Abrar Imon, Assignment 4

In[28]:=
```
(*Part 1*)
(*This cell creates a function that takes in an equation, an error tolerance,
and a seed value. Performs the newton's method on the equation,
starting with the seed value and ending when the difference between
 two consecutive guess for a root is less than the error tolerance. This
 function also has Mathematica calculate the actual root. Then,
this function will return a list values which we will use to create a table.*)
newtonMethod[eq_, ϵ_, seed_] := (
  count = 1; lastP = seed;
  While[count ≤ 250, p = N[lastP - (eq[lastP] / eq'[lastP]), 32];
   If[(eq[p] == 0) || ((Abs[lastP - p]) < ϵ), Break[]];
   count++; lastP = p];
  computerRoot = r /. FindRoot[{eq[r]}, {r, 2}];
  Return[{"y =" eq[x], ϵ, N[p, 6], computerRoot, Abs[p - computerRoot], count}]
 )
```

In[ ]:=
```
(*This cell translates the equations into functions. We want to rewrite the equations as
 functions so that the newtonMethod 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[{newtonMethod[equation1, 0.1, 1], newtonMethod[equation1, 0.05, 1],
  newtonMethod[equation1, 0.01, 1], newtonMethod[equation1, 0.001, 1],
  newtonMethod[equation2, 0.1, 1], newtonMethod[equation2, 0.05, 1],
  newtonMethod[equation2, 0.01, 1], newtonMethod[equation2, 0.001, 1],
  newtonMethod[equation3, 0.1, 1], newtonMethod[equation3, 0.05, 1],
  newtonMethod[equation3, 0.01, 1], newtonMethod[equation3, 0.001, 1]}, Frame → All]
```

In[ ]:=
```
(*This cell titles each of the columns of the grid.*)
ReplacePart[%15, 1 →
  Prepend[First[%15], {"Polynomial", "Error Tolerance (ϵ)", "Root from newton's method",
    "Actual root", "Actual error", "Number of iterations used by newton's method"}]]
```

In[ ]:=
```
(*This cell aligns everything in the grid.*)
Insert[%20, Alignment → Left, 2]
```

Out[ ]=

Polynomial	Error Tolerance (ϵ)	Root from newton's method	Actual root	Actual error	Number of iterations used by newton's method
$y = (-1 - 2x^4 + x^5)$	0.1	2.064	2.05597	0.00842715	42
$y = (-1 - 2x^4 + x^5)$	0.05	2.06	2.05597	0.0001273	43
$y = (-1 - 2x^4 + x^5)$	0.01	2.06	2.05597	0.0001273	43
$y = (-1 - 2x^4 + x^5)$	0.001	2.06	2.05597	0.0001273	43
$y = (-1 - x + x^3)$	0.1	1.32520	1.32472	0.000482442	3
$y = (-1 - x + x^3)$	0.05	1.32520	1.32472	0.000482442	3
$y = (-1 - x + x^3)$	0.01	1.32472	1.32472	2.16754×10^{-7}	4
$y = (-1 - x + x^3)$	0.001	1.32472	1.32472	2.16754×10^{-7}	4
$y = (-5 + 2x + x^2)$	0.1	1.45000	1.44949	0.000510257	2
$y = (-5 + 2x + x^2)$	0.05	1.44949	1.44949	5.31352×10^{-8}	3
$y = (-5 + 2x + x^2)$	0.01	1.44949	1.44949	5.31352×10^{-8}	3
$y = (-5 + 2x + x^2)$	0.001	1.44949	1.44949	5.31352×10^{-8}	3

In[19]:=
```
(*Now we will work on creating a table to compare the number of iterations
 from newton's method with the previous two root finding methods.*)
(*These are the three functions,
one for each of the root finding method we have coded so far. I have
 modified those to only return the number of iterations they use.*)
bisectionCounter[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[24]:=
```
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]
  )
```

In[54]:=
```
(*The newton method worked for the the equations with
 seed equal to 1. So we do not need to keep "seed" as an input.*)
newtonMethodCounter[eq_, ϵ_] := (
   count = 1; lastP = 1;
   While[count ≤ 250, p = N[lastP - (eq[lastP] / eq'[lastP]), 32];
    If[(eq[p] == 0) || ((Abs[lastP - p]) < ϵ), Break[]];
    count++; lastP = p];
   computerRoot = r /. FindRoot[{eq[r]}, {r, 2}];
   Return[count]
  )
```

In[60]:=
```
(*These next two cells will work to create a list of lists,
called "listFinal", which will have the information we
 need for the next grid in the format we need for the next grid.*)
list = Table[{"y =" eq[x], ϵ, bisectionCounter[eq, ϵ],
    regulaFalsiCounter[eq, ϵ], newtonMethodCounter[eq, ϵ]},
   {eq, {equation1, equation2, equation3}}, {ϵ, {0.1, 0.05, 0.01, 0.001}}]
```

In[62]:=
```
listFinal = Join[list[[1]], list[[2]], list[[3]]]
```

Out[62]= $\{\{y\ =\ (-1-2x^4+x^5),\ 0.1,\ 4,\ 19,\ 42\},\ \{y\ =\ (-1-2x^4+x^5),\ 0.05,\ 5,\ 22,\ 43\},$
$\{y\ =\ (-1-2x^4+x^5),\ 0.01,\ 7,\ 28,\ 43\},\ \{y\ =\ (-1-2x^4+x^5),\ 0.001,\ 10,\ 37,\ 43\},$
$\{y\ =\ (-1-x+x^3),\ 0.1,\ 4,\ 4,\ 3\},\ \{y\ =\ (-1-x+x^3),\ 0.05,\ 5,\ 6,\ 3\},$
$\{y\ =\ (-1-x+x^3),\ 0.01,\ 7,\ 11,\ 4\},\ \{y\ =\ (-1-x+x^3),\ 0.001,\ 10,\ 17,\ 4\},$
$\{y\ =\ (-5+2x+x^2),\ 0.1,\ 4,\ 2,\ 2\},\ \{y\ =\ (-5+2x+x^2),\ 0.05,\ 5,\ 2,\ 3\},$
$\{y\ =\ (-5+2x+x^2),\ 0.01,\ 7,\ 3,\ 3\},\ \{y\ =\ (-5+2x+x^2),\ 0.001,\ 10,\ 5,\ 3\}\}$

In[67]:=
```
(*This cell creates the grid.*)
Grid[listFinal, Frame → All]

(*This cell aligns the grid.*)
Insert[%63, Alignment → Left, 2]
```

In[69]:=
```
(*This cell titles the columns of the grid.*)
ReplacePart[%64, 1 → Prepend[First[%64],
   {"Polynomial", "Error Tolerance (∈)", "Bisection Method: Count of Iterations",
    "Regular Falsi: Count of Iterations", "Newton's Method: Count of Iterations"}]]
```

Out[69]=

Polynomial	Error Tolerance (∈)	Bisection Method: Count of Iterations	Regular Falsi: Count of Iterations	Newton's Method: Count of Iterations
$y = (-1-2x^4+x^5)$	0.1	4	19	42
$y = (-1-2x^4+x^5)$	0.05	5	22	43
$y = (-1-2x^4+x^5)$	0.01	7	28	43
$y = (-1-2x^4+x^5)$	0.001	10	37	43
$y = (-1-x+x^3)$	0.1	4	4	3
$y = (-1-x+x^3)$	0.05	5	6	3
$y = (-1-x+x^3)$	0.01	7	11	4
$y = (-1-x+x^3)$	0.001	10	17	4
$y = (-5+2x+x^2)$	0.1	4	2	2
$y = (-5+2x+x^2)$	0.05	5	2	3
$y = (-5+2x+x^2)$	0.01	7	3	3
$y = (-5+2x+x^2)$	0.001	10	5	3

In[76]:=
```
(*Part 2*)
(*Setup the new polynomial*)
equation4[x_] := x^3 - 2 x - 2
```

In[81]:=
```
(*To show that Newton's Method fails to converge for this new equation with seed as x =
 0, we will write a function to help us see each of the iterations
 clearly for this new poynomial and seed. This function takes in as input,
the last root the newtonMethod gave,
and uses it to compute and return the next root from Newton's Method.*)
newtonMethodStep[lastP_] := (
  p = lastP - (equation4[lastP] / equation4'[lastP]);
  Return[p]
 )
```

In[85]:=
```
(*Now, we use this function and enter 0 as seed,
and get that the next guess by the Newton Method is -1.*)
newtonMethodStep[0]
```

Out[85]= -1

In[86]:=
```
(*Now, we use -
 1 as the last root guess to compute the next root and we get that the guess is 0.*)
newtonMethodStep[-1]
```

Out[86]= 0

Since an input of 0 returns -1 and an input of -1 returns 0, the Newton's Method will never make another guess for the root of this equation other than 0 and -1 if our seed is 0. Hence, it will never converge.

```
(*Here,
I will call our original newtonMethod function to give us an estimate of the root of y =
 x^3 - 2x - 2 with a seed as x=1. We will see that it gives us
   an answer within an error tolerance of 0.01 after 6 iterations.*)
newtonMethod[equation4, 0.01, 1]
```

Out[87]= $\{y = (-2 - 2x + x^3), 0.01, 1.76930, 1.76929, 9.0432 \times 10^{-6}, 6\}$