```
(*Part 1a:*)
(*This function takes in an equation (in the form of a function) and a tolerance level,
performs the Regula Falsi method to find a root between 1 and 3,
then outputs a list with the values we need to put in the table.*)
regulaFalsi[eq_, ϵ_] := (
  count = 1; a = 1; b = 3;
  computerRoot = r /. FindRoot[{eq[r]}, {r, 2}];
  While[count ≤ 100, p = N[b - eq[b] * (b - a) / (eq[b] - eq[a]), 32];
   (*About this next line: the primary stopping method in the book does not work,
   because the difference between p and the last p does not tell
    us anything about our max error; this is because the root is
    not guaranteed to be between p and the last p. In practice,
   this method will stop the first equation for tolerance = 0.1, when its absolute error
    is above 0.1. I searched for stopping methods online. The most popular one seems
    to be: "If Abs[eq[p]]<ϵ, then break". The problem with this is that it changes the
       meaning of our "ϵ" from being a value that dictates how far off we can be
       in the x-direction to a value that dictates how far off we can be in the y-
      direction. This would not work for this assignment,
   because we have to compare this Regula Falsi method to the Bisection method,
   which uses an ϵ that refers to the error in the x-direction. So,
   I decided to just use the Mathematica calculated root in the stopping method,
   because it is the only way I can think of to stop this
    calculation while keeping the meaning of "ϵ" the same as it
    was in the Bisection method. I know this is basically cheating,
   but this is the only way I can think of that allows me to do
    the next steps (the comparing to Bisection method step)
    of this assignment properly. If you have the time,
   please inform me if my evaluation of the method given in the book is
    incorrect or if there is a better stopping method that I missed.*)
   If[Abs[p - computerRoot] < ϵ, Break[]];
   count++;
   If[eq[p] * eq[b] < 0, a = b];
   b = p]; computerRoot = r /. FindRoot[{eq[r]}, {r, 2}];
  Return[{"y =" eq[x], ϵ, N[p, 5], computerRoot, Abs[p - computerRoot], count}]
 )

In[•]:= (*These three lines translate the equations into
 functions. We want to rewrite the equations as functions so that
 the bisectionTechnique function can take the equations as inputs.*)
equation1[x_] := x^5 - 2 x^4 - 1
equation2[x_] := x^3 - x - 1
equation3[x_] := x^2 + 2 x - 5
```

In[•]:=
```
(*This cell makes a grid for the three
 equations for the four different tolerance levels.*)
Grid[{regulaFalsi[equation1, 0.1], regulaFalsi[equation1, 0.05],
  regulaFalsi[equation1, 0.01], regulaFalsi[equation1, 0.001],
  regulaFalsi[equation2, 0.1], regulaFalsi[equation2, 0.05], regulaFalsi[equation2, 0.01],
  regulaFalsi[equation2, 0.001], regulaFalsi[equation3, 0.1], regulaFalsi[equation3, 0.05],
  regulaFalsi[equation3, 0.01], regulaFalsi[equation3, 0.001]}, Frame → All]

(*This cell aligns everything in the grid.*)
Insert[%25, Alignment → Left, 2]
```

In[•]:=
```
(*This cell titles the columns.*)
ReplacePart[%26,
 1 → Prepend[First[%26], {"Polynomial", "Error Tolerance (∈)", "Root from Regula Falsi",
    "Actual Root", "Actual Error", "Number of Steps Regula Falsi method Used"}]]
```

Out[•]=

Polynomial	Error Tolerance (∈)	Root from Regula Falsi	Actual Root	Actual Error	Number of Steps Regula Falsi method Used
$y = \left(-1 - 2x^4 + x^5\right)$	0.1	1.9701	2.05597	0.0858582	19
$y = \left(-1 - 2x^4 + x^5\right)$	0.05	2.0148	2.05597	0.0411209	22
$y = \left(-1 - 2x^4 + x^5\right)$	0.01	2.0473	2.05597	0.00869315	28
$y = \left(-1 - 2x^4 + x^5\right)$	0.001	2.0552	2.05597	0.000799469	37
$y = \left(-1 - x + x^3\right)$	0.1	1.2349	1.32472	0.0897825	4
$y = \left(-1 - x + x^3\right)$	0.05	1.2805	1.32472	0.0442236	6
$y = \left(-1 - x + x^3\right)$	0.01	1.3176	1.32472	0.00708411	11
$y = \left(-1 - x + x^3\right)$	0.001	1.3240	1.32472	0.000763978	17
$y = \left(-5 + 2x + x^2\right)$	0.1	1.4211	1.44949	0.0284371	2
$y = \left(-5 + 2x + x^2\right)$	0.05	1.4211	1.44949	0.0284371	2
$y = \left(-5 + 2x + x^2\right)$	0.01	1.4426	1.44949	0.00686679	3
$y = \left(-5 + 2x + x^2\right)$	0.001	1.4491	1.44949	0.000397399	5

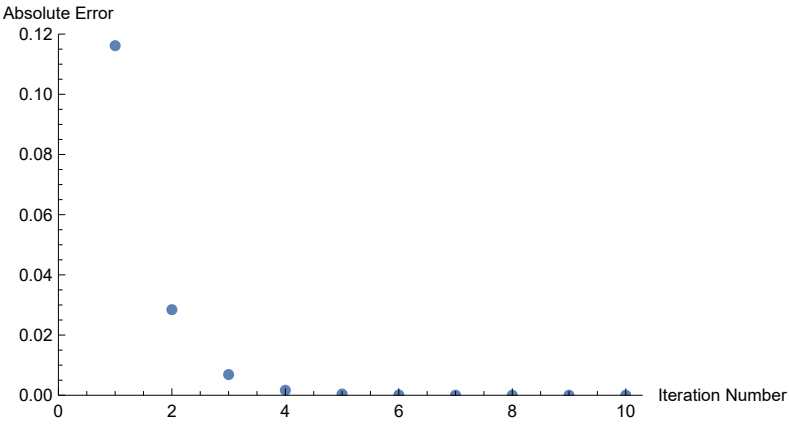
In[ ]:=
```
(*Part 1b:*)
(*This function will take an iteration number as an input,
apply that many iterations of the Regula Falsi to the 3rd equation,
and output the absolute error after that many iterations.*)
rfError3rdEquation[iteration_] := ( count = 1; a = 1; b = 3;
  computerRoot = r /. FindRoot[{equation3[r]}, {r, 2}];
  While[count ≤ iteration,
   p = N[b - equation3[b] * (b - a) / (equation3[b] - equation3[a]), 32];
   count++;
   If[equation3[p] * equation3[b] < 0, a = b];
   b = p];
  Return[Abs[p - computerRoot]]

 )
```

In[ ]:=
```
rfError3rdEquation[1]
```

Out[ ]=
```
0.116156
```

In[ ]:=
```
(*This cell will first create a table with iterations numbers 1 through 10
 and the rfError3rdEquation output for each of the iteration numbers. Then,
it will plot the table and label each of the axes of the plot.*)
errorTable = Table[{i, rfError3rdEquation[i]}, {i, 1, 10}];
ListPlot[errorTable, PlotMarkers → {Automatic, 6},
 AxesLabel → {"Iteration Number", "Absolute Error"}, PlotRange → {{0, 10.3}, {0, .12}}]
```

Out[ ]=



Unlike the bisection method, the Regula Falsi method seems to always reduce absolute error when iteration number is increased.

In[ ]:=
```
(*Comparing Bisection and Regula Falsi*)
(*This function is like the normal regulaFalsi but only returns count.*)
regulaFalsiCounter[eq_, ϵ_] := (
  count = 1; a = 1; b = 3;
  computerRoot = r /. FindRoot[{eq[r]}, {r, 2}];
  While[count ≤ 100, p = N[b - eq[b] * (b - a) / (eq[b] - eq[a]), 32];
   If[Abs[p - computerRoot] < ϵ, Break[]];
   count++;
   If[eq[p] * eq[b] < 0, a = b];
   b = p]; computerRoot = r /. FindRoot[{eq[r]}, {r, 2}];
  Return[count]
 )


(*This function is like the normal bisectionTechnique
 from previous assignment but only returns count.*)
bisectionTechniqueCounter[eq_, ϵ_] := (
  count = 0; a = 1; b = 3;
  While[count ≤ 1000, p = (a + (b - a) / 2);
   If[(eq[p] == 0) || (((b - a) / 2) < ϵ), myRoot = p;
    Break[]];
   count++ ;
   If[eq[p] * eq[a] > 0, a = p, b = p]];
  computerRoot = r /. FindRoot[{eq[r]}, {r, 2}]; Return[count])
```

In[ ]:=
```
(*Creates a list of lists of elements that
 will be in our next table to compare the two methods.*)
countTableParts = Table[{"y =" i[x], j, regulaFalsiCounter[i, j],
   bisectionTechniqueCounter[i, j], Ceiling[(Log2[2 / j] - 1)]},
  {i, {equation1, equation2, equation3}}, {j, {0.1, 0.05, 0.01, 0.001}}];
countTable = Join[countTableParts[[1]], countTableParts[[2]], countTableParts[[3]]
```

Out[ ]=
$$\{\{y = (-1 - 2x^4 + x^5), 0.1, 19, 4, 4\}, \{y = (-1 - 2x^4 + x^5), 0.05, 22, 5, 5\},$$
$$\{y = (-1 - 2x^4 + x^5), 0.01, 28, 7, 7\}, \{y = (-1 - 2x^4 + x^5), 0.001, 37, 10, 10\},$$
$$\{y = (-1 - x + x^3), 0.1, 4, 4, 4\}, \{y = (-1 - x + x^3), 0.05, 6, 5, 5\},$$
$$\{y = (-1 - x + x^3), 0.01, 11, 7, 7\}, \{y = (-1 - x + x^3), 0.001, 17, 10, 10\},$$
$$\{y = (-5 + 2x + x^2), 0.1, 2, 4, 4\}, \{y = (-5 + 2x + x^2), 0.05, 2, 5, 5\},$$
$$\{y = (-5 + 2x + x^2), 0.01, 3, 7, 7\}, \{y = (-5 + 2x + x^2), 0.001, 5, 10, 10\}\}$$

In[ ]:=
```
(*This cell creates a grid from the list.*)
Grid[countTable, Frame → All]
```

In[ ]:=
```
(*This cell aligns the grid.*)
Insert[%127, Alignment → Left, 2]
```

In[•]:=
```
(*This cell titles the columns.*)
ReplacePart[%128, 1 → Prepend[First[%128],
   {"Polynomials", "Error Tolerance (∈)", "Regula Falsi method: Number of Steps",
    "Bisection Method:  Number of Steps", "Bisection Method: Maximum Number of Steps"}]]
```

Out[•]=

Polynomials	Error Tolerance (∈)	Regula Falsi method: Number of Steps	Bisection Method: Number of Steps	Bisection Method: Maximum Number of Steps
$y = \left(-1 - 2x^4 + x^5\right)$	0.1	19	4	4
$y = \left(-1 - 2x^4 + x^5\right)$	0.05	22	5	5
$y = \left(-1 - 2x^4 + x^5\right)$	0.01	28	7	7
$y = \left(-1 - 2x^4 + x^5\right)$	0.001	37	10	10
$y = \left(-1 - x + x^3\right)$	0.1	4	4	4
$y = \left(-1 - x + x^3\right)$	0.05	6	5	5
$y = \left(-1 - x + x^3\right)$	0.01	11	7	7
$y = \left(-1 - x + x^3\right)$	0.001	17	10	10
$y = \left(-5 + 2x + x^2\right)$	0.1	2	4	4
$y = \left(-5 + 2x + x^2\right)$	0.05	2	5	5
$y = \left(-5 + 2x + x^2\right)$	0.01	3	7	7
$y = \left(-5 + 2x + x^2\right)$	0.001	5	10	10

The Regula Falsi converges more quickly, when we are checking inside an interval in which the graph is close to a straight line in that interval. In other words, Regula Falsi converges to actual root more quickly when the derivative of the line throughout the interval is significantly close to a constant (its average rate of change).