Homework 3

SETUP
##

In [59]:
```
####SETUP####
import numpy as np
import matplotlib.pyplot as plt
def driver(F='Nan',A='Nan',B='Nan',tol='Nan'):
    # use routines
    f = lambda x: (x-1)*x**2
    a = 0.5
    b = 2

    if F =='Nan':
        F=f
    if A =='Nan':
        A=a
    if B =='Nan':
        B=b

    # f = lambda x: np.sin(x)
    # a = 0.1
    # b = np.pi+0.1
    if tol == 'Nan':

        tol = 1e-5
    [astar,ier,count] = bisection(F,A,B,tol)
    print('the approximate root is',astar)
    print('the error message reads:',ier)
    print("The number of iterations is:",count)
    print('f(astar) =', F(astar))
# define routines
def bisection(f,a,b,tol):
# Inputs:
# f,a,b - function and endpoints of initial interval
# tol - bisection stops when interval length < tol
# Returns:
# astar - approximation of root
# ier - error message
# - ier = 1 => Failed
# - ier = 0 == success
# first verify there is a root we can find in the interval
    fa = f(a)
    fb = f(b)
    count = 0
    if (fa*fb>0):
        print("same sign")
        ier = 1
        astar = a
        return [astar, ier,count]
```

```
    # verify end points are not a root
    if (fa == 0):
        print("a=0")
        astar = a
        ier =0
        return [astar, ier, count]
    if (fb ==0):
        print("b=0")
        astar = b
        ier = 0
        return [astar, ier, count]

    d = 0.5*(a+b)
    while (abs(d-a)> tol):
        fd = f(d)
        if (fd ==0):
            print("d=0")
            astar = d
            ier = 0

            return [astar, ier,count]
        if (fa*fd<0):
            b = d
        else:
            a = d
            fa = fd
        print('a =',a)
        print('b=',b)
        d = 0.5*(a+b)
        count = count +1
        # print('abs(d-a) = ', abs(d-a))
    print('not in loop')
    astar = d
    ier = 0
    return [astar, ier,count]
def fixedptbetter(g,p0,tol=1e-10,max_iter=100):
     approx = [p0]

     for i in range(max_iter):
          p1=g(p0)
          approx.append(p1)

          if abs(p1-p0)<tol:
               break
          p0=p1

     return approx,p0

def fixedpt(f,x0,tol,Nmax,iterative):

     ''' x0 = initial guess'''
     ''' Nmax = max number of iterations'''
     ''' tol = stopping tolerance'''
     xlist = [x0]
     count = 0
     while (count <Nmax):
```

```
        count = count +1
        x1 = f(x0)
        xlist.append(x1)
        if (abs(x1-x0) <tol):
            if iterative == True:
                return [xlist,count]
            xstar = x1
            ier = 0
            return [xstar,ier]
        x0 = x1
    if iterative == True:
        return xlist,count
    xstar = x1
    ier = 1
    return [xstar, ier]
```

##

Problem 1:

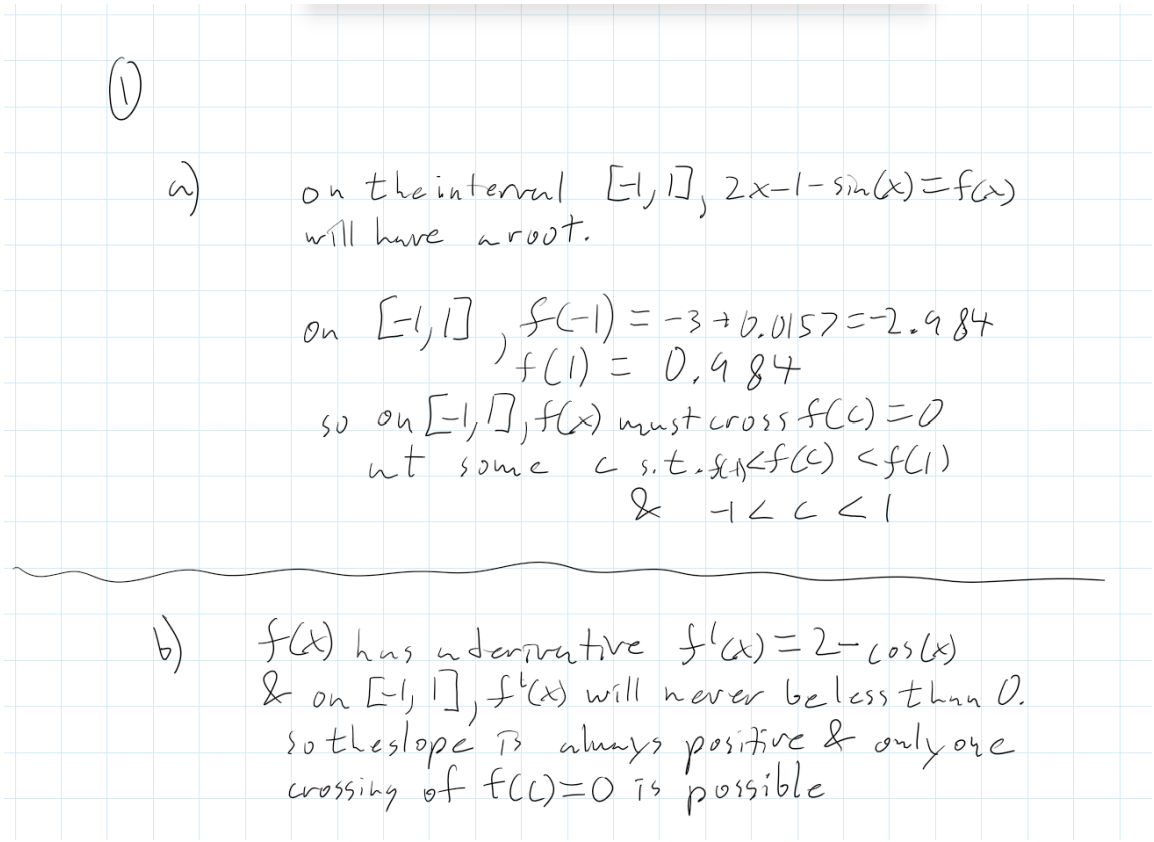
1. Consider the equation $2x - 1 = \sin x$.

   (a) Find a closed interval $[a, b]$ on which the equation has a root r, and use the Intermediate Value Theorem to prove that r exists.

   (b) Prove that r from (a) is the only root of the equation (on all of $\mathbb{R}$).

   (c) Use the bisestion code from class (or your own) to approximate r to eight correct decimal places. Include the calling script, the resulting final approximation, and the total number of iterations used.

a)-b)



The above is the analytical solutions for the first two parts of question 1.

c)

In [5]:
```
f= lambda x: 2*x-1-np.sin(x)
driver(f,-1,1)
```

```
the approximate root is 0.8878555297851562
the error message reads: 0
The number of iterations is: 17
f(astar) = -9.146867554155058e-06
```

##

Problem 2:

1. Consider the equation $2x - 1 = \sin x$.

   (a) Find a closed interval $[a, b]$ on which the equation has a root r, and use the Intermediate Value Theorem to prove that r exists.

   (b) Prove that r from (a) is the only root of the equation (on all of $\mathbb{R}$).

   (c) Use the bisestion code from class (or your own) to approximate r to eight correct decimal places. Include the calling script, the resulting final approximation, and the total number of iterations used.

a)

In [13]:
```
f21 = lambda x: (x-5)**9
```

```
driver(f21,4.82,5.2,1e-4)
```

```
the approximate root is 5.000073242187501
the error message reads: 0
The number of iterations is: 11
f(astar) = 6.065292655789404e-38
```

b)

In [18]:
```
f22 = lambda x: x**9-45*x**8+900*x**7-10500*x**6+78750*x**5-393750*x**4+1312500*x**
driver(f22,4.82,5.2,1e-4)
```

```
a = 5.01
b= 5.2
a = 5.105
b= 5.2
a = 5.105
b= 5.1525
the approximate root is 5.12875
the error message reads: 0
The number of iterations is: 3
f(astar) = 0.0
```

c)

This is an error where the boundaries a and b drifted away from the interval immediately. This likely happened in the line where $F(a)F(d) < 0$ is calculated. It is likely that $F(d) > 0$ for the expanded version due to the extra subtraction, but that it should not be, ie $F(d) < 0$ for the un-expanded version.

##

Problem 3:

3. (a) Use a theorem from class (Theorem 2.1 from text) to find an upper bound on the number of iterations in the bissection needed to approximate the solution of $x^3 + x - 4 = 0$ lying in the interval $[1, 4]$ with an accuracy of 10^{-3}.

   (b) Find an approximation of the root using the bisection code from class to this degree of accuracy. How does the number of iterations compare with the upper bound you found in part (a)?

a)

The maximum error in the bisection method will always be half the interval $[a, b]$.So if you want the max to be 10^{-3}, you need to satisfy $(a-b)/2^{n+1}$ or in this case $10^{-3} = 3/(2^{n+1})$ so $n = 12$

b)

In [19]:
```
f32=lambda x: x**3+x-4
driver(f32,1,4,1e-3)
```

```
a = 1
b= 2.5
a = 1
b= 1.75
a = 1.375
b= 1.75
a = 1.375
b= 1.5625
a = 1.375
b= 1.46875
a = 1.375
b= 1.421875
a = 1.375
b= 1.3984375
a = 1.375
b= 1.38671875
a = 1.375
b= 1.380859375
a = 1.3779296875
b= 1.380859375
a = 1.3779296875
b= 1.37939453125
the approximate root is 1.378662109375
the error message reads: 0
The number of iterations is: 11
f(astar) = -0.0009021193400258198
```

This is a good match to how many iterations I said we would need.

##

Problem 4:

4. **Definition 1** *Suppose* $\{p_n\}_{n=0}^{\infty}$ *is a sequence that converges to* p *with* $p_n \neq p$ *for all* n. *If there exists postive constants* λ *and* α *such that*

$$\lim_{n\to\infty} \frac{|p_{n+1} - p|}{|p_n - p|^{\alpha}} = \lambda$$

then $\{p_n\}_{n=1}^{\infty}$ *converges to* p *with an order* α *and asympotic error constant* λ. *If* $\alpha = 1$ *and* $\lambda < 1$ *then the sequence converges* linearly. *If* $\alpha = 2$, *the sequence is* quadratically *convergent.*

Which of the following iterations will converge to the indicated fixed point x_* (provided x_0 is sufficiently close to x_*)? If it does converge, give the order of convergence; for linear convergence, give the rate of linear convergence.

(a) (10 points) $x_{n+1} = -16 + 6x_n + \frac{12}{x_n}$, $x_* = 2$

(b) (10 points) $x_{n+1} = \frac{2}{3}x_n + \frac{1}{x_n^2}$, $x_* = 3^{1/3}$

(c) (10 points) $x_{n+1} = \frac{12}{1+x_n}$, $x_* = 3$

4

a) $x_{n+1} = 6x_n + \frac{12}{x_n} - 16$

$g(x) = 6x + \frac{12}{x} - 16 \quad g'(x) = 6 - \frac{12}{x^2}$, in $[x-\delta, x+\delta]$ $|g'(x)| \le 1$

$g''(x) = \frac{24}{x^3}$

so $\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|} = \lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|} = g'(x^*)$ → from lecture notes class 8

$g'(x^*) = 6 - 3 = 3$ so not linear as $|g'(x^*)| \not< 1$

$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^2} = \lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|^2} = \lim_{n\to\infty} \frac{g'(x^*)}{e_n} + \frac{1}{2} g''(x^*)$

Does not converge

---

b) $g(x) = \frac{2}{3}x + \frac{1}{x^2} \quad g'(x) = \frac{2}{3} - \frac{2}{x^3}$, $e_{n+1} = g'(x^*)e_n + \frac{1}{2}g''(x^*)e_n^2$

$g''(x) = \frac{6}{x^4}$ $|g'(x)| \le 1$ on $[3^{1/3} - \delta, 3^{1/3} + \delta]$

so $\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|} = g'(x^*) = 0$

so check $\alpha = 2$

$\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|^2} = \frac{1}{2} g''(x^*) = 3 \frac{1}{3^{4/3}} = 3^{-1/3} < 1$

so quadratically convergent

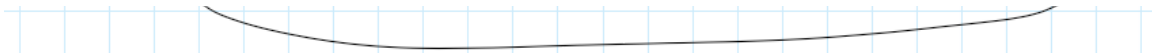
---

c) $g(x) = \frac{12}{1+x}$, $g'(x) = \frac{-12}{(1+x)^2}$, $g''(x) = \frac{24}{(1+x)^3}$

$|g'(x)| \le 1$ on $[3-\delta, 3+\delta]$

so $\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|} = g'(x^*) = -3/4$ so $|g'(x^*)| \le 1$ & $|g'(x^*)| \neq 0$

So linear convergence with rate 3/4.



##

Problem 5:

5. All the roots of the scalar equation

$$x - 4\sin(2x) - 3 = 0,$$

are to be determined with at least 10 accurate digits[1].

(a) Plot $f(x) = x - 4\sin(2x) - 3$ (using your `Python` toolbox). All the zero crossings should be in the plot. How many are there?

(b) Write a program or use the code from class to compute the roots using the fixed point iteration

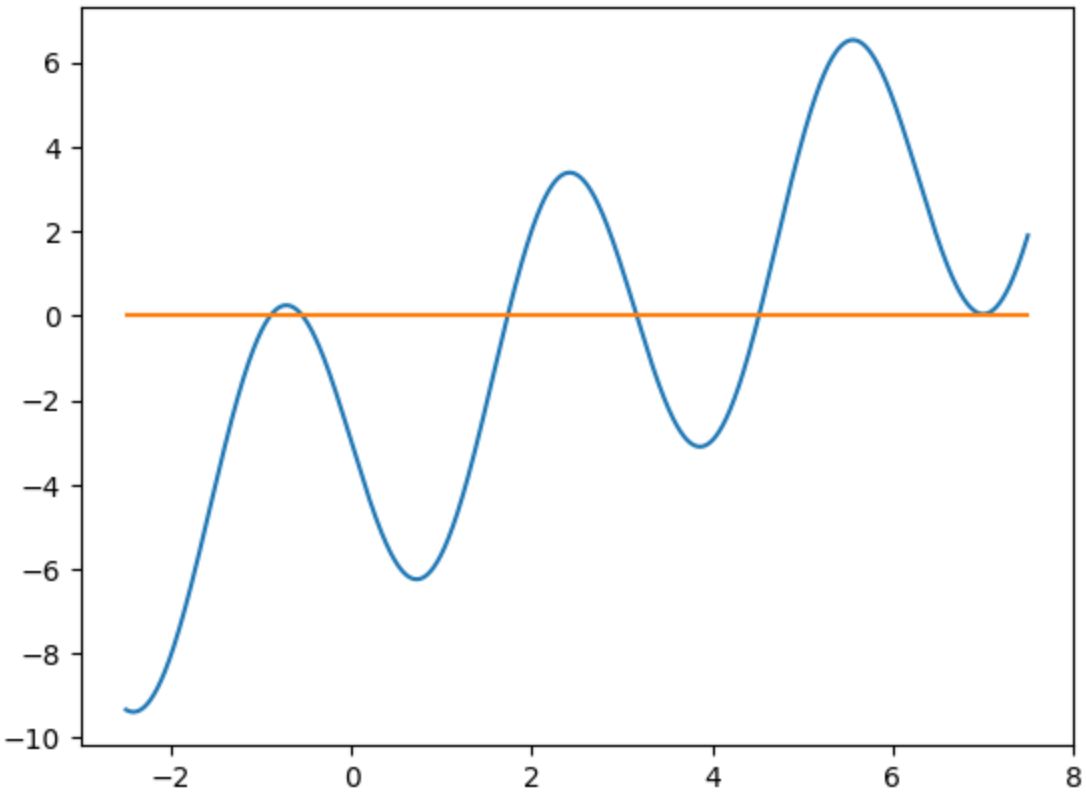
$$x_{n+1} = -\sin(2x_n) + 5x_n/4 - 3/4.$$

Use a stopping criterium that gives an answer with ten correct digits. (*Hint: you may have to change the error used in determing the stopping criterion.*) Find, empirically which of the roots that can be found with the above iteration. Give a theoretical explanation.

a)

In [71]:
```
f51 = lambda x: x-4*np.sin(2*x)-3
x=np.linspace(-2.5,7.5,1000)
plt.plot(x,f51(x))
y=np.zeros_like(x)
plt.plot(x,y)
```

Out[71]:
```
[<matplotlib.lines.Line2D at 0x2869f3c42d0>]
```



There are 6 crossing points.

b)

In [70]:
```
f52 = lambda x: -np.sin(2*x)+5*x/4-3/4
fixedptguesses = [-1,0,1,3,4,6]
for guess in fixedptguesses:

    p0,iter =fixedpt(f52,guess,1e-10,100,False)

    print('xstar=',p0)
    print('fstar =',f51(p0))
    print(iter)
```

```
xstar= -11691525728.540548
fstar = -11691525732.37378
1
xstar= -0.5444424007083105
fstar = 7.476996799482549e-11
0
xstar= -0.544442400663504
fstar = -4.6556536403841164e-11
0
xstar= 3.1618264865119454
fstar = 2.7974245142559084e-10
0
xstar= 3.1618264865177657
fstar = 2.390390108075735e-10
0
xstar= 13142416243.285658
fstar = 13142416242.218386
1
```

The above code makes it clear that for the points $P_0 = -0.54442400663504$ and $P_0 = 3.1618264865119454$ the iteration method works. For the other 4 points, no guess will result in the correct output as the slope at $F(x) = 0$ is too high, therefore the requirement $G'(x) \leq 1$ is not valid.

In [ ]: