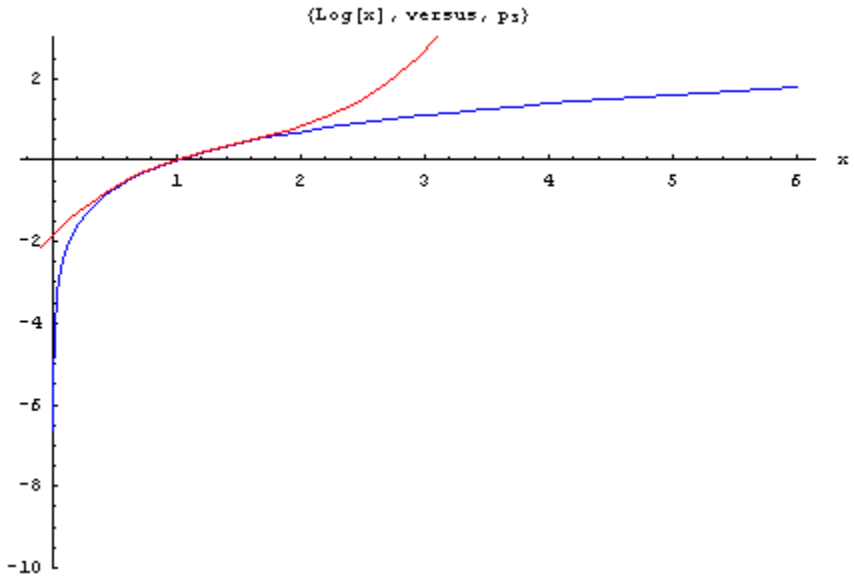
In[66]:=

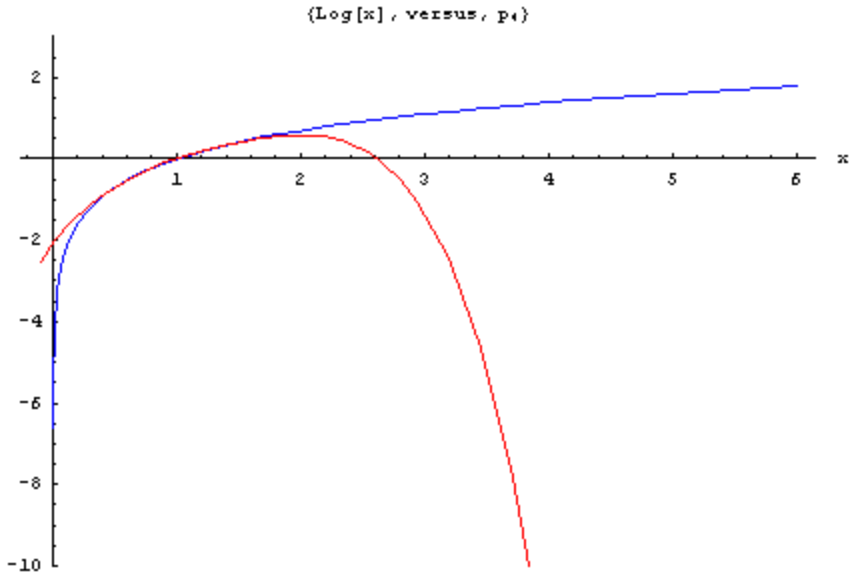
```
Do[compare[f, a, n, xlimits, ylimits], {n, 0, 
```

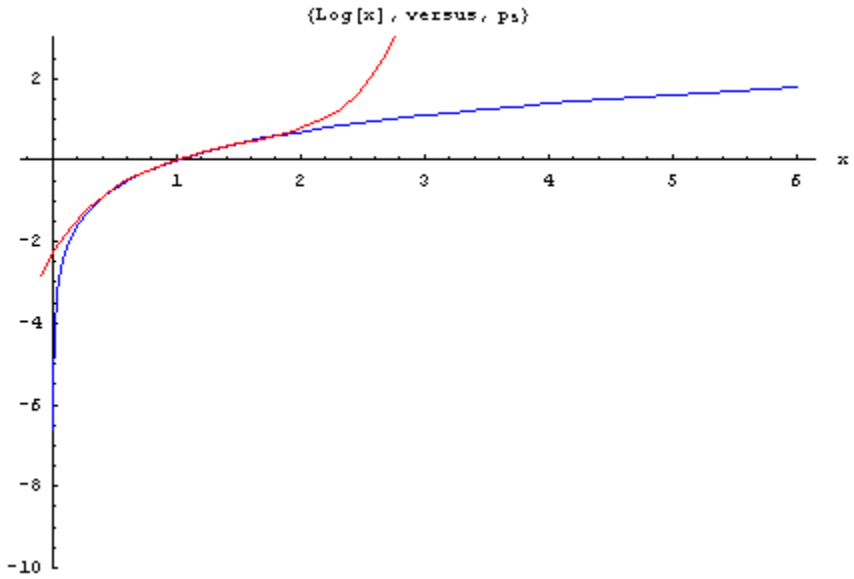


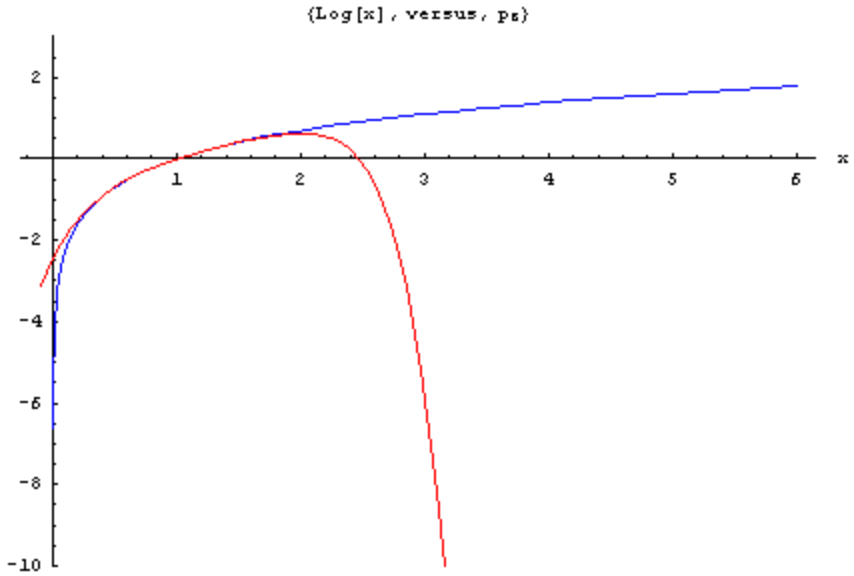


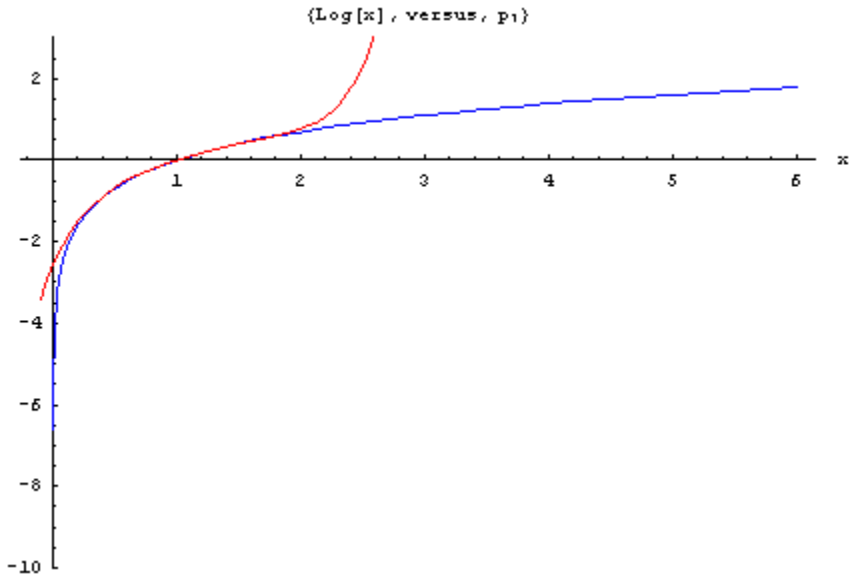

{Log[x], versus, p1}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

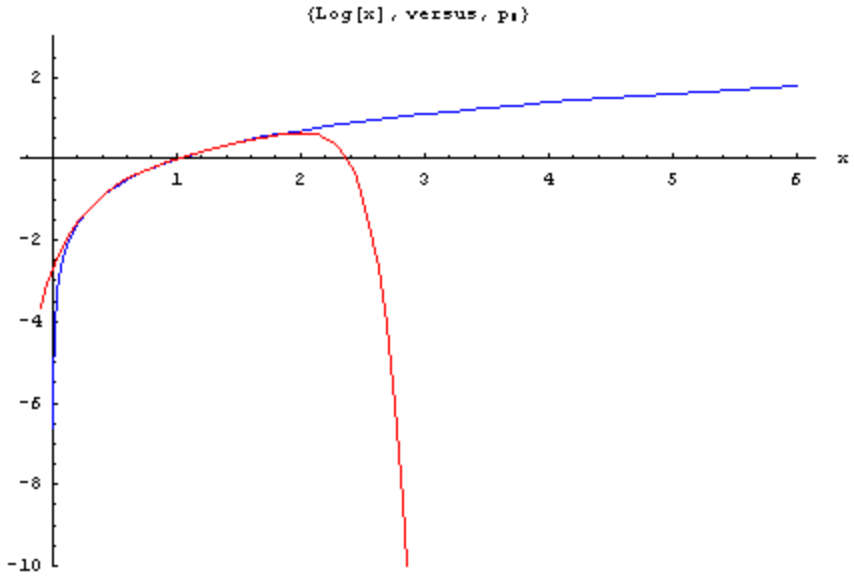

{Log[x], versus, p2}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

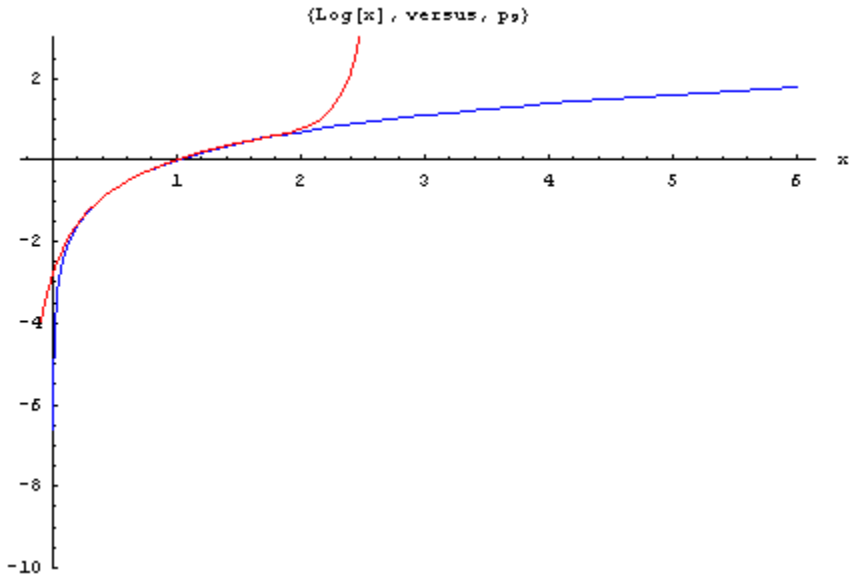

{Log[x], versus, p4}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

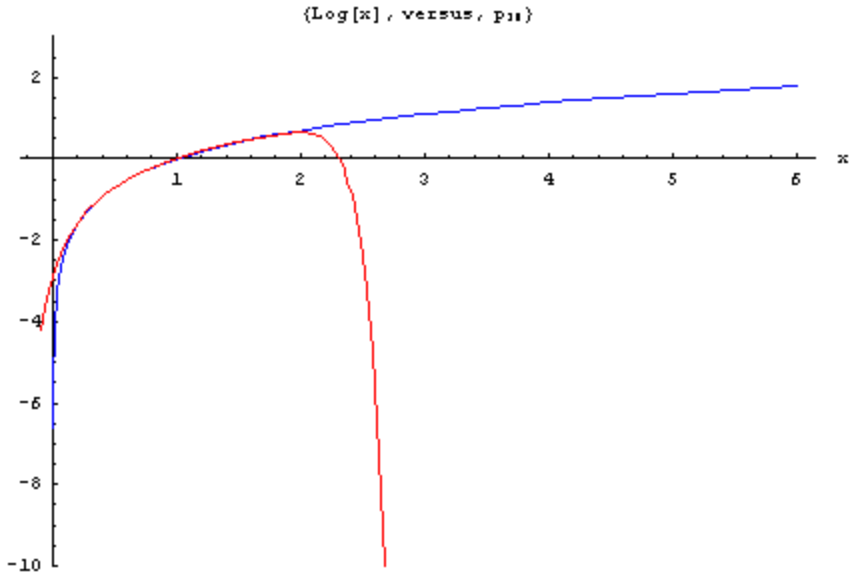

{Log[x], versus, p5}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

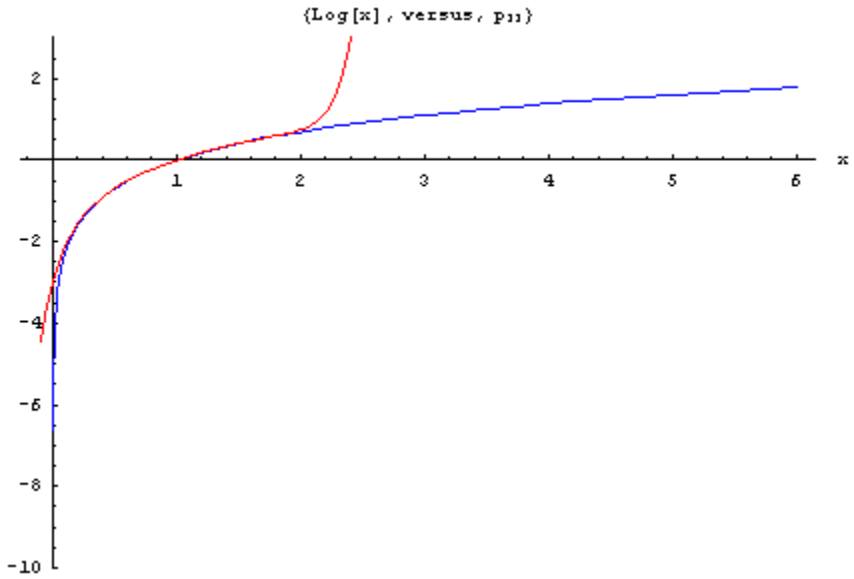

{Log[x], versus, p5}
2
-2
-4
-6
-8
-10
1
2
3
4
5
6
x

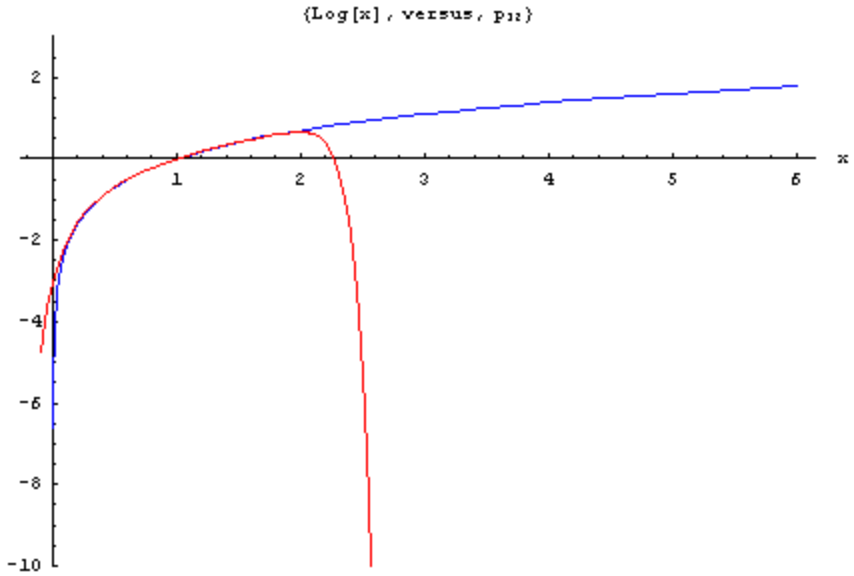

{Log[x], versus, p1}
2
-2
-4
-6
-8
-10
1
2
3
4
5
6
x

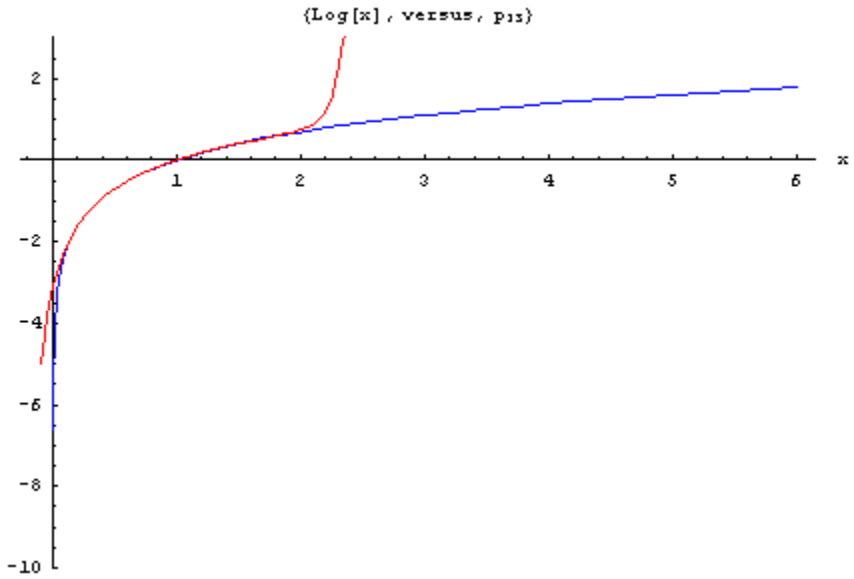

{Log[x], versus, p1}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

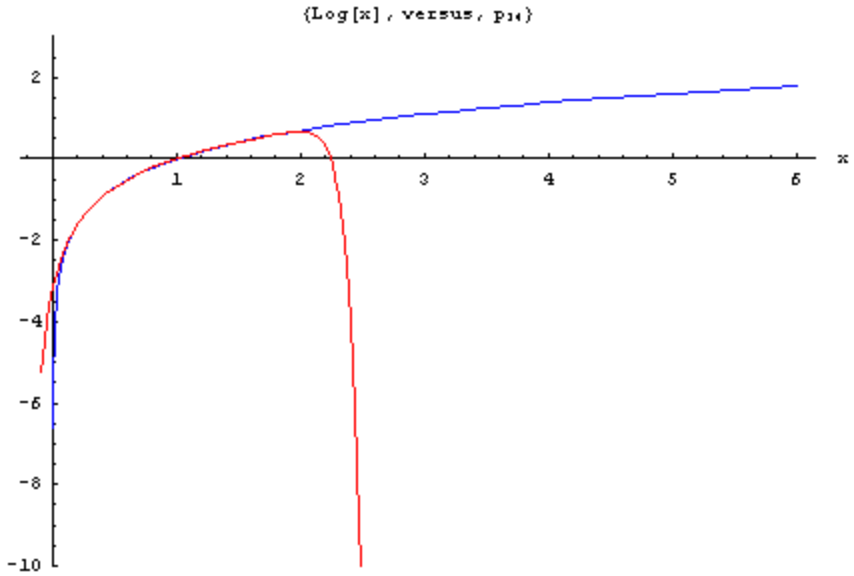

{Log[x], versus, p2}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

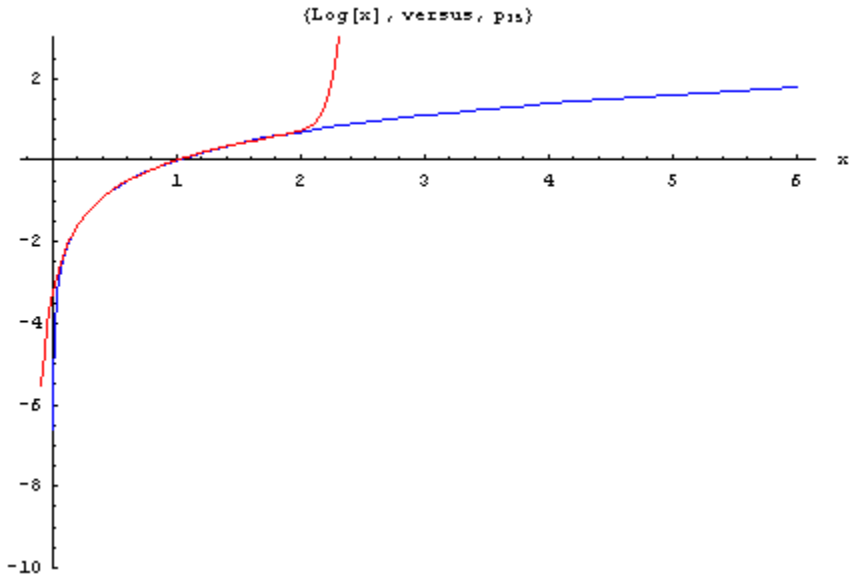

{Log[x], versus, p11}
2
1
2
3
4
5
6
x
-2
-4
-6
-8
-10

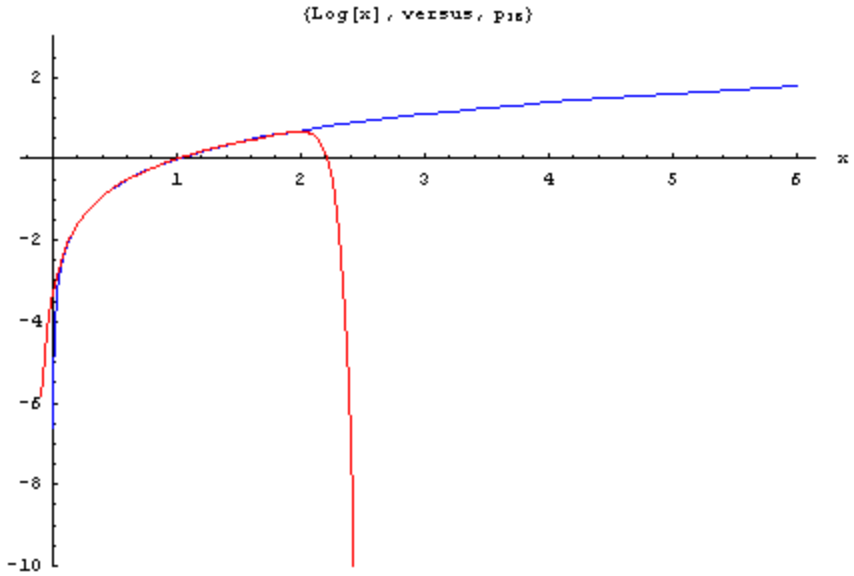

{Log[x], versus, p11}
2
1
2
3
4
5
6
x
-2
-4
-6
-8
-10

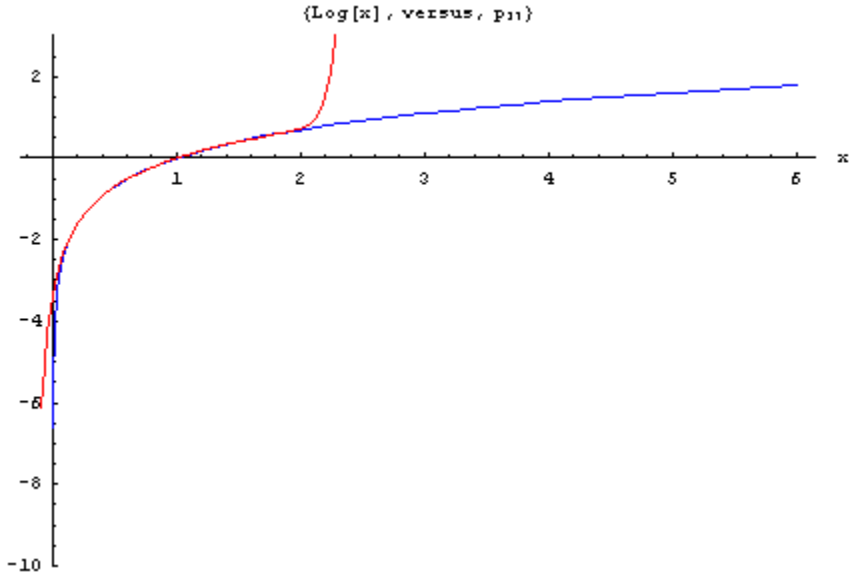

{Log[x], versus, p12}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

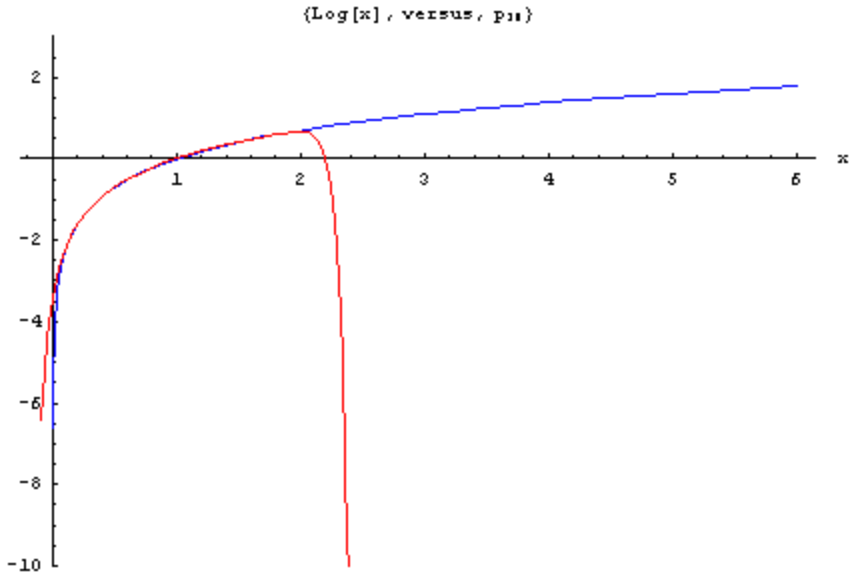

{Log[x], versus, p13}
x
2
1
2
3
4
5
6
-2
-4
-6
-8
-10

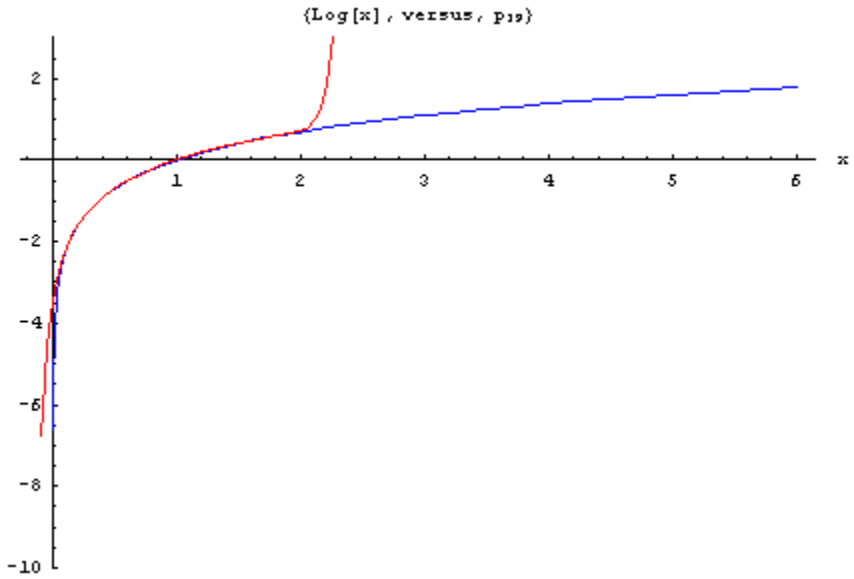

{Log[x], versus, p14}
2
-2
-4
-6
-8
-10
1
2
3
4
5
6
x

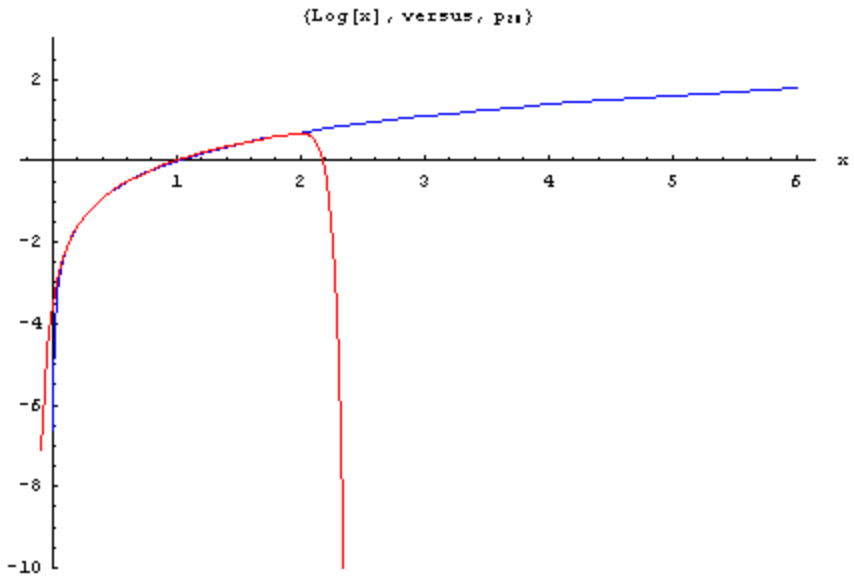

{Log[x], versus, p15}
2
-2
-4
-6
-8
-10
1
2
3
4
5
6
x


{Log[x], versus, p16}
2
1
2
3
4
5
6
x
-2
-4
-6
-8
-10


{Log[x], versus, p11}
2
1
2
3
4
5
6
x
-2
-4
-6
-8
-10


{Log[x], versus, p11}
2
1
2
3
4
5
6
x
-2
-4
-6
-8
-10


{Log[x], versus, p19}
2
1
2
3
4
5
6
x
-2
-4
-6
-8
-10



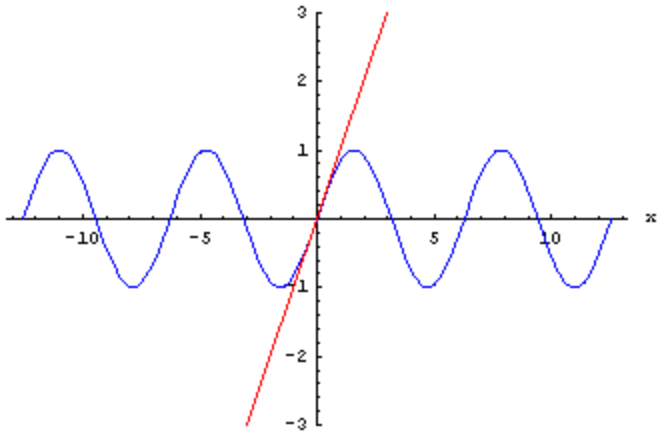
The Taylor polynomial seems to approximate the function well in the interval between 0 and 2, but look at what is happening beyond x=2.

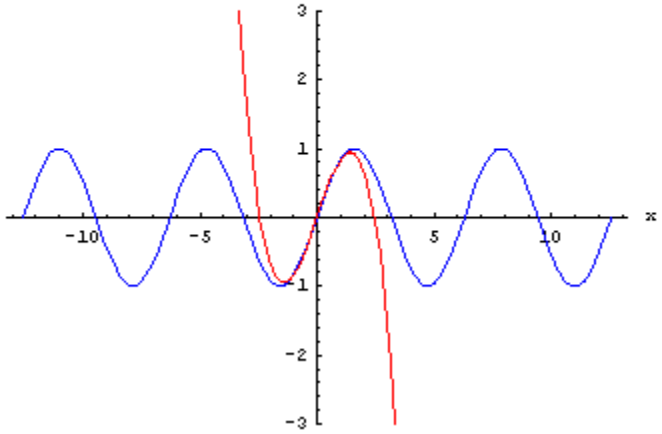
Part II: Demonstrations of Convergence and Lack of Convergence

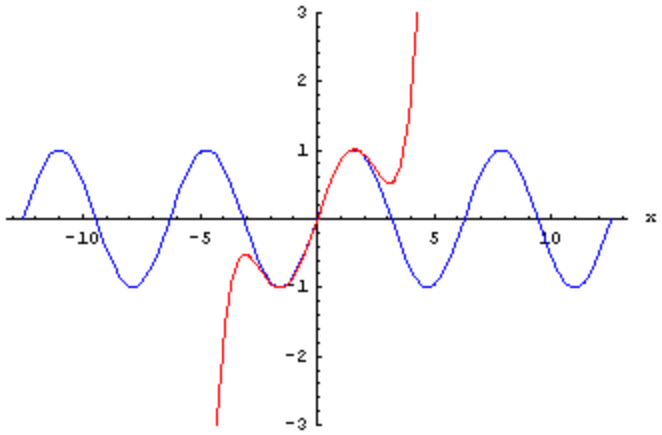
The **taylorpolydemo[f_, a_, nrange_, xlimits_, ylimits_]** command is used to demonstrate the convergence of Taylor polynomials to a function that has derivatives of all orders over some interval of its domain as the degree of the polynomial increases. The arguments are **f** (a function of *x*), **a** (the point about which the function is expanded), **nrange** (the range of degrees of polynomials to be compared with **f**, and the increment size), **xlimits** (the *x*-axis limits on the graphics window), and **ylimits** (the *y*-axis limits on the graphics window). You can animate the graphics by mouse-clicking the outermost cell bracket on the right edge of the notebook window and then typing Ctrl+Y. Following are some examples that you can use, or you can add some of your own.

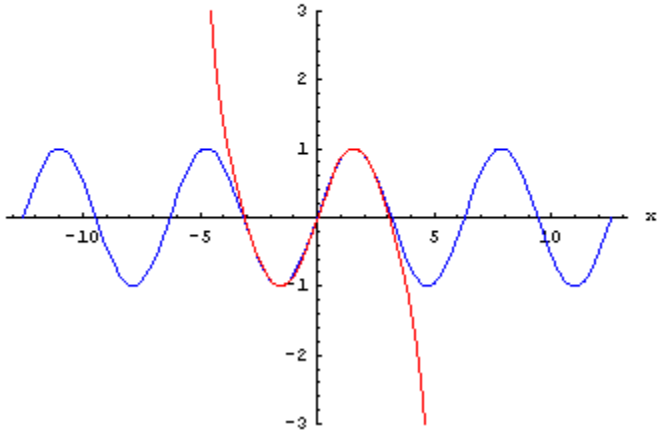
In[67]:=

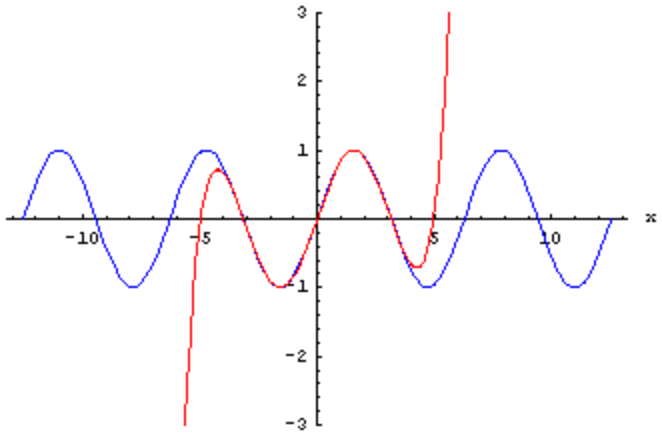
```
taylorpolydemo[Sin[x], 0, {1, 19, 2}, {-4 * Pi,
 {-3, 3}];
```

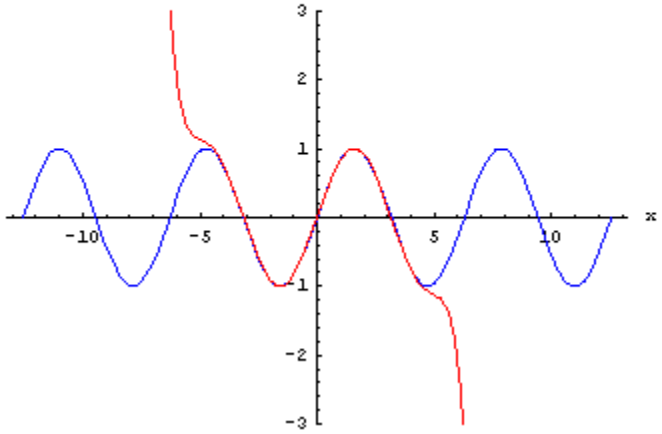

3
2
1
-10
-5
5
10
x
-1
-2
-3

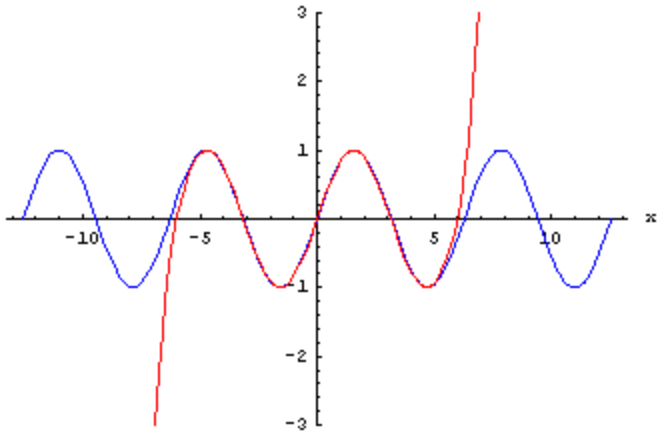

3
2
1
-10
-5
5
10
x
-1
-2
-3

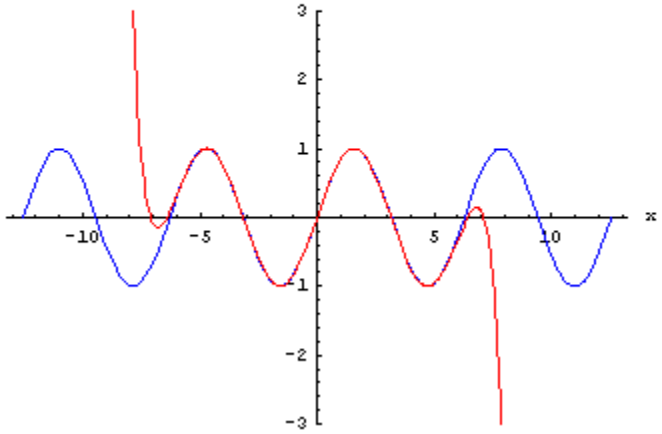

3
2
1
-10
-5
5
10
x
-1
-2
-3

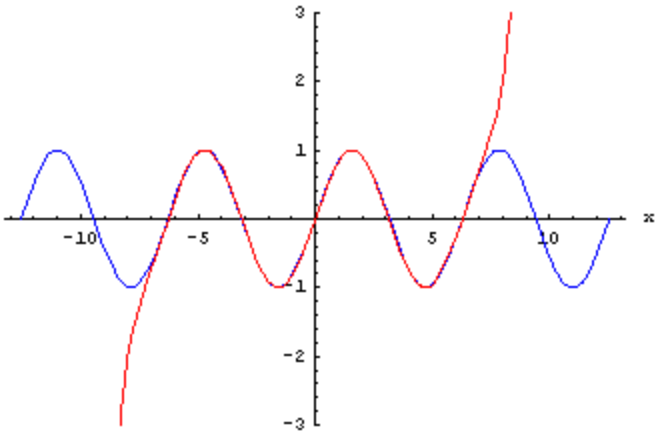

3
2
1
-10
-5
5
10
x
-1
-2
-3

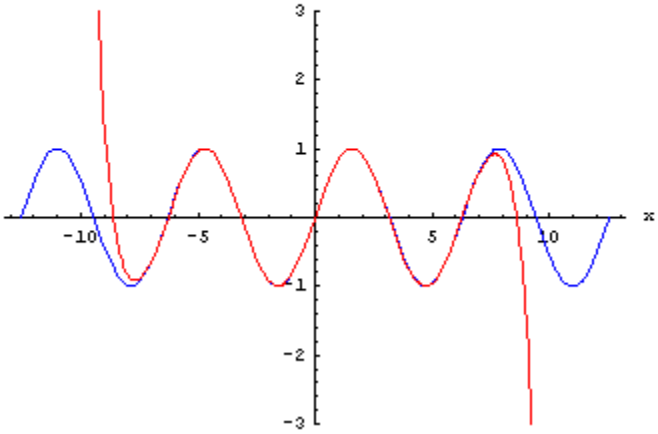

3
2
1
-10
-5
5
10
x
-1
-2
-3


3
2
1
-10
-5
5
10
x
-1
-2
-3


3
2
1
-10
-5
5
10
x
-1
-2
-3


3
2
1
-10
-5
5
10
x
-1
-2
-3

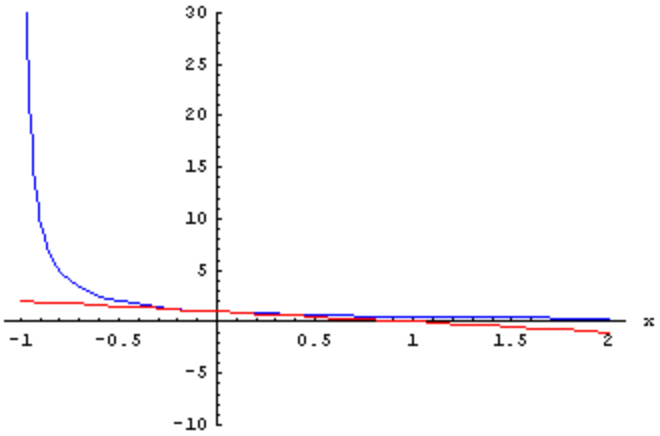


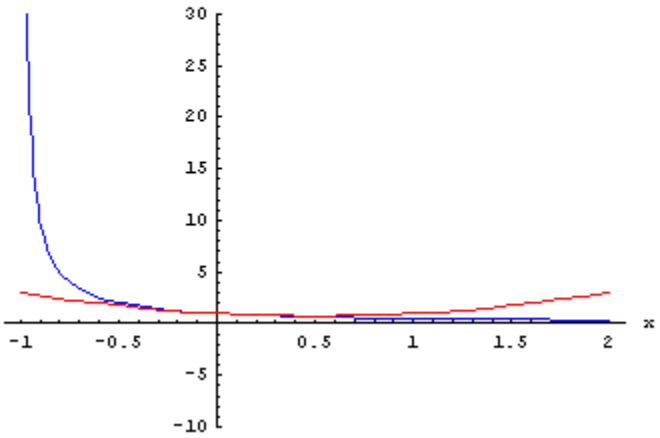


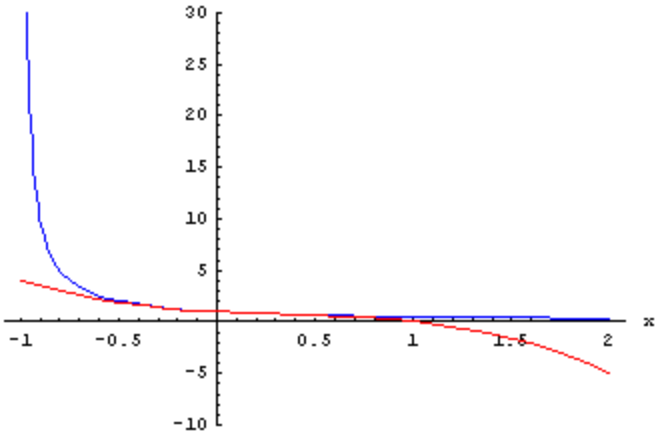
The sine function you just plotted is very well behaved. Consider the following rational function, and see what happens.

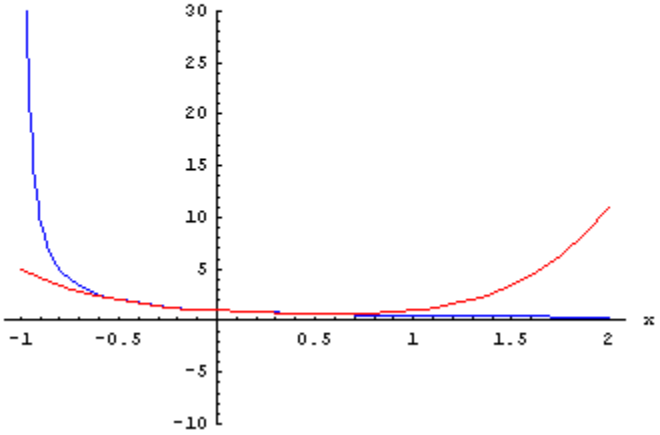
In[68]:=

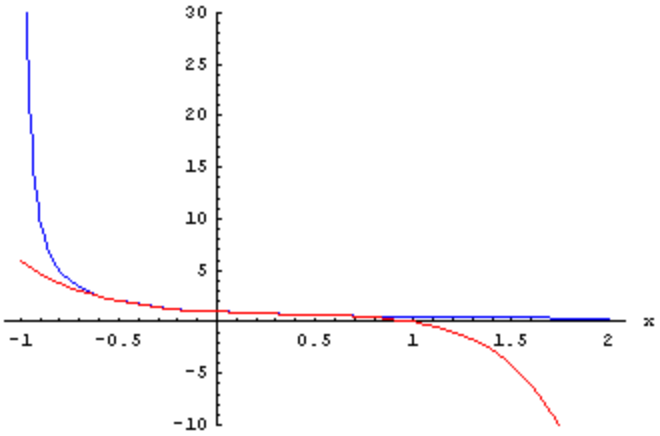
```
taylorpolydemo[1/(x+1), 0, {1, 20, 1}, {-1, 2
```

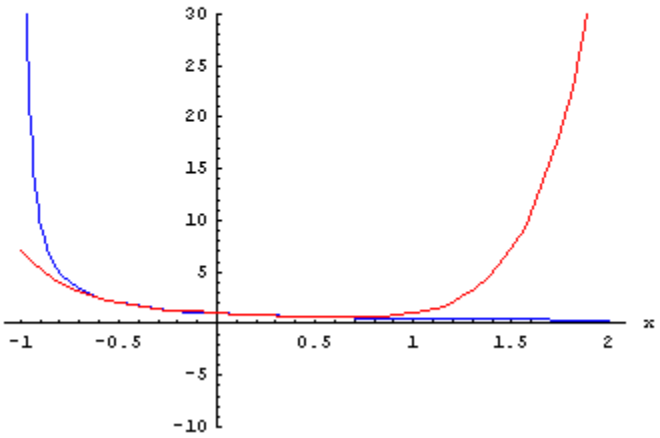

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

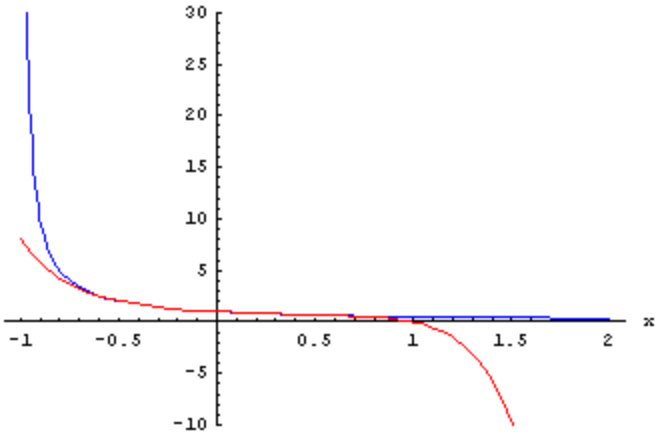

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

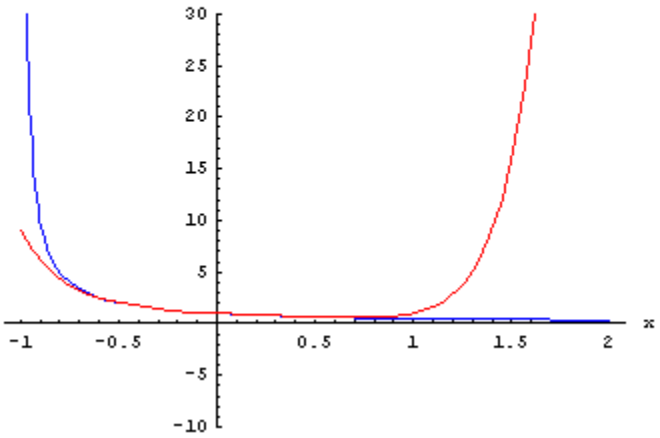

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

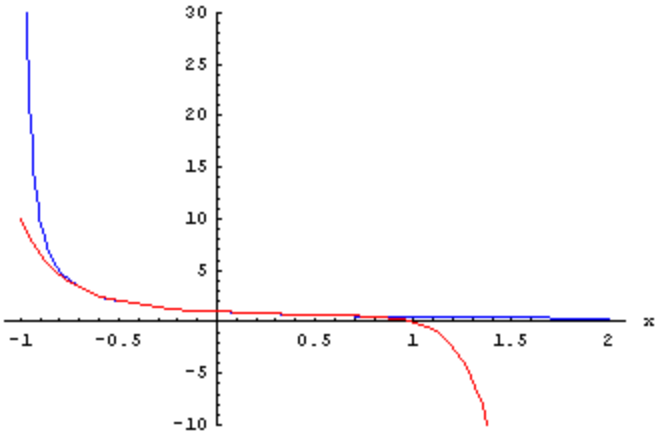

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

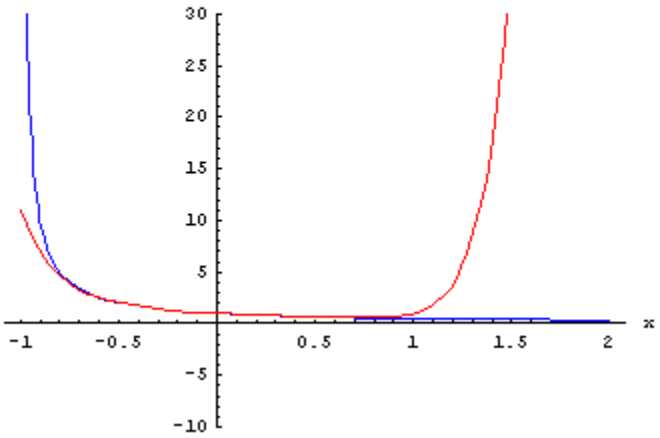

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

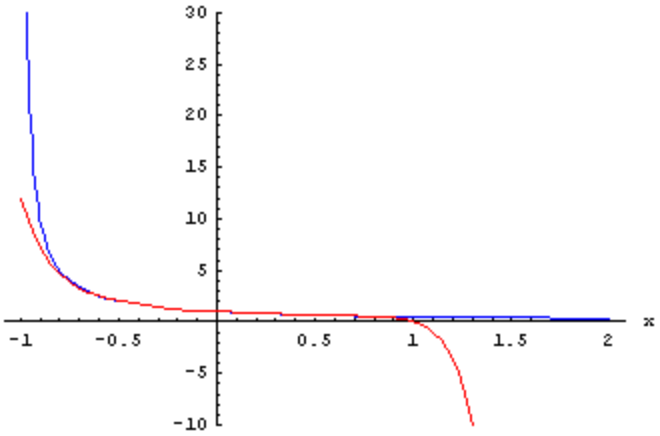

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

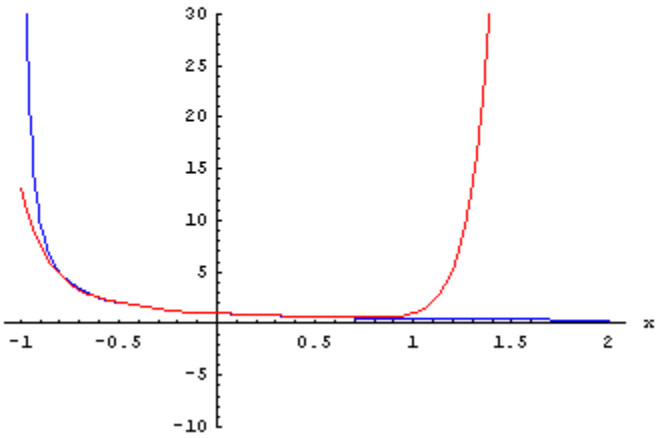

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

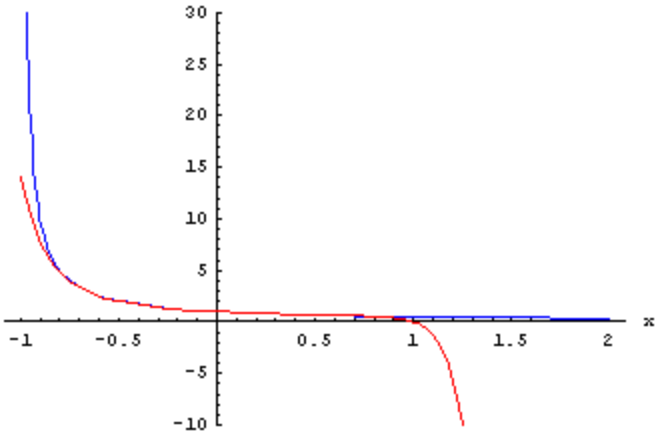

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

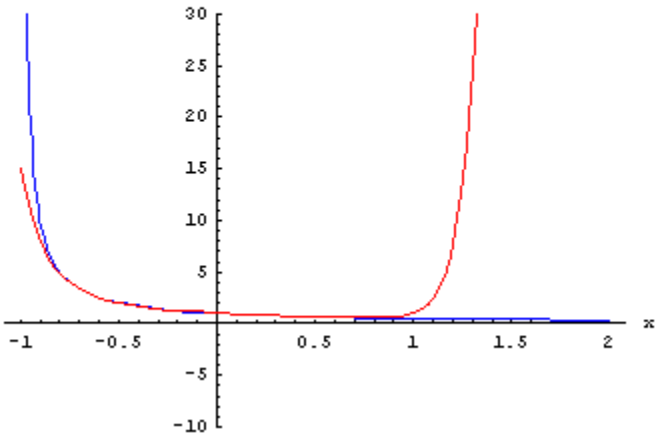

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

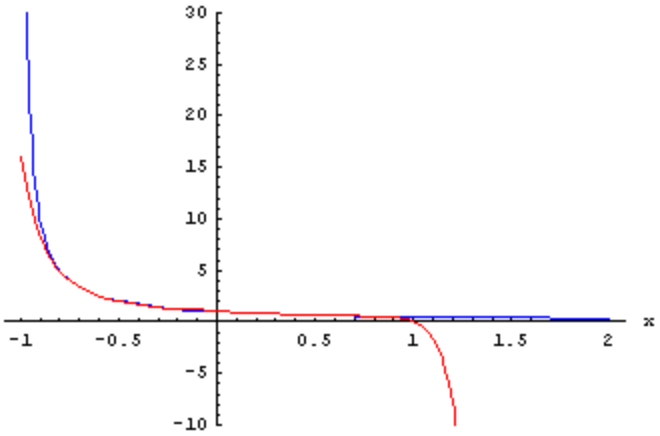

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

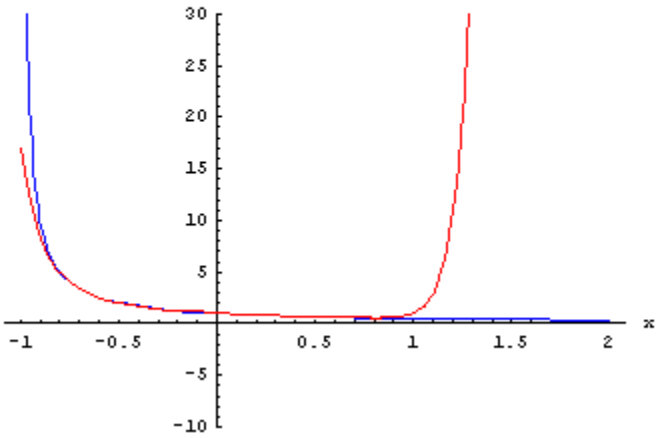

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

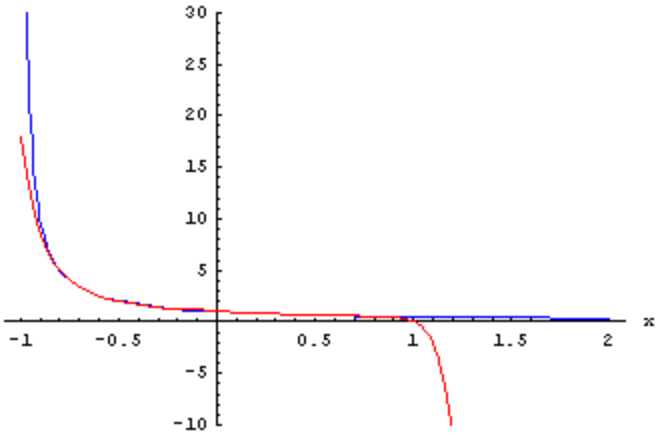

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

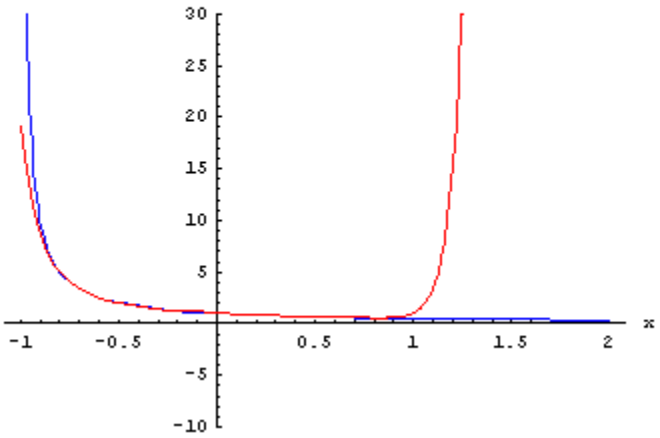

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

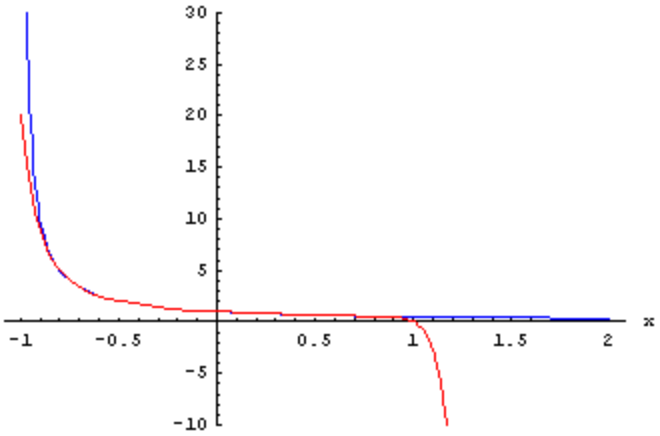

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10

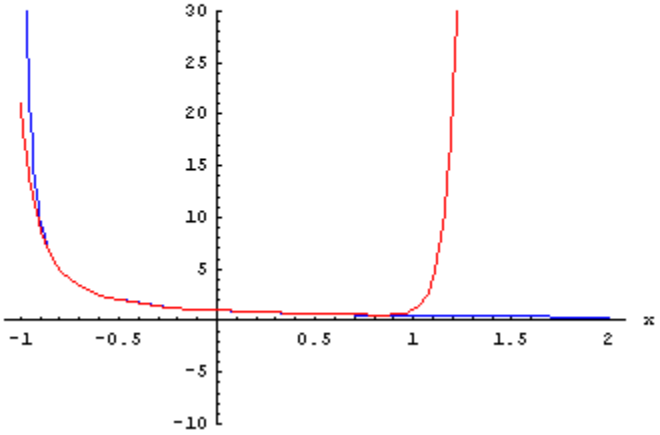

30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10


30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10


30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10


30
25
20
15
10
5
-1
-0.5
0.5
1
1.5
2
x
-5
-10





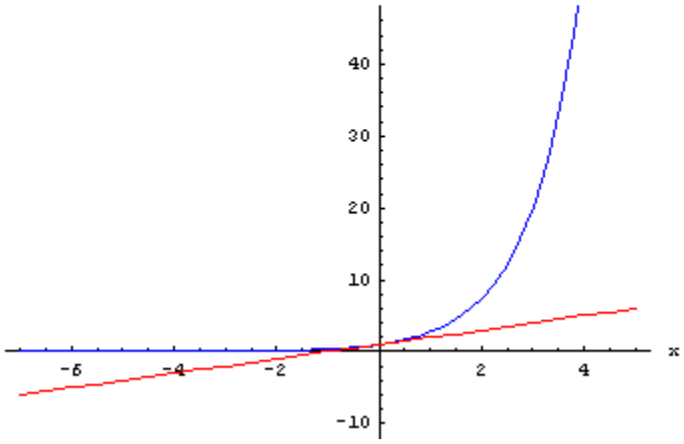
You Try It: Part II

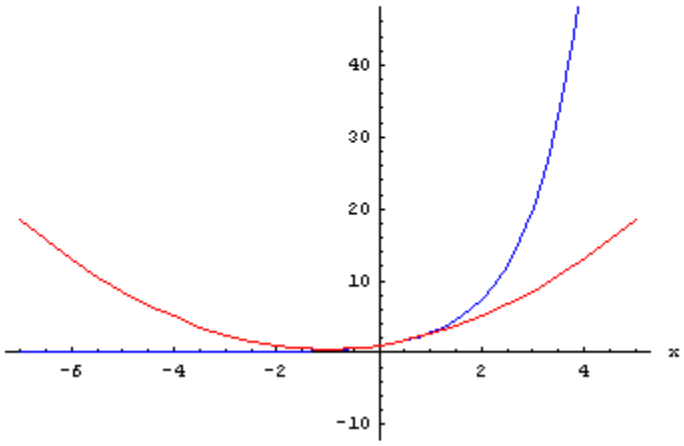
Select your own function, and change the terms in red. Be careful to use correct *Mathematica* format when inserting your own functions. Some functions that you may want to try, if you haven't already, are: **Cos[x]** at a=0, **Exp[x]** at a=0, **Log[x]** (the natural log) at a=1, and **Sqrt [x]** at a=4. The commands in the next cell are for **Exp[x]** at a=0.

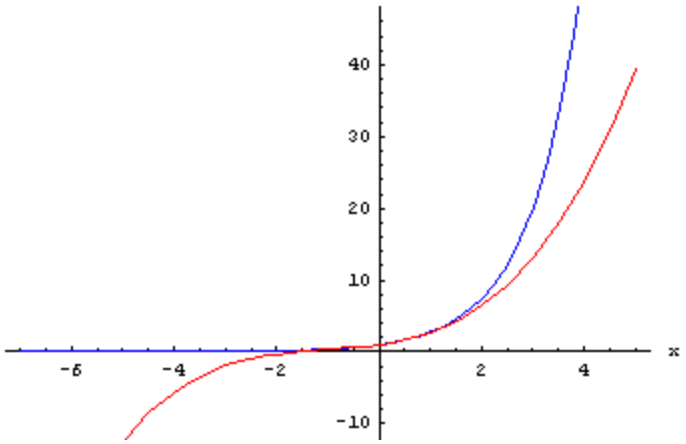
In[69]:=

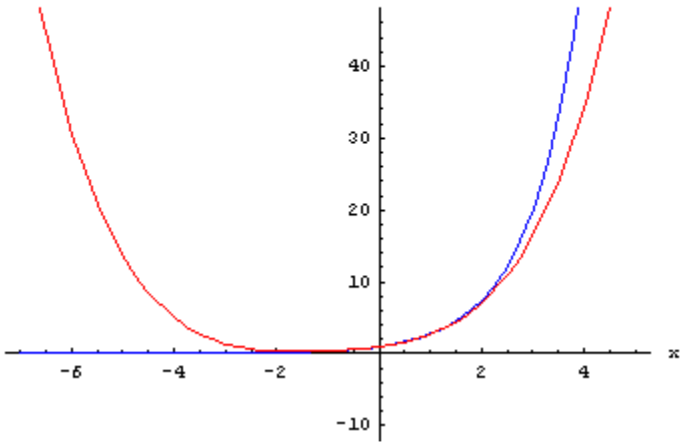
```
f = Exp[x];
a = 0;
```

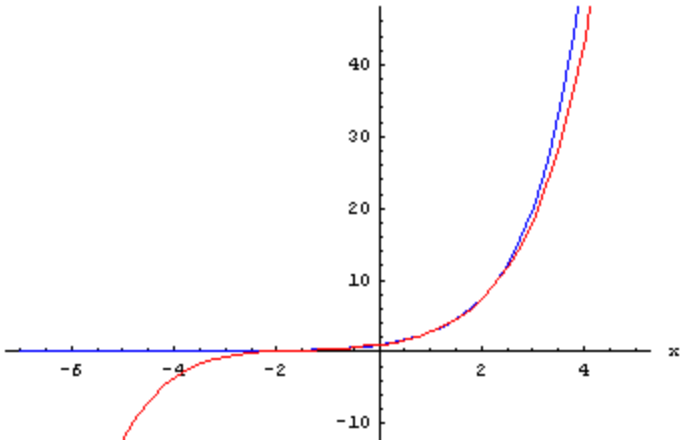
```
xlimits = {-7, 5};

ylimits = {-12, 48};

nrange = {1, 14, 1};

taylorpolydemo[f, a, nrange, xlimits, ylimits
```

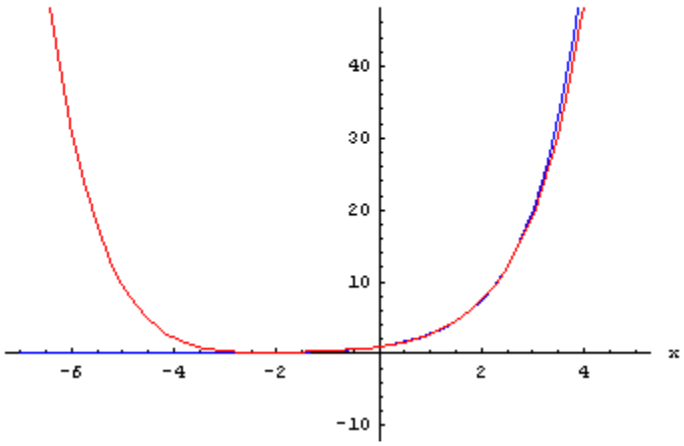


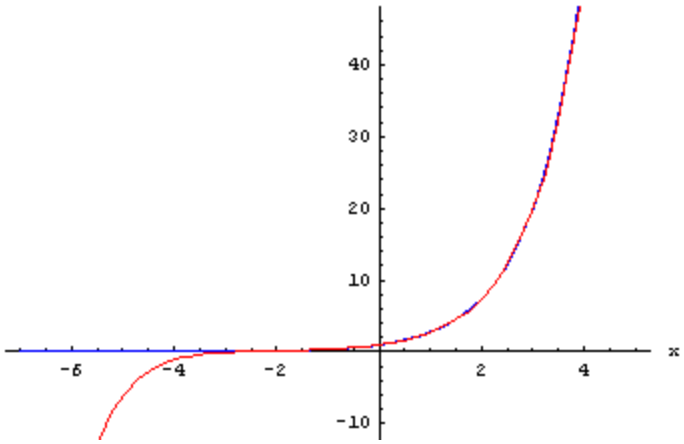


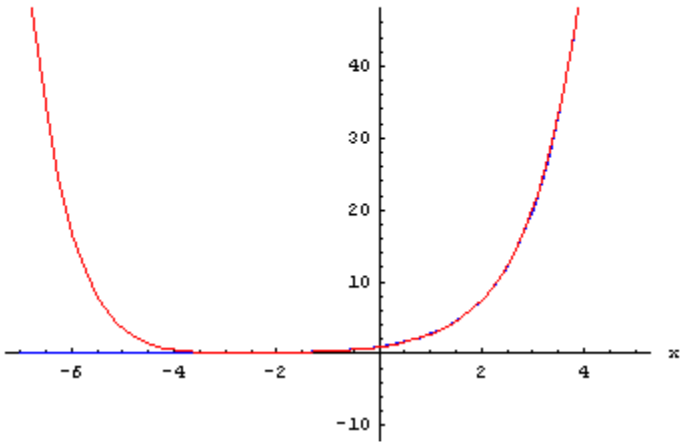

40
30
20
10
-10
-6
-4
-2
2
4
x

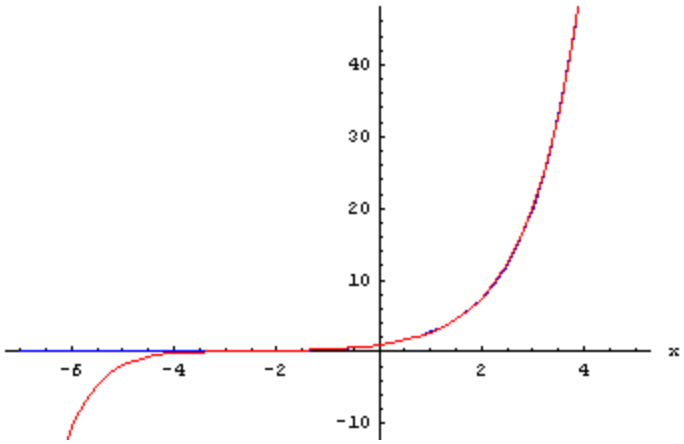

40
30
20
10
-10
-6
-4
-2
2
4
x

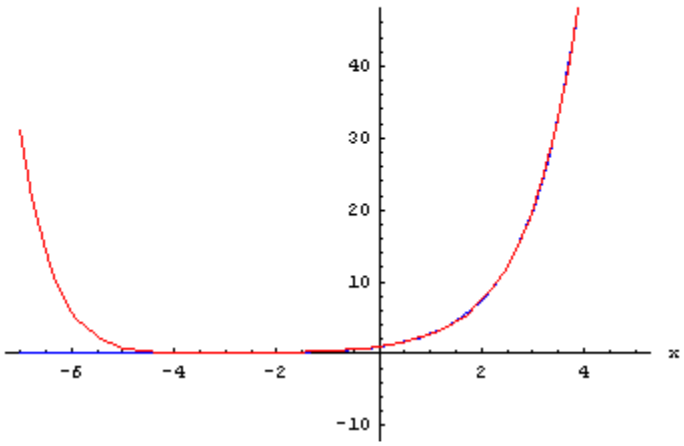

40
30
20
10
-10
-6
-4
-2
2
4
x

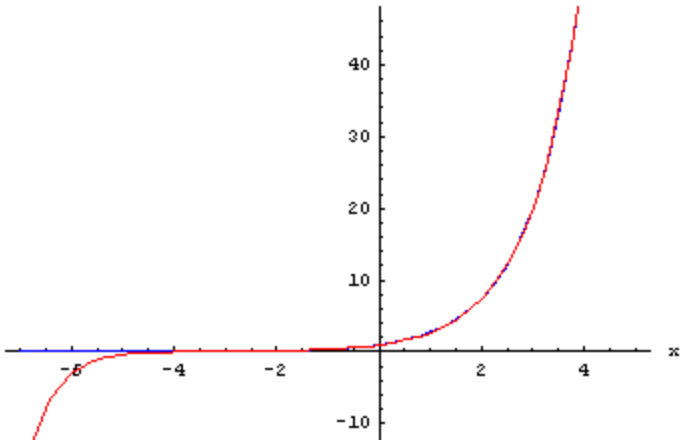

40
30
20
10
-10
-6
-4
-2
2
4
x

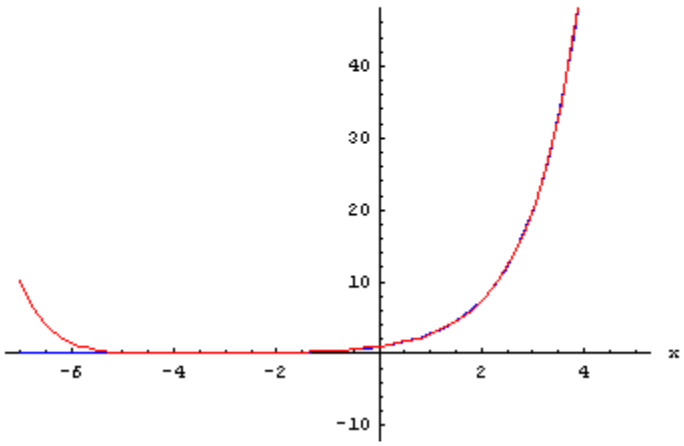

40
30
20
10
-10
-6
-4
-2
2
4
x

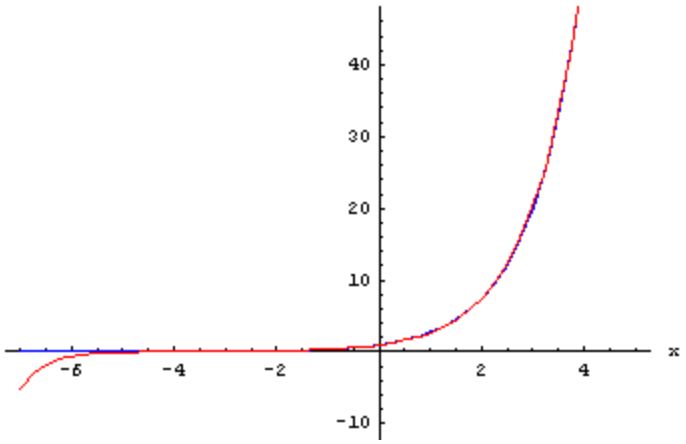

40
30
20
10
-10
-6
-4
-2
2
4
x

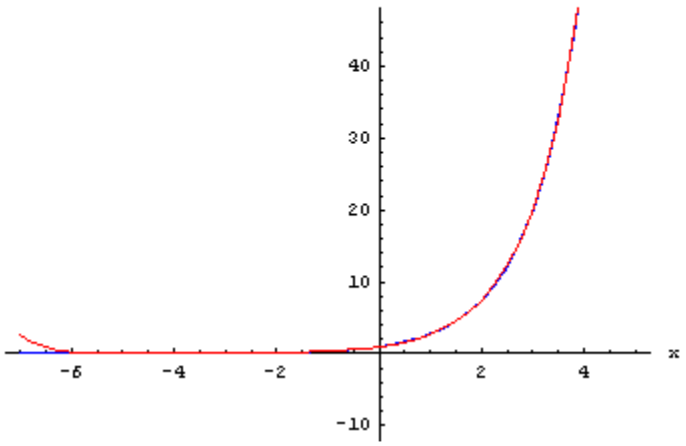

40
30
20
10
-10
-6
-4
-2
2
4
x


40
30
20
10
-10
-6
-4
-2
2
4
x


40
30
20
10
-10
-6
-4
-2
2
4
x


40
30
20
10
-10
-6
-4
-2
2
4
x





About *Mathematica*

Note that in the **Plot[ ]** and **Show[ ]** commands, the **DisplayFunction → Identity** option turns off the graphics display generated by the command. The **DisplayFunction → $DisplayFunction** turns on the graphics display. If an option is not specified, by default the graphics will display. Go back.