Math 4610 HW1 - Kaden Taylor A02257212 - 9/16/2022

Task 1

This is my code for the Fixed Point Iteration that fulfills the requirements for task 1 and 2.

```
def fixedPointIter(g_expression, x0, tol, maxIter, v):
    values = []
    error = 0
    g = lambda x: eval(g_expression)
    x0 = float(x0)
    tol = float(tol)
    maxIter = int(maxIter)
    v = int(v)

    for i in range(0, maxIter):
        x1 = g(x0)
        error = abs(x1 - x0)
        x0 = x1
        if v:
            values.append(x0)
            values.append(error)
        if error < tol:
            break
    if not v:
        values.append(x0)
        values.append(error)
    return values
```

I have a main.py function that wraps up this `fixedPointIter` function, takes inputs from the user as prompted or from the command line.The commandline call is:

```
Commandline call: $python main.py f(x) g(x) a b x0 tolerance maxIteration v
```

My main function is here:

```
import fixedPointIteration as fp
import bisection as bi
import math
import sys
```

```
if len(sys.argv) > 8:
        f = sys.argv[1]
        g = sys.argv[2]
        a = sys.argv[3]
        b = sys.argv[4]
        x0 = sys.argv[5]
        tol = sys.argv[6]
        maxIter = sys.argv[7]
        v = sys.argv[8]
else:
    print("Don't add any parenthesis to the equations")
    f = input("Enter a value for f(x) = ")
    g = input("Enter a value for g(x) = ")
    a = input("Enter a value for a = ")
    b = input("Enter a value for b = ")
    x0 = input("Enter a value for x0: ")
    tol = input("Enter a value for tol: ")
    maxIter = input("Enter a value for maxIter: ")
    v = input("Print full table? 1 = Yes, 0 = No: ")

fixed = fp.fixedPointIter(g, x0, tol, maxIter, v)
bisec = bi.bisection(f, a, b, tol, maxIter, v)

function = lambda x: eval(f)

print("\nIf the full table is not requested, the final iteration will show as iteration 0\n")

print("----Fixed Point Iteration-----------------------------------------------------------------")
print("iter |             x               |            f(x)             |      Error  ")
print("------------------------------------------------------------------------------------------")
for i in range(0, int(len(fixed)/2)):
    print(f"{i:>3}  |  {fixed[i*2]:<25}  |  {function(fixed[i*2]):<25}  |  {fixed[i*2 + 1]:<25}")
print("------------------------------------------------------------------------------------------")

print("----------Bisection-----------------------------------------------------------------------")
print("iter |             x               |            f(x)             |      Error  ")
print("------------------------------------------------------------------------------------------")
for i in range(0, int(len(bisec)/2)):
    print(f"{i:>3}  |  {bisec[i*2]:<25}  |  {function(bisec[i*2]):<25}  |  {bisec[i*2 + 1]:<25}")
print("------------------------------------------------------------------------------------------")
```

The output from this input from Below is the output for `f(x) = x*math.e**-x` and `g(x) = x - f(x)` :

```
Commandline call: $python main.py x*math.e**-x x-x*math.e**-x -10 5 1 .001 20 0
```

```
----Fixed Point Iteration-----------------------------------------------------------------
iter |             x               |            f(x)             |      Error
------------------------------------------------------------------------------------------
  0  |  9.439212148590554e-10      |  9.439212139680681e-10      |  3.0722592907016176e-05
------------------------------------------------------------------------------------------
```

The output from this input from Below is the output for `f(x) = x*math.e**-x` and `g(x) = x + f(x)` :

```
Commandline call: $python main.py x*math.e**-x x-x*math.e**-x -10 5 1 .001 20 0
```

```
----Fixed Point Iteration------------------------------------------------------------------
iter |             x               |            f(x)             |      Error
-------------------------------------------------------------------------------------------
  0  |  9.439212148590554e-10      |  9.439212139680681e-10      |  3.0722592907016176e-05
-------------------------------------------------------------------------------------------
```

Task 2

There is a flag in the function call that determines if an entire table is returned or just the last iteration. The above results show the single form. If v=1 in the function call, the output looks like this:

```
Commandline call: $python main.py x*math.e**-x x-x*math.e**-x -10 5 1 .001 20 1
```

```
----Fixed Point Iteration------------------------------------------------------------------
iter |             x               |            f(x)             |      Error
-------------------------------------------------------------------------------------------
  0  |  0.6321205588285577         |  0.3359490712340276         |  0.36787944117144233

  1  |  0.29617148759453005        |  0.22025085548269294        |  0.3359490712340276
  2  |  0.07592063211183711        |  0.07037005715467573        |  0.22025085548269294
  3  |  0.0055505749571613805      |  0.0055198514203331495      |  0.07037005715467573
  4  |  3.0723536828231035e-05     |  3.0722592907016176e-05     |  0.0055198514203331495
  5  |  9.439212148590554e-10      |  9.439212139680681e-10      |  3.0722592907016176e-05
-------------------------------------------------------------------------------------------
----------Bisection------------------------------------------------------------------------
iter |             x               |            f(x)             |      Error
-------------------------------------------------------------------------------------------
  0  |  -2.5                       |  -30.456234901758677        |  15.0
  1  |  1.25                       |  0.35813099607523763        |  7.5
  2  |  -0.625                     |  -1.167653723395139         |  3.75
  3  |  0.3125                     |  0.22862988404582557        |  1.875
  4  |  -0.15625                   |  -0.18267475721398505       |  0.9375
  5  |  0.078125                   |  0.072253813532516          |  0.46875
  6  |  -0.0390625                 |  -0.04061857309907149       |  0.234375
  7  |  0.01953125                 |  0.019153481428506612       |  0.1171875
  8  |  -0.009765625               |  -0.009861459612460419      |  0.05859375
  9  |  0.0048828125               |  0.004859028755127261       |  0.029296875
 10  |  -0.00244140625             |  -0.0024473739963599586     |  0.0146484375
 11  |  0.001220703125             |  0.0012192139180053556      |  0.00732421875
 12  |  -0.0006103515625           |  -0.0006107242052398171     |  0.003662109375
 13  |  0.00030517578125           |  0.0003050826632019477      |  0.0018310546875
-------------------------------------------------------------------------------------------
```

Task 3

I modified the main.py function slightly to give to try a bunch of x0 guess every .1 from -3 to 7. This lets us see where the zeros may be for the function: `10.14*(((math.e)**x)**2)*math.cos(math.pi/x)`

```
----Fixed Point Iteration------------------------------------------------------------------
x0 |             x               |            f(x)            |      Error
-------------------------------------------------------------------------------------------
 -3.0  |  -4.823263367331518         |  0.0005213385273275942      |  0.0005218548962648839
 -2.9  |  -4.819404813342688         |  0.0005251683876057543      |  0.0005256923002985303
 -2.8  |  -4.815992562116736         |  0.0005285782285170161      |  0.0005291089030237117
 -2.7  |  -4.8129368632380976        |  0.0005316501767101608      |  0.0005321869795729128
 -2.6  |  -4.810153409204822         |  0.0005344636541851514      |  0.00053500610019519
 -2.5  |  -4.807557028040642         |  0.0005371011891758015      |  0.0005376489518473804
 -2.4  |  -4.805050362519581         |  0.0005396596843509862      |  0.0005402126287750164
 -2.3  |  -4.802498313308939         |  0.0005422767676354487      |  0.0005428350373453128
 -2.2  |  -4.799651751120293         |  0.0005452105391457947      |  0.0005457748084438663
 -2.1  |  -4.795790990217591         |  0.0005492144388976383      |  0.0005497869471948746
 -2.0  |  -2.0                       |  1.137210558948954e-17      |  0.0
 -1.9  |  -463590500.69965225        |  0.0                        |  0.0
 -1.8  |  -58066714154389.5          |  0.0                        |  0.0
 -1.7  |  -44458.78798664412         |  0.0                        |  0.0
 -1.6  |  -8716150.52421957          |  0.0                        |  0.0
 -1.5  |  -4.818996145769679         |  0.0005255756274271688      |  0.000526100345441094
 -1.4  |  -651884094710669.8         |  0.0                        |  0.0
 -1.3  |  -12.393557354325896        |  1.686653828831354e-10      |  1.6866508190105378e-10
 -1.2  |  -5.879794807812941e+16     |  0.0                        |  0.0
 -1.1  |  -6.191939977044987         |  3.7094689695999755e-05     |  3.709737934087798e-05
 -1.0  |  -351069981735.2846         |  0.0                        |  0.0
 -0.9  |  -1850.381810558566         |  0.0                        |  0.0
 -0.8  |  -5.036554493524679         |  0.00034732174374098665     |  0.0003475524863523205
 -0.7  |  -25630585.25787543         |  0.0                        |  0.0
 -0.6  |  -4.7964844942081175        |  0.0005484931069429897      |  0.0005490641265781804
 -0.5  |  -4.965965971718268         |  0.00039741793441794716     |  0.00039771940605426437
 -0.4  |  -0.4000000000000014        |  1.2684308810963777e-13     |  1.3877787807814457e-15
 -0.3  |  -434.26556217674977        |  0.0                        |  0.0
 -0.2  |  -4840962.262171073         |  0.0                        |  0.0
 -0.1  |  -8.402404603879065         |  4.7501909735038024e-07     |  4.750195454761297e-07
  0.1  |  -12.285023967953558        |  2.094323376922945e-10      |  2.0943247136528953e-10
  0.2  |  -204080942016140.44        |  0.0                        |  0.0
  0.3  |  -1848277054.3814836        |  0.0                        |  0.0
  0.4  |  0.39999999999999314        |  -3.039704234875971e-12     |  6.8833827526759706e-15
  0.5  |  -27.063377740574722        |  3.1343971574334113e-23     |  0.0
  0.6  |  -16.232992798274356        |  7.90779065591779e-14       |  7.815970093361102e-14
  0.7  |  -3460709575.9842196        |  0.0                        |  0.0
  0.8  |  -3.5152996909454065e+32    |  0.0                        |  0.0
  0.9  |  -7.178826350505909e+51     |  0.0                        |  0.0
  1.0  |  -8.980658876130352e+66     |  0.0                        |  0.0
  1.1  |  -1.694875801293176e+78     |  0.0                        |  0.0
  1.2  |  -1.3413141237199784e+86    |  0.0                        |  0.0
  1.3  |  -7.847477300712724e+90     |  0.0                        |  0.0
  1.4  |  -3.355046049205245e+92     |  0.0                        |  0.0
```

```
1.5 | -5.758700291137465e+90     | 0.0                        | 0.0
1.6 | -1.2077746376508013e+85    | 0.0                        | 0.0
1.7 | -5.062336012437051e+74     | 0.0                        | 0.0
1.8 | -3.487630573761014e+58     | 0.0                        | 0.0
1.9 | -1.4685828063421149e+35    | 0.0                        | 0.0
2.0 | 1.999999999999966          | -1.4717639989691995e-11    | 3.397282455352979e-14
2.1 | -48.43244420258345         | 8.654403426172689e-42      | 0.0
2.2 | -115.33950542091375        | 6.656644051593969e-100     | 0.0
2.3 | -202.94051772882537        | 5.421156027289644e-176     | 0.0
2.4 | -316.4949917545852         | 1.2646187632426574e-274    | 0.0
2.5 | -462.54258988491046        | 0.0                        | 0.0
2.6 | -649.1994317404241         | 0.0                        | 0.0
2.7 | -886.5232595113874         | 0.0                        | 0.0
2.8 | -1186.9629150152998        | 0.0                        | 0.0
2.9 | -1565.9111324660105        | 0.0                        | 0.0
3.0 | -2042.383983008167         | 0.0                        | 0.0
3.1 | -2639.8555998551437        | 0.0                        | 0.0
3.2 | -3387.2832856333603        | 0.0                        | 0.0
3.3 | -4320.366023243269         | 0.0                        | 0.0
3.4 | -5483.0891039364           | 0.0                        | 0.0
3.5 | -6929.6194492952045        | 0.0                        | 0.0
3.6 | -8726.630723771015         | 0.0                        | 0.0
3.7 | -10956.155106086515        | 0.0                        | 0.0
3.8 | -13719.080338397534        | 0.0                        | 0.0
3.9 | -17139.4372910392          | 0.0                        | 0.0
4.0 | -21369.65585568378         | 0.0                        | 0.0
4.1 | -26597.00684107631         | 0.0                        | 0.0
4.2 | -33051.496320033075        | 0.0                        | 0.0
4.3 | -41015.53855357531         | 0.0                        | 0.0
4.4 | -50835.80663081499         | 0.0                        | 0.0
4.5 | -62937.74928653665         | 0.0                        | 0.0
4.6 | -77843.37163016437         | 0.0                        | 0.0
4.7 | -96193.01118813943         | 0.0                        | 0.0
4.8 | -118772.00416553291        | 0.0                        | 0.0
4.9 | -146543.336821923          | 0.0                        | 0.0
5.0 | -180687.6214617335         | 0.0                        | 0.0
5.1 | -222652.0357007599         | 0.0                        | 0.0
5.2 | -274210.2295414301         | 0.0                        | 0.0
5.3 | -337535.6522268502         | 0.0                        | 0.0
5.4 | -415291.29801116383        | 0.0                        | 0.0
5.5 | -510739.53908548964        | 0.0                        | 0.0
5.6 | -627876.5320781792         | 0.0                        | 0.0
5.7 | -771596.6849862544         | 0.0                        | 0.0
5.8 | -947893.8946516517         | 0.0                        | 0.0
5.9 | -1164107.7605504785        | 0.0                        | 0.0
6.0 | -1429224.8093188582        | 0.0                        | 0.0
6.1 | -1754247.00017059          | 0.0                        | 0.0
6.2 | -2152642.5147124752        | 0.0                        | 0.0
6.3 | -2640897.1762998216        | 0.0                        | 0.0

6.4 | -3239188.9292601105        | 0.0                        | 0.0
6.5 | -3972212.8023694973        | 0.0                        | 0.0
6.6 | -4870189.88595144          | 0.0                        | 0.0
6.7 | -5970101.314831978         | 0.0                        | 0.0
6.8 | -7317197.371948089         | 0.0                        | 0.0
6.9 | -8966842.97852798          | 0.0                        | 0.0
7.0 | -10986774.46724677         | 0.0                        | 0.0
```

```
-------------------------------------------------------------------------------------------
```

My modified `main.py` file is called `Task3.py` and the code is here:

```
import fixedPointIteration as fp
import bisection as bi
import math
import sys
import inspect


def f(x):
    return 10.14*(((math.e)**x)**2)*math.cos(math.pi/x)


def g(x):
    return x - f(x)


tol = input("Enter a value for tol: ")
maxIter = input("Enter a value for maxIter: ")

x0_array = []
for i in range(-30, 71):
    if i != 0:
        x0_array.append(.1*i)

fixed = []
for x0 in x0_array:
    fixed += fp.fixedPointIter(inspect.getsource(g), x0, tol, maxIter, 0)


print("\nIf the full table is not requested, the final iteration will show as iteration 0\n")

print("----Fixed Point Iteration------------------------------------------------------------------")
print("x0 |             x               |            f(x)                 |      Error  ")
print("-------------------------------------------------------------------------------------------")
for i in range(0, int(len(fixed)/2)):
    print(f"{round(x0_array[i], 1):>5}  |  {fixed[i*2]:<25}  |  {f(fixed[i*2]):<25}  |  {fixed[i*2 + 1
print("-------------------------------------------------------------------------------------------")
```

I did have to modify `fixedPointIteration.py` slightly and that is here:

```
import math


def f(x):
    return 10.14*(((math.e)**x)**2)*math.cos(math.pi/x)


def g(x):
    return x - f(x)
    return x - f(x)


def fixedPointIter(g_expression, x0, tol, maxIter, v):
    values = []
    error = 0
    # g = lambda x: eval(g_expression)
    x0 = float(x0)
    tol = float(tol)
    maxIter = int(maxIter)
    v = int(v)


    for i in range(0, maxIter):
        x1 = g(x0)
        error = abs(x1 - x0)
        x0 = x1
        if v:
            values.append(x0)
            values.append(error)
        if error < tol:
            break
    if not v:
        values.append(x0)
        values.append(error)
    return values
```

Task 4

My bisection algorithm uses the same main.py function and is in the module `bisection.py` and is here:

```
import math


def bisection(f_expression, a, b, tol, maxIter, v):
    f = lambda x: eval(f_expression)
    values = []
    a = float(a)
    b = float(b)
    tol = float(tol)
    maxIter = int(maxIter)
    c = 1010101010101010101.0
    v = int(v)
    k = int(math.log(tol / (b-a)) / math.log(1/2)) + 1
    if k > maxIter:
        k = maxIter
    for i in range(0, k):
        c = .5 * (a + b)
        if v:
            values.append(c)
            values.append(b - a)
        if f(c) == 0:
            break
        if f(a) * f(c) < 0:
```

```
        if f(a) * f(c) < 0:
            b = c
        else:
            a = c
    if not v:
        values.append(c)
        values.append(b - a)
    return values
```

The output for the bisection when given the function `x*math.e**-x` `Command line: $ python main.py x*math.e**-x x+x*math.e**-x -10 5 1 .001 20 1`

```
----------Bisection------------------------------------------------------------------------
iter |             x              |            f(x)            |      Error
-------------------------------------------------------------------------------------------
  0  |  -2.5                      |  -30.456234901758677       |  15.0
  1  |  1.25                      |  0.35813099607523763       |  7.5
  2  |  -0.625                    |  -1.167653723395139        |  3.75
  3  |  0.3125                    |  0.22862988404582557       |  1.875
  4  |  -0.15625                  |  -0.18267475721398505      |  0.9375
  5  |  0.078125                  |  0.072253813532516         |  0.46875
  6  |  -0.0390625                |  -0.04061857309907149      |  0.234375
  7  |  0.01953125                |  0.019153481428506612      |  0.1171875
  8  |  -0.009765625              |  -0.009861459612460419     |  0.05859375
  9  |  0.0048828125              |  0.004859028755127261      |  0.029296875
 10  |  -0.00244140625            |  -0.002447373963599586     |  0.0146484375
 11  |  0.001220703125            |  0.0012192139180053556     |  0.00732421875
 12  |  -0.0006103515625          |  -0.0006107242052398171    |  0.003662109375
 13  |  0.00030517578125          |  0.0003050826632019477     |  0.0018310546875
-------------------------------------------------------------------------------------------
```

The output for the bisection when given the function `` is:

- I couldn't get the program to run by entering the function through the command line so I hard coded it.
- I also found that if I gave the bisection program the entire range from -3 to 7 that it diverge. I limited the range to -3 to 0 and got these results.

```
----------Bisection------------------------------------------------------------------------
iter |             x              |            f(x)            |      Error
-------------------------------------------------------------------------------------------
  0  |  -1.5                      |  -0.25242043662507013      |  3.0
  1  |  -2.25                     |  0.019560638026380514      |  1.5
  2  |  -1.875                    |  -0.024926896675217247     |  0.75
  3  |  -2.0625                   |  0.007798572994563874      |  0.375
  4  |  -1.96875                  |  -0.004928761022269477     |  0.1875
  5  |  -2.015625                 |  0.0021918345032279065     |  0.09375
  6  |  -1.9921875                |  -0.0011620446851059753    |  0.046875
  7  |  -2.00390625               |  0.0005642465794853744     |  0.0234375
  8  |  -1.998046875              |  -0.0002862862981275645    |  0.01171875
  9  |  -2.0009765625             |  0.00014209855842054175    |  0.005859375
 10  |  -1.99951171875            |  -7.130994530994772e-05    |  0.0029296875
 11  |  -2.000244140625           |  3.558974604408759e-05     |  0.00146484375
 12  |  -1.9998779296875          |  -1.7811172157203772e-05    |  0.000732421875
 13  |  -2.00006103515625         |  8.901510355545747e-06      |  0.0003662109375
 14  |  -1.999969482421875        |  -4.451773991102433e-06     |  0.00018310546875
---------------------------------------------------------------------------------------------
```

Task 5

My code is uploaded to github and the repository has been shared with you. Here is a link to my github: Kaden Taylor GitHub.com