

```

"""
Andy Lu
SMU Mathematics
MATH 5315 Fall 2017

Pseudocode for Homework 3
- This script is my implementation of a root finder using interpolating
  polynomials to do the work. A choice of using the Newton form of interpolating
  was made because it is easier to recompute the values of the coefficients
  rather than constantly rebuild polynomials of degree 0 up to n. Helper
  functions have been written to improve source code readability.
"""

quadratic_sol(INPUTS):

Usage: x, its = quadratic_sol( Inputs )
Inputs:
    Ffun      Nonlinear function name/handle
    x          Initial guess at solution
    maxit      max allowed number of iterations
    Srtol      relative solution tolerance
    Satol      absolute solution tolerance
    Rrtol      relative residual tolerance
    Ratol      absolute residual tolerance
    output     Boolean to output iteration history

Outputs:
    x          Approximate solution
    its        Number of iterations used

# Check EACH input arguments
if bad input:
    set to a default value
    output warning to the screen

# Initialize variables
x0 = x

# Set up the two other initial guesses
if x0 equal 0:
    x1 = x + 1e-2
    x2 = x - 1e-2
else:
    x1 = x(1 + 1e-2)
    x2 = x(1 - 1e-2)

LOOP: from 1 to maxit:
    # Call utility functions to build a quadratic interpolating function
    # clist will be [a, b, c] for  $ax^2 + bx + c$ 
    clist = newtwoncoeff(Ffun, x0, x1, x2)

    # Using a,b,c plug those into the quadratic question and return a root, and if
    # the root is imaginary
    root, imag = quad_equation(clist)

    # if root is imaginary, print message and quit
    if (imag):
        print error message and quit

```

```

# Shift guesses
x2 = root
x1 = x2
x0 = x1

# Check for convergence
if (exit_condition_true):
    break
END LOOP
END QUADRATIC_SOL

quad_equation()
Usage: root, imag = quad_equation(clist)
Inputs:
            clist            A list of coefficients in the order a,b,c for
                             ax^2 +bx + c

Outputs:
            root            Solution to the quadratic
            imag            Boolean describing if the root returned is imaginary
root = -(clist[1]) + sqrt(c[1]^2 - 4*c[0]*c[2]) / 2*c[0]
if c[1]^2 - 4*c[0]*c[2] < 0
    return root and true
else
    return root and false

newtoncoeff()
Usage: clist = newtoncoeff(Ffun, x0, x1, x2)
Inputs:
            Ffun            Function handle
            x0              A point to interpolate
            x1              A point to interpolate
            x2              A point to interpolate

Outputs:
            clist            A list of the coefficients that describe the
                             interpolating polynomial

# calculate divided differences
# store in an array
arr[i] = Ffun(xi) # for i = 0, 1, ...
arr[i] = arr[i] - arr[i-1] / xi - xi-1 # for i = 1, 2, ...
arr[i] = arr[i] - arr[i-1] / (xi - xi-1)(xi - xi-2) #for i = 2, 3, ...
# No need to do anything else because we don't need to evaluate
# our interpolant, just need to find one so we can find a root
# return the array
return arr

```