```
# import libraries
import numpy as np

def driver():

# use routines
    f = lambda x: (x**9 - 45*x**8 + 900*x**7 - 10500*x**6 + 78750*x**5
            - 393750*x**4 + 1312500*x**3 - 2812500*x**2
            + 3515625*x - 1953125)
    a = 4.82
    b = 5.2

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

    tol = 1e-3
    print('')
    [astar,ier,count] = bisection(f,a,b,tol)
    print('the approximate root is',astar)
    print('the error message reads:',ier)
    print('f(astar) =', f(astar))

    print('Iterations:', count)

# 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);
    if (fa*fb>0):
       ier = 11
       astar = a
       return [astar, ier, 0]

#   verify end points are not a root
    if (fa == 0):
      astar = a
      ier =0
      return [astar, ier, 0]

    if (fb ==0):
      astar = b
```

```
      ier = 0
      return [astar, ier, 0]

    count = 0
    d = 0.5*(a+b)
    while (abs(d-a)> tol):
      fd = f(d)
      if (fd ==0):
         astar = d
         ier = 0
         return [astar, ier, count]
      if (fa*fd<0):
         b = d
      else:
         a = d
         fa = fd
      d = 0.5*(a+b)
      count = count +1
#      print('abs(d-a) = ', abs(d-a))

    astar = d
    ier = 0
    return [astar, ier, count]

driver()
```

Terminal print

```
the approximate root is 5.12875
the error message reads: 0
f(astar) = 0.0
Iterations: 3
```