Sevika Singh
CSC210 Homework
6th September, 2012

1. *Read the original paper of Heron (see the link above). Carry out one more iteration of Heron's algorithm by hand (using paper and pencil, that is). Make sure to use fractions only.*
   (First Iteration from document.)
   1) $720 \div 27 = 26\ \frac{2}{3}$
   2) $26\frac{2}{3} + 27 = 53\frac{2}{3}$
   3) $53\frac{2}{3} \times \frac{1}{2} = 26\frac{5}{6}$

   Second Iteration carried out by hand.
   1) $720 \div 26\frac{5}{6} = 26\frac{134}{161}$
   2) $26\ \frac{134}{161} + 26\ \frac{5}{6} = 53\ \frac{643}{966}$
   3) $53\ \frac{643}{966} \times \frac{1}{2} = 26\ \frac{83281573}{100000000}$

   What is the error bound after this second iteration?
   There is no error bound. 26 $\frac{83281573}{100000000}$ squared equals 720, therefore that is the root.

   Write the precise steps in the Heron procedure to make it an algorithm that a computer can understand.

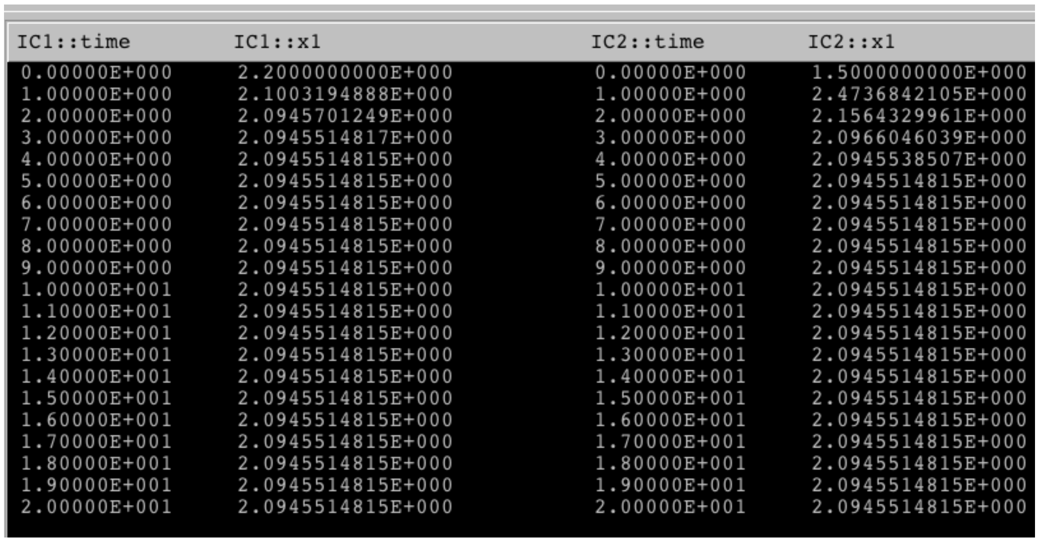
   1) Decide on a Y_0 that you want the square root of.
   2) Find the closest number to Y_0 that is a square number and make it Y_1.
   3) Take the square root of Y_1, make it x_o.
   4) Divide Y_0 by x_o. Add x_o to the result.
   5) Multiply that result by 1/2, and the current result is now x_1.
   6) Repeat steps 4-5, replacing x_n by x_{n+1} each time, until the smallest error bound is achieved.

   After some web search, write a paragraph about the life of Heron.
   Heron was an ancient Greek mathematician and engineer in Alexandria, Roman Egypt. He lived between the years of 10-70 AD. Among some of his achievements are items like the aeolipile ("Hero's Engine"), the first vending machine, one of the first types of wind-powered machines (organ), a force pump, syringe, and much more. Heron described a method of iteratively computing the square root, as well as Heron's formula, which is to find a triangles area from its side lengths.

2. *Using the modern version of Newton-Raphson method described in the lectures, find an approximation to a root of Newton's cubic polynomial y^3 - 2y -5 = 0 starting with the initial guesses y_0 = 2.2, and also y_0 = 1.5. How many iterations do you have to make to get Newton's finding 2.09455148? (See the Newton's original paper above.)*

   With the initial guess y_0 = 2.2, you have to make 3 iterations to get Newton's finding. With the initial guess y_0 = 1.5, you have to make 5 iterations to get Newton's finding.

   These iterations are seen below in the Phaser calculation.



```
IC1::time       IC1::x1                    IC2::time       IC2::x1
0.00000E+000    2.2000000000E+000          0.00000E+000    1.5000000000E+000
1.00000E+000    2.1003194888E+000          1.00000E+000    2.4736842105E+000
2.00000E+000    2.0945701249E+000          2.00000E+000    2.1564329961E+000
3.00000E+000    2.0945514817E+000          3.00000E+000    2.0966046039E+000
4.00000E+000    2.0945514815E+000          4.00000E+000    2.0945538507E+000
5.00000E+000    2.0945514815E+000          5.00000E+000    2.0945514815E+000
6.00000E+000    2.0945514815E+000          6.00000E+000    2.0945514815E+000
7.00000E+000    2.0945514815E+000          7.00000E+000    2.0945514815E+000
8.00000E+000    2.0945514815E+000          8.00000E+000    2.0945514815E+000
9.00000E+000    2.0945514815E+000          9.00000E+000    2.0945514815E+000
1.00000E+001    2.0945514815E+000          1.00000E+001    2.0945514815E+000
1.10000E+001    2.0945514815E+000          1.10000E+001    2.0945514815E+000
1.20000E+001    2.0945514815E+000          1.20000E+001    2.0945514815E+000
1.30000E+001    2.0945514815E+000          1.30000E+001    2.0945514815E+000
1.40000E+001    2.0945514815E+000          1.40000E+001    2.0945514815E+000
1.50000E+001    2.0945514815E+000          1.50000E+001    2.0945514815E+000
1.60000E+001    2.0945514815E+000          1.60000E+001    2.0945514815E+000
1.70000E+001    2.0945514815E+000          1.70000E+001    2.0945514815E+000
1.80000E+001    2.0945514815E+000          1.80000E+001    2.0945514815E+000
1.90000E+001    2.0945514815E+000          1.90000E+001    2.0945514815E+000
2.00000E+001    2.0945514815E+000          2.00000E+001    2.0945514815E+000
```

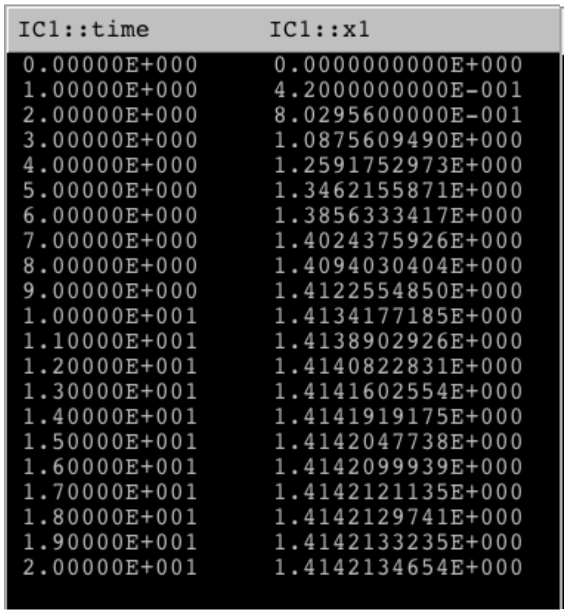
How good is this answer as an approximate root? To to this problem, you first need to determine which function to iterate; then enter this function into Phaser as a Custom Equation (MAP). Does Newton's polynomial have any other real root? (Hint: Using a function plotter, plot the graph of the cubic polynomial.)

This answer is as close to an approximate root as you will get, as no matter how many iterations you continue with, you will get the same result. There is no other real root.

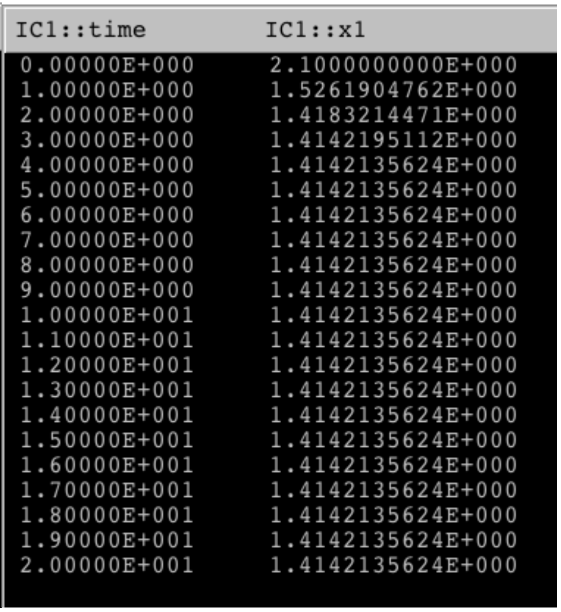
3. *Enter the difference equation x1 -> x1 - 0.21*(x1*x1 - 2) into PHASER. Determine the initial conditions whose iterates converge to sqrt(2). Compare the rate of convergence of this method of computing sqrt(2) with that of Newton's method. To make a fair comparison, use the same initial condition, say 2.1, in both iterations.*

The initial condition was 2.1, with 27 plots for the difference equation and only 5 necessary for Newton's method. The difference equation took approximately 26 iterations to get close enough, whereas Newton's method took 4 iterations.

The Phaser results below show the iterations of 2.1 for the difference equation on the left, and Newton's method on the right.



```
IC1::time        IC1::x1
0.00000E+000     0.0000000000E+000
1.00000E+000     4.2000000000E-001
2.00000E+000     8.0295600000E-001
3.00000E+000     1.0875609490E+000
4.00000E+000     1.2591752973E+000
5.00000E+000     1.3462155871E+000
6.00000E+000     1.3856333417E+000
7.00000E+000     1.4024375926E+000
8.00000E+000     1.4094030404E+000
9.00000E+000     1.4122554850E+000
1.00000E+001     1.4134177185E+000
1.10000E+001     1.4138902926E+000
1.20000E+001     1.4140822831E+000
1.30000E+001     1.4141602554E+000
1.40000E+001     1.4141919175E+000
1.50000E+001     1.4142047738E+000
1.60000E+001     1.4142099939E+000
1.70000E+001     1.4142121135E+000
1.80000E+001     1.4142129741E+000
1.90000E+001     1.4142133235E+000
2.00000E+001     1.4142134654E+000
```

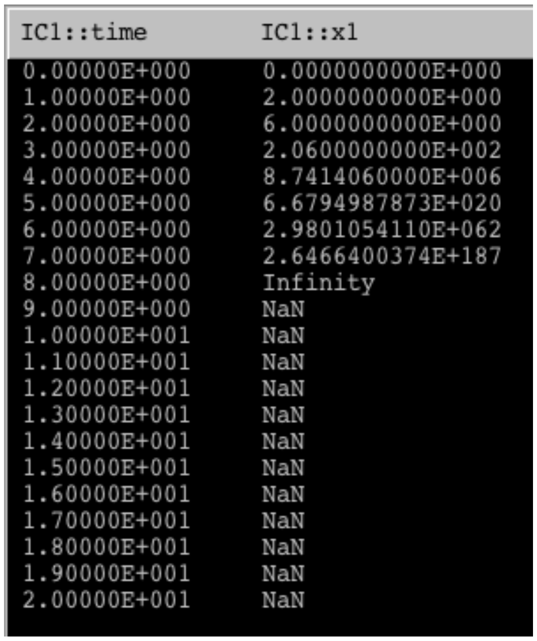


```
IC1::time        IC1::x1
0.00000E+000     2.1000000000E+000
1.00000E+000     1.5261904762E+000
2.00000E+000     1.4183214471E+000
3.00000E+000     1.4142195112E+000
4.00000E+000     1.4142135624E+000
5.00000E+000     1.4142135624E+000
6.00000E+000     1.4142135624E+000
7.00000E+000     1.4142135624E+000
8.00000E+000     1.4142135624E+000
9.00000E+000     1.4142135624E+000
1.00000E+001     1.4142135624E+000
1.10000E+001     1.4142135624E+000
1.20000E+001     1.4142135624E+000
1.30000E+001     1.4142135624E+000
1.40000E+001     1.4142135624E+000
1.50000E+001     1.4142135624E+000
1.60000E+001     1.4142135624E+000
1.70000E+001     1.4142135624E+000
1.80000E+001     1.4142135624E+000
1.90000E+001     1.4142135624E+000
2.00000E+001     1.4142135624E+000
```

4. *Using Newtons's method, with starting value x = 0, try to find a root of the function f(x) = x^3 - 2x + 2. Does the iteration converge?*

   The root of the function is 2. The iteration does not converge, as the function is infinite.

   This is seen here in the iteration values.



```
IC1::time          IC1::x1
0.00000E+000       0.0000000000E+000
1.00000E+000       2.0000000000E+000
2.00000E+000       6.0000000000E+000
3.00000E+000       2.0600000000E+002
4.00000E+000       8.7414060000E+006
5.00000E+000       6.6794987873E+020
6.00000E+000       2.9801054110E+062
7.00000E+000       2.6466400374E+187
8.00000E+000       Infinity
9.00000E+000       NaN
1.00000E+001       NaN
1.10000E+001       NaN
1.20000E+001       NaN
1.30000E+001       NaN
1.40000E+001       NaN
1.50000E+001       NaN
1.60000E+001       NaN
1.70000E+001       NaN
1.80000E+001       NaN
1.90000E+001       NaN
2.00000E+001       NaN
```

5. *Compare two of the genomic sequences we examined in class, with accession numbers 'AF176722' and 'AF315498'. These are the mitochondrial D-loop sequences of the Chimp Schweinfurthii and Chimp Vellerosus respectively. Please report the number of substitutions and insertion/deletions (indels) to transform one sequence to the other. These can be found by looking at the alignment on the bottom of the result page from blast2seq.*

   Substitutions: 29
   Insertions/Deletions: 2

   What is the ratio of substitutions/indels?

   29 substitutions to 2 indels, or 8.61% substitutions to 0.59% indels.

   Which one do you think is more probable to occur by chance in a genomic sequence with time, a substitution or an indel? Why?

   I think a substitution is more likely to occur. For one, this example clearly displays a much greater amount of substitutions. Another reason this is more likely is because rather than genes being added or removed completed, it is more likely that they are just altered.