Isaac Newton's method to approximate square roots works like this: given an approximation $x$ of $\sqrt{n}$, a better approximation is $\frac{x + \frac{n}{x}}{2}$. Repeat until you get the desired precision.

Write a subroutine `sqrt` implementing Newton's algorithm for approximating a square root. Start with an arbitrary guess for the approximation (1.0 is a good starting point), and loop until your approximation squared is within $10^{-5}$ of $n$, i.e. $|x^2 - n| < 0.00001$. (A better way is to test if the absolute value of the ratio $\left(\frac{|x^2 - n|}{n}\right)$ is close to zero – why?) You may adapt the `newton.asm` program from my Web site for this.

Write a MIPS program that prompts the user for the $(x, y)$ coordinates of two points in the real (Cartesian) plane, and then calculates and displays the distance between the points with reasonable descriptive text. Prompt for the coordinates using a little subroutine using simple linkage — do not duplicate the code for this.

Calculate the distance by using the Pythagorean Theorem:

$$d = \sqrt{\Delta x^2 + \Delta y^2}$$

Use all single-precision (32-bit float) arithmetic for this assignment. Use the simple register-based linkage convention, with nothing passed on the stack. However, because this is floating-point math, don't use $a$ registers for arguments and $v0$ for the return value. For this program, put any arguments in registers $f16$ through $f19$ and the return value(s) in $f0$ and $f1$ (and don't worry about using odd-numbered floating-point registers).