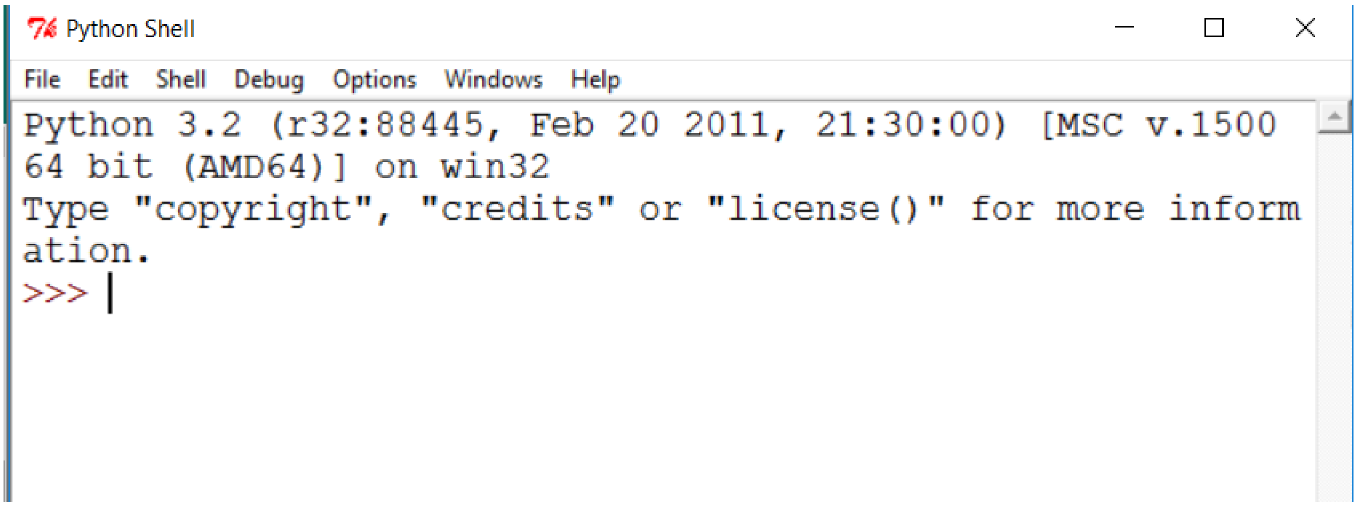
Turtles—Quadratic Equations

Name:	Class:	Date:

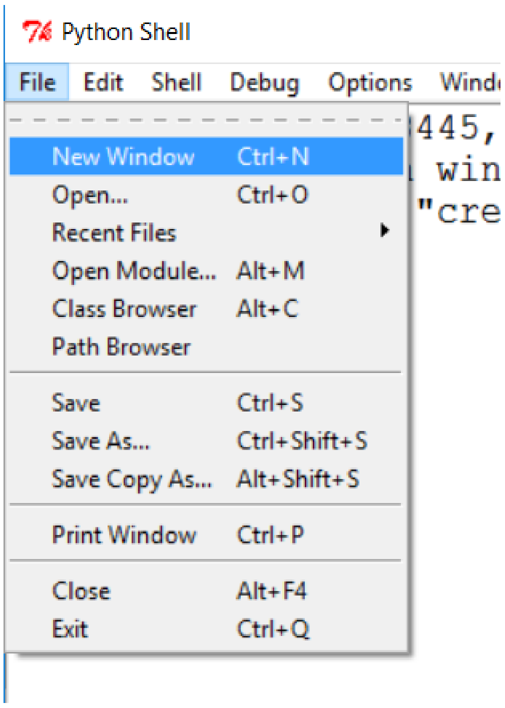
Create a new Python project as follows:

Open IDLE 3.x



```
Python 3.2 (r32:88445, Feb 20 2011, 21:30:00) [MSC v.1500
64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more inform
ation.
>>> |
```

Click file and create a New Window:



Call it QuadraticTurtle.py Great! Now we can start coding!

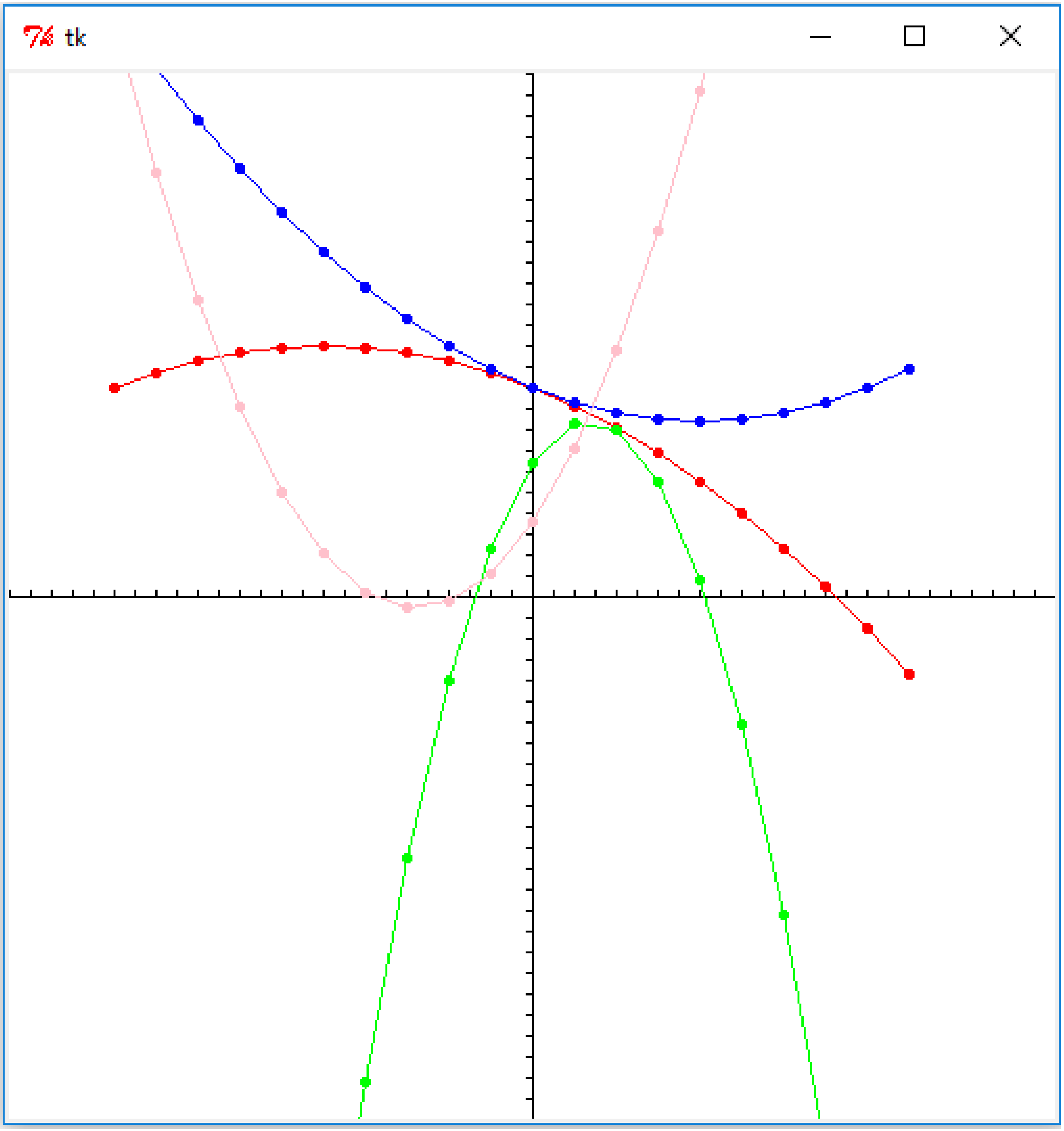
Create another file and call it Chart.py

Turtles— Quadratic Equations

This worksheet continues from Linear Turtles and you need to complete it first.

Have you ever seen how on games characters move in curves depending on a variety of factors such as speed, height, etc.

This is done using the mathematics of quadratic equations, which is basically the maths of a curve.



Turtles— Quadratic Equations

Let’s start by creating a GUI so that the Quadratic Turtle we create later can be instantiated.

Part of this will also be writing a function which will have a turtle which draws the graph axis.

```
from QuadraticTurtle import *
from tkinter import *
from tkinter import ttk

def draw_chart(canvas):

    graphy = turtle.RawTurtle(canvas)
    graphy.ht()
    graphy.speed(0)
    graphy.penup()
    graphy.goto(250, 0)
    graphy.pendown()
    for i in range(0, 50):
        graphy.back(10)
        graphy.left(90)
        graphy.forward(3)
        graphy.back(3)
        graphy.right(90)
    graphy.penup()
    graphy.goto(0, -250)
    graphy.pendown()
    graphy.left(90)
    for i in range(0, 50):
        graphy.forward(10)
        graphy.left(90)
        graphy.forward(3)
        graphy.back(3)
        graphy.right(90)

class GUI():

    root = Tk()

    #This is the canvas the turtle will exist in
    canvas = Canvas(root, width = 500, height = 500)
    canvas.pack()

    draw_chart(canvas)

    root.mainloop()

GUI()
```

Turtles— Quadratic Equations

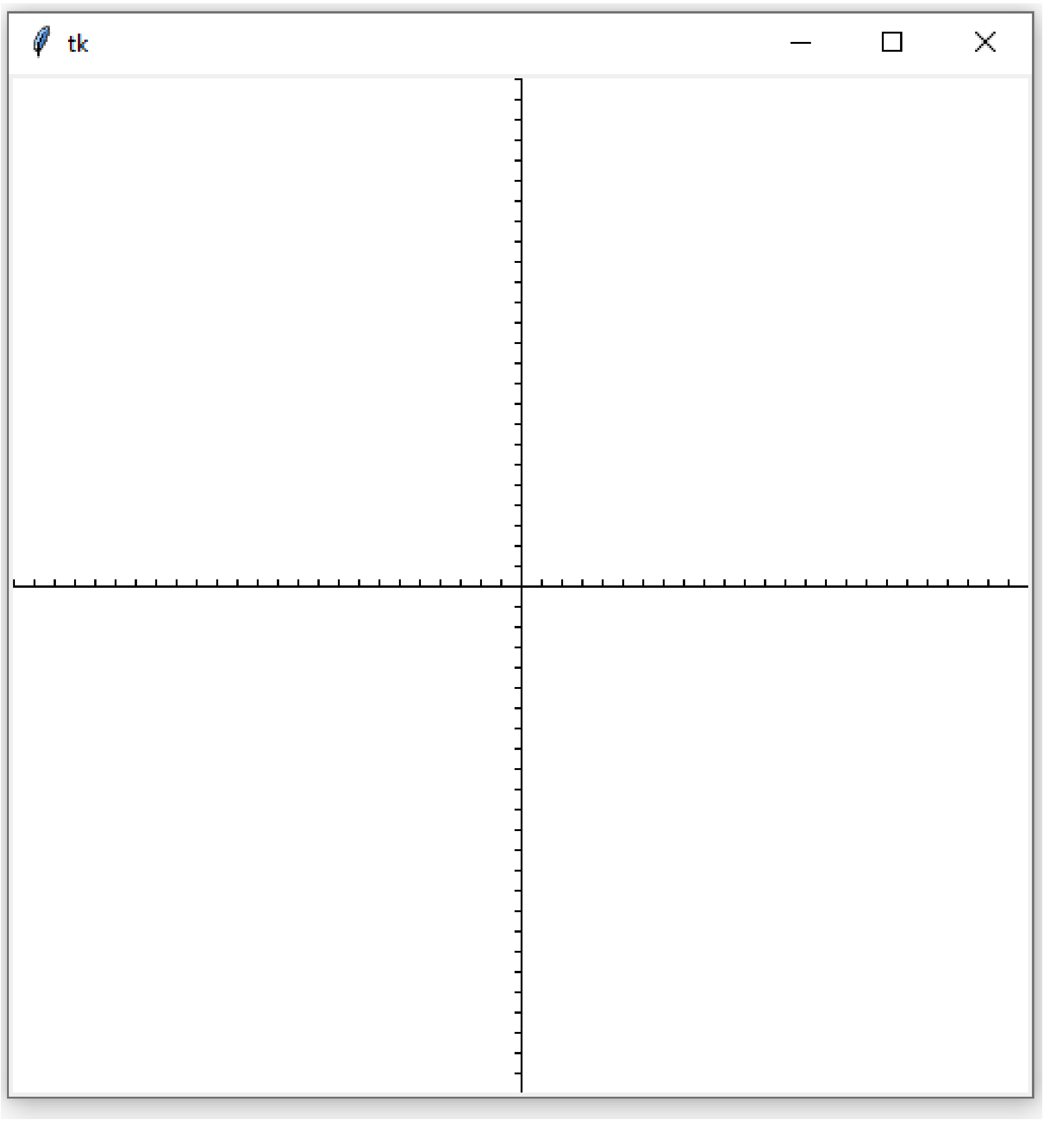
We now have a graph with 0,0 in the centre.

The turtle that did this has been hidden away.

Unlike with linear equations, turtles cannot just draw a curve between two points.

Simple curves can be done by doing short forward moves and minor degree adjustments, but this is not perfect if you need to work with a character that is moving at different speeds, or want to represent gravity.

The best option is to use quadratic equations to suggest the trajectory of different objects on screen.



Turtles— Quadratic Equations

It is now a good time to begin building QuadraticTurtle.

The design of QuadraticTurtle is similar to LinearTurtle as most of the issues met apply such as factors to widen the plot points etc.

It will even need to plot points to make this work.

Code QuadraticTurtle based upon the design below.

Class table: QuadraticTurtle—Inherits RawTurtle

Fields

Private factor **As** Integer *‘Sets a scale to increase the distance between points.*

Properties

Public Read Write p_factor() **As** Integer

Constructor

Public New (canvas **As** Canvas) *‘Use a canvas to hold the turtle.*

Methods

plot_points(x As Real, y As Real)

Pseudocode

```
‘Constructor for the  class.
‘Preconditions: Takes one parameter, a canvas to hold the turtle.
Public Constructor New (canvas As canvas)
    Super Constructor New(canvas As canvas)
End Constructor
Public Procedure plot_points(x  As Real, y As Real)
    setposition(x * factor, y * factor)
    dot()          ‘Instructs the turtle top place a dot. Not required, but useful for testing.
End Procedure
```

Turtles—Linear Equations

Now we need to work out how to create a quadratic function.

All quadratic equations are as follows:

$ax^2 + bx + c = y$ where y is always 0

In any quadratic equation we know the values of a, b, c, but not x.

In addition x can have two possible values when y is 0.

In fact coding a curve with a turtle is as simple as using this equation.

a: will define the distance between the two y intercepts. When a is negative we get an upwards curve. When it is positive we get a down wards curve.

b: moves the apex of the curve further positive or negative along x. At the same time moving the apex higher or lower.

c: Changes the height of the apex along the y axis.

Turtles—Linear Equations

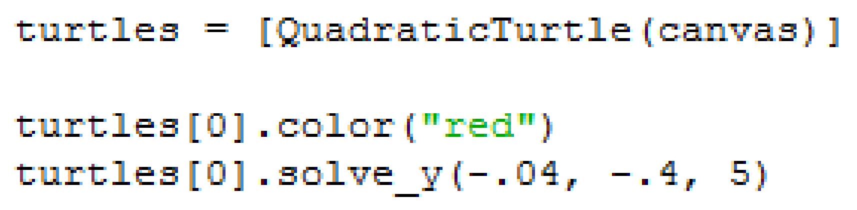
We will use a brute force method similar to that used in Linear turtle. Start with an x coordinate and end with an x coordinate.

```
Public Procedure solve_y(a As Real, b As Real, c As Real, x1 As Integer = -10, x2 As Integer=10)

    penup()
    first_point = True
    For x In range(x1, x2)
        If first_point Then   'Needed to stop the turtle drawing the first line to start point
            penup()
            first_point = False
        Else
            pendown()
        End If
        y = a * (x ^ 2)+(b * x) + c
        plot_points(x , y)
End Function
```

Code the solve_y() procedure now in Python.

You should now be able to call solve_y in the GUI:



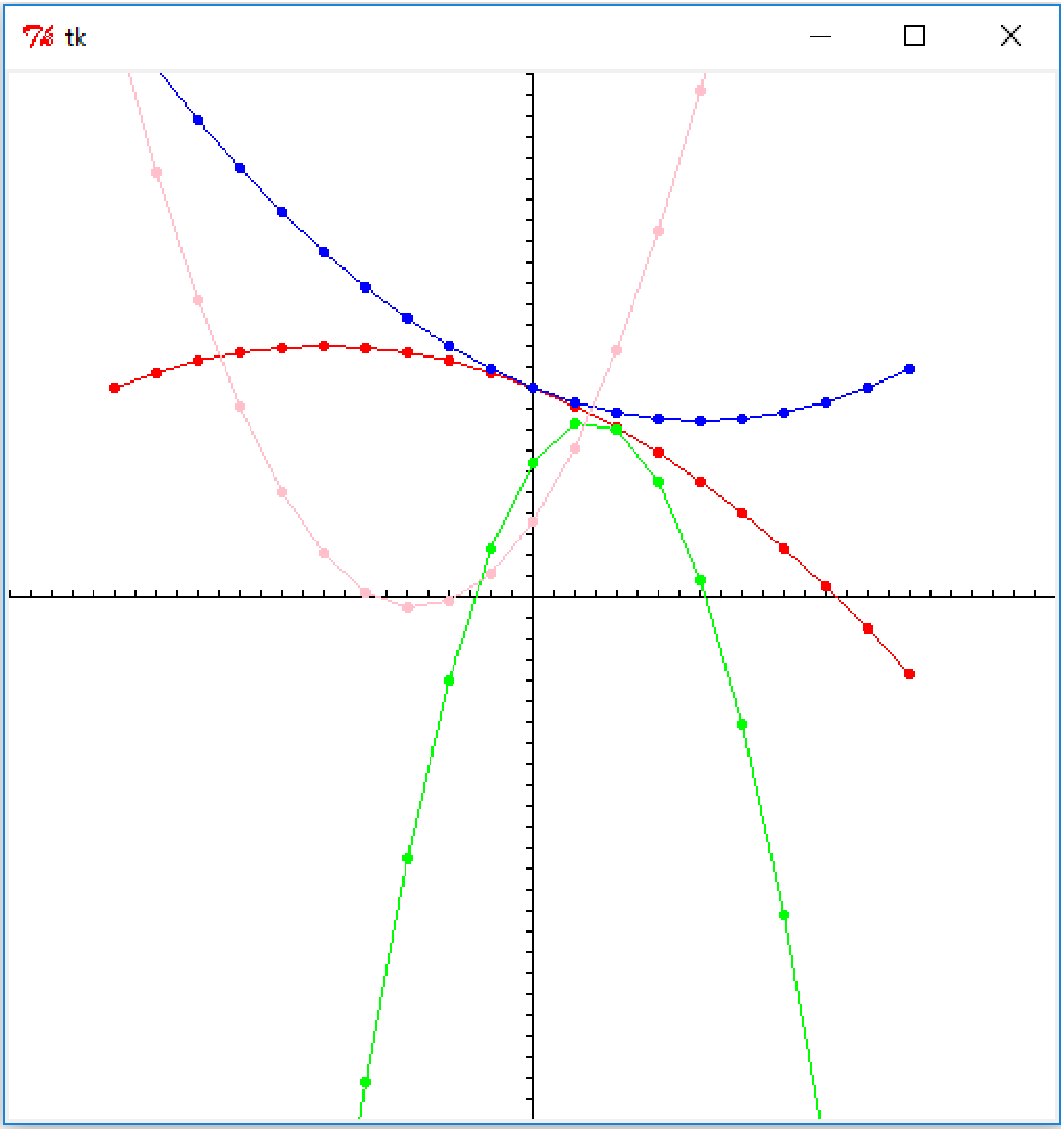
```
turtles = [QuadraticTurtle(canvas)]

turtles[0].color("red")
turtles[0].solve_y(-.04, -.4, 5)
```

Turtles— Quadratic Equations

One issue with using quadratic equations is working out start and end points for the curve.

After all an equation is only quadratic if it intercepts the x axis twice. This means that the blue line is not quadratic, as it will not intercept x.



Turtles— Quadratic Equations

Challenge 1

Create a few more curves by setting the values of a, b, c. You can also alter the start x and end x by adding the parameters.

This example set the star and end x as –15 and 15.

```
turtles[0].color("red")
turtles[0].solve_y(-.04, -.4, 5, -15, 15)
```

Can you identify which are quadratic?

Challenge 2

Create a new turtle PlottingTurtle that uses multiple inheritance to form a turtle that can call both quadratic and linear functions.

Ultra Mega Challenge!!!

Using y 0 and x -250 as the starting point move the turtle in a linear path along x.

Upon reaching x -100 it will then start a curve upwards before coming back down at x 60.

This is not easy as you need to work out the y intercept point for x -100 which also has a y intercept for x 60.

Otherwise you will get this:

Have fun brute forcing this!

Here's an idea though.

c moves up and down y.