

Fibonacci Numbers and the Golden Ratio

James Emery

4/30/2011

Contents

1	Fibonacci Numbers	2
2	Some Large Fibonacci Numbers	12
3	The Binet Formula	12
4	The Golden Section Search	15
5	Phyllotaxis, the Golden Rectangle, Numerology, and Pseudoscience	18
6	The Pentagon and the Golden Ratio	19
7	Appendix: How the Figures in This Document Were Drawn	23
8	Appendix: Continued Fractions	42
8.1	The Continued Fraction Representation of a Rational Number	42
8.2	The Continued Fraction Representation of a Real Number . . .	46
8.3	Computing the Convergents	48
8.4	Properties of the Convergents	52
8.5	Continued Fractions Bibliography	55
9	Notes	56
10	Bibliography	56

1 Fibonacci Numbers

The Fibonacci numbers are defined to be an infinite sequence of numbers f_0, f_1, f_2, \dots . The first two numbers are $f_0 = 0$, and $f_1 = 1$. The third number is the sum of the previous two, so is $f_2 = 1$. The k th number in the sequence is the sum of the previous two numbers. So

$$f_k = f_{k-2} + f_{k-1}.$$

The first few numbers in the sequence are

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \dots$$

A second sequence of ratios of Fibonacci numbers,

$$\{r_n\}_1^\infty,$$

is also of interest. These ratios are defined by

$$r_n = \frac{f_{n+1}}{f_n}.$$

It turns out that this second sequence converges to the Golden Ratio. So let us define the Golden Ratio. Suppose a rectangle has height h and width w . Suppose the ratio of h to w is the same as the ratio of w to the sum of the sides $h + w$. Then

$$\frac{h}{w} = \frac{w}{h + w}.$$

Without loss of generality suppose $h = 1$, then

$$\frac{1}{w} = \frac{w}{1 + w},$$

or

$$w^2 = 1 + w.$$

$$w^2 - w - 1 = 0.$$

Thus

$$w = \frac{1 + \sqrt{5}}{2}.$$

A rectangle with these proportions is called the golden rectangle and

$$\phi = \frac{w}{h} = \frac{1 + \sqrt{5}}{2},$$

is called the golden ratio. This is approximately

$$\phi = 1.618033988749895$$

As a second way of defining ϕ , let us consider the division of a unit interval. Let the unit interval be divided into two intervals, one of length x and the other of length $1 - x$. Let us assume that

$$x \geq x - 1.$$

The extreme ratio is defined as

$$r_e = \frac{1}{x}$$

The mean ratio is defined as

$$r_m = \frac{x}{1 - x}.$$

If $r = r_e = r_m$, the interval is said to be divided in extreme and mean ratio. This is a concept that appears in Euclid's Elements.

Definition Euclid's Elements, Book 6, Definition 2: *A straight line said to have been cut in extreme and mean ratio when, as the whole line is to the greater segment, so is the greater to the lesser.*

In this case

$$r = \frac{1}{x} = \frac{x}{1 - x}.$$

So

$$x^2 + x - 1 = 0.$$

And so

$$x = \frac{-1 + \sqrt{5}}{2}.$$

Thus

$$r = \frac{1}{x} = \frac{1 + \sqrt{5}}{2},$$

which again is the golden ratio ϕ .

Here is another way of constructing a golden rectangle. Draw a square of side 1. Bisect the lower horizontal edge and take it as a circle center c . Let the radius of the circle be the distance from the center c to the upper left corner of the square. Then

$$r = \sqrt{(1/2)^2 + 1^2} = \frac{\sqrt{5}}{2}$$

Swing the circle down from the upper left corner of the square to the line through the base of the square. Let this intersection point define the lower left edge of a rectangle. The length of this edge is

$$\frac{1}{2} + r = \frac{1 + \sqrt{5}}{2}.$$

This again defines the golden rectangle. See figure 1.

We claim that the sequence of ratios of Fibonacci numbers converges to the golden ratio ϕ . We have

$$r_n = \frac{f_{n+1}}{f_n} = \frac{f_{n-1} + f_n}{f_n} = \frac{f_{n-1}}{f_n} + 1 < 2,$$

for $n > 3$. So the sequence of ratios is bounded. Note also that $r_n < 2r_{n-1}$. So suppose r_n converges to x . Then

$$d_n = r_n - r_{n-1},$$

converges to zero. Then

$$\begin{aligned} d_n = r_n - r_{n-1} &= \frac{f_{n+1}}{f_n} - \frac{f_n}{f_{n-1}} \\ &= \frac{f_{n+1}f_{n-1} - f_n^2}{f_n f_{n-1}} \\ &= \frac{(f_n + f_{n-1})f_{n-1} - f_n^2}{f_n f_{n-1}} \\ &= \frac{f_{n-1}^2 + f_n f_{n-1} - f_n^2}{f_n f_{n-1}} \end{aligned}$$

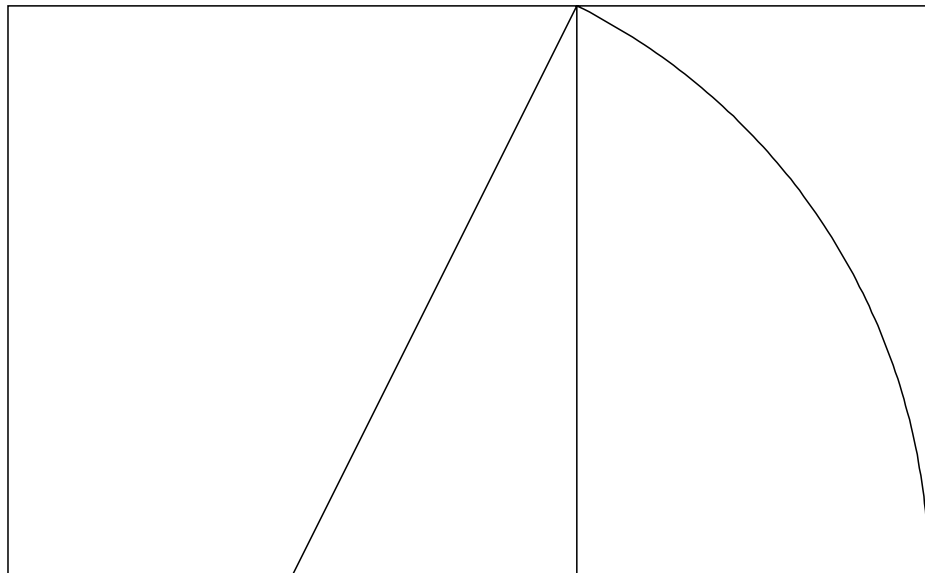


Figure 1: A construction of the Golden Rectangle. Draw a square with side equal to the height of the Golden Rectangle. From the midpoint of the base of the square draw an arc as shown to define the width of the Golden Rectangle.

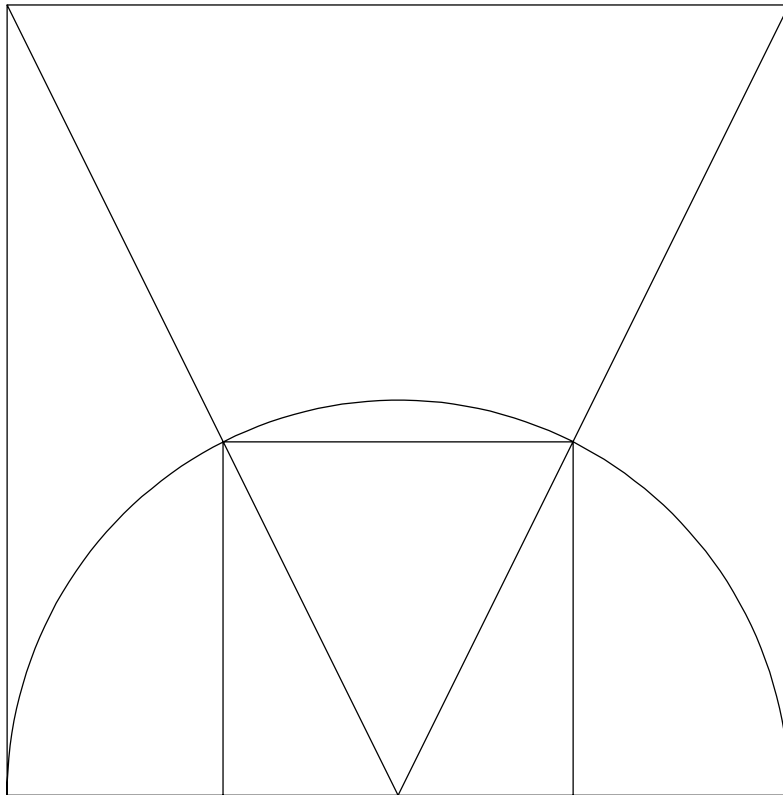


Figure 2: To inscribe a square in a semicircle, draw a large square using the diameter as base. Then draw diagonals from the semicircle center to the upper vertices of the square. The intersection points of the diagonals with the semicircle define the inscribed square. This follows by similarity of triangles. The Golden Rectangle is related to this inscribed square as shown in the next figure.

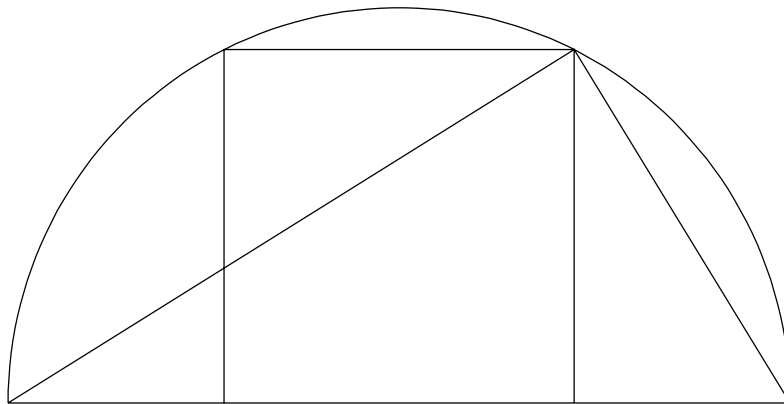


Figure 3: The Golden Rectangle may be constructed by using an inscribed square in a semicircle. Draw the inscribed square. Complete the figure as shown by adding two diagonal lines. Let b be the length of a side of the square. The diameter of the semicircle consists of three segments, two of length a and one of length b . There are two similar triangles with common vertical side of length b . Because these triangles are similar, the ratio of $a + b$ to b equals the ratio of b to a . So the rectangle with base $a + b$ and height b is the Golden Rectangle. The midpoint of the lower edge of the square is the center of the semicircle. So given a square, drawing an arc, with the midpoint as center, from the upper right vertex of the square to the baseline locates the lower left vertex of the golden rectangle.

$$\begin{aligned}
&= \frac{f_{n-1}}{f_n} + 1 - \frac{f_n}{f_{n-1}} \\
&= \frac{1}{r_{n-1}} + 1 - r_{n-1}.
\end{aligned}$$

Taking limits on both sides we get

$$0 = \frac{1}{x} + 1 - x.$$

This gives the equation for the golden ratio

$$x^2 - x - 1 = 0$$

So $x = \phi$. So if r_n converges, then it converges to ϕ . We shall show that r_n is the n th convergent of the continued fraction

$$[1, 1, 1, 1, 1, 1, \dots].$$

Further it turns out that all infinite simple continued fractions converge. For a proof of this see my document called **Continued fractions**.

<http://stem2.org/je/continuedfractions.pdf>

So the Fibonacci ratios converge to ϕ .

The Binet formula for the n th Fibonacci number is

$$f_n = \frac{1}{\sqrt{5}} \left[\left(\frac{1 + \sqrt{5}}{2} \right)^n - \left(\frac{1 - \sqrt{5}}{2} \right)^n \right].$$

(See Huntley, p148. His proof of convergence is a bit obscure). See my proof of the Binet formula below. Here is the output of a program named **fib.ftn**. The second column is the Fibonacci number, the third is the Binet formula value, and the fourth is the ratios of succeeding numbers:

1	1	1	
2	1	1	1.0000000000000000
3	2	2	2.0000000000000000
4	3	3	1.5000000000000000
5	5	5	1.6666666666666667
6	8	8	1.6000000000000000
7	13	13	1.6250000000000000
8	21	21	1.615384615384615

9	34	34	1.619047619047619
10	55	55	1.617647058823529
11	89	89	1.618181818181818
12	144	144	1.617977528089888
13	233	233	1.618055555555556
14	377	377	1.618025751072961
15	610	610	1.618037135278515
16	987	987	1.618032786885246
17	1597	1597	1.618034447821682
18	2584	2584	1.618033813400125
19	4181	4181	1.618034055727554
20	6765	6765	1.618033963166706
21	10946	10946	1.618033998521803
22	17711	17711	1.618033985017358
23	28657	28657	1.618033990175597
24	46368	46368	1.618033988205325
25	75025	75025	1.618033988957902
26	121393	121393	1.618033988670443
27	196418	196418	1.618033988780243
28	317811	317811	1.618033988738303
29	514229	514229	1.618033988754323
30	832040	832040	1.618033988748204
31	1346269	1346269	1.618033988750541
32	2178309	2178309	1.618033988749648
33	3524578	3524578	1.618033988749989
34	5702887	5702887	1.618033988749859
35	9227465	9227465	1.618033988749909
36	14930352	14930352	1.618033988749890
37	24157817	24157817	1.618033988749897
38	39088169	39088169	1.618033988749894
39	63245986	63245986	1.618033988749895
40	102334155	102334155	1.618033988749895
41	165580141	165580141	1.618033988749895
42	267914296	267914296	1.618033988749895
43	433494437	433494437	1.618033988749895
44	701408733	701408733	1.618033988749895
45	1134903170	1134903170	1.618033988749895
46	1836311903	1836311903	1.618033988749895

The continued fraction

$$1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \dots}}}}}$$

has convergents

$$r_n = \frac{f_{n+1}}{f_n}.$$

This continued fraction has the quotients

$$[1, 1, 1, 1, 1, 1, \dots] = [a_1, a_2, a_3, \dots].$$

The convergents of the continued fraction are given by the ratios p_i/q_i , where the p_i, q_i are defined iteratively by (see my **Continued fractions**, or Olds page 21)

$$p_0 = 1, p_{-1} = 0$$

$$q_0 = 0, q_{-1} = 1$$

and

$$p_i = a_i p_{i-1} + p_{i-2}$$

$$q_i = a_i q_{i-1} + q_{i-2}.$$

Because each $a_i = 1$ here, then the p_i and the q_i are successive Fibonacci numbers, and

$$r_n = \frac{p_n}{q_n}.$$

As stated above every simple continued fraction, like the one above, converges. Again see my **Continued fractions** or the book by C. D. Olds, **Continued Fractions**, p67. Therefore the sequence r_n converges. Also see my continued fraction programs: **cf.java**, **cfrx.java**, and **convergents.java**.

Count	Fibonacci	Binet	Ratio
1	1	1	
2	1	1	1.0000000000000000
3	2	2	2.0000000000000000
4	3	3	1.5000000000000000
5	5	5	1.6666666666666667
6	8	8	1.6000000000000000
7	13	13	1.6250000000000000
8	21	21	1.615384615384615
9	34	34	1.619047619047619
10	55	55	1.617647058823529
11	89	89	1.618181818181818
12	144	144	1.617977528089888
13	233	233	1.6180555555555556
14	377	377	1.618025751072961
15	610	610	1.618037135278515
16	987	987	1.618032786885246
17	1597	1597	1.618034447821682
18	2584	2584	1.618033813400125
19	4181	4181	1.618034055727554
20	6765	6765	1.618033963166706
21	10946	10946	1.618033998521803
22	17711	17711	1.618033985017358
23	28657	28657	1.618033990175597
24	46368	46368	1.618033988205325
25	75025	75025	1.618033988957902
26	121393	121393	1.618033988670443
27	196418	196418	1.618033988780243
28	317811	317811	1.618033988738303
29	514229	514229	1.618033988754323
30	832040	832040	1.618033988748204
31	1346269	1346269	1.618033988750541
32	2178309	2178309	1.618033988749648
33	3524578	3524578	1.618033988749989
34	5702887	5702887	1.618033988749859
35	9227465	9227465	1.618033988749909
36	14930352	14930352	1.618033988749890
37	24157817	24157817	1.618033988749897
38	39088169	39088169	1.618033988749894
39	63245986	63245986	1.618033988749895
40	102334155	102334155	1.618033988749895
41	165580141	165580141	1.618033988749895
42	267914296	267914296	1.618033988749895
43	433494437	433494437	1.618033988749895
44	701408733	701408733	1.618033988749895
45	1134903170	1134903170	1.618033988749895
46	1836311903	1836311903	1.618033988749895

2 Some Large Fibonacci Numbers

Here is a simple Python program to display a range of Fibonacci numbers. This program called `fibonacci.py` displays the 390th Fibonacci number through the 400th number.

```
# A list of Fibonacci numbers from m to n
m=390
n=400
a = 0L
b = 1L
i=1
for i in range(1,n+1):
    if (i > m-1):
        print i,b,"\n"
        c=b
        b=a+b
        a=c
        i=i+1
```

Python has the ability to compute with arbitrarily long integers, an ability that most languages do not have. An integer is treated as a long integer by adding "L" to the end of the number. By typing

```
python fibonacci.py
```

we run the program and output the 390th through the 400th Fibonacci number.

```
390 1431181439314368982806567444246559718305231073036073662367607266895781713447936520
391 2315700212878644019141884587055058551781936851295739770295042651311250305217930509
392 3746881652193013001948452031301618270087167924331813432662649918207032018665867029
393 6062581865071657021090336618356676821869104775627553202957692569518282323883797538
394 9809463517264670023038788649658295091956272699959366635620342487725314342549664567
395 15872045382336327044129125268014971913825377475586919838578035057243596666433462105
396 25681508899600997067167913917673267005781650175546286474198377544968911008983126672
397 41553554281937324111297039185688238919607027651133206312776412602212507675416588777
398 67235063181538321178464953103361505925388677826679492786974790147181418684399715449
399 108788617463475645289761992289049744844995705477812699099751202749393926359816304226
400 176023680645013966468226945392411250770384383304492191886725992896575345044216019675
```

3 The Binet Formula

Recall that the Golden Ratio is the number

$$\phi = \frac{1 + \sqrt{5}}{2},$$

and is a root of the equation

$$w^2 - w - 1 = 0.$$

Let the other root be

$$\psi = \frac{1 - \sqrt{5}}{2}.$$

Then necessarily,

$$\phi^2 = \phi + 1,$$

and

$$\psi^2 = \psi + 1.$$

Approximate values for these numbers are

$$\phi = 1.618033988749895,$$

and

$$\psi = -\phi + 1 = -.618033988749895.$$

Binet's Formula for the n th Fibonacci number is

$$\text{Fib}(n) = \frac{1}{\sqrt{5}}(\phi^n - \psi^n).$$

The problem of proving this formula is Exercise 1.14 on page 43 of Abelson and Sussman.

Proposition. The n th Fibonacci number is given by the Binet formula

$$\frac{1}{\sqrt{5}}(\phi^n - \psi^n),$$

where

$$\phi = \frac{1 + \sqrt{5}}{2},$$

and

$$\psi = \frac{1 - \sqrt{5}}{2}.$$

Proof. We shall prove this by induction. We have

$$\text{Fib}(0) = \frac{1}{\sqrt{5}}(\phi^0 - \psi^0) = 0.$$

And

$$\text{Fib}(1) = \frac{1}{\sqrt{5}}(\phi - \psi) = 1.$$

Given integer $n > 1$, suppose the formula holds for all non-negative integers less than n . We have then

$$\begin{aligned} (\phi^n - \psi^n) &= \phi^2\phi^{n-2} - \psi^2\psi^{n-2} \\ &= (\phi + 1)\phi^{n-2} - (\psi + 1)\psi^{n-2} \\ &= (\phi^{n-1} + \phi^{n-2}) - (\psi^{n-1} + \psi^{n-2}) \\ &= (\phi^{n-1} - \psi^{n-1}) + (\phi^{n-2} - \psi^{n-2}). \end{aligned}$$

Dividing by $\sqrt{5}$ we find that

$$\frac{1}{\sqrt{5}}(\phi^n - \psi^n) = \text{Fib}(n-1) + \text{Fib}(n-2) = \text{Fib}(n),$$

which proves the formula for all nonnegative integers.

Notice that

$$\left| \text{Fib}(n) - \frac{1}{\sqrt{5}}\phi^n \right| = \left| \frac{1}{\sqrt{5}}(\phi^n - \psi^n) - \frac{1}{\sqrt{5}}\phi^n \right| = \frac{1}{\sqrt{5}}|\psi|^n$$

is a decreasing sequence, and goes to zero as n goes to ∞ . We have

$$\frac{1}{\sqrt{5}}\psi^0 = .44721$$

$$\frac{1}{\sqrt{5}}\psi^1 = -.27639$$

$$\frac{1}{\sqrt{5}}\psi^2 = .17082$$

$$\frac{1}{\sqrt{5}}\psi^3 = -.10557$$

Therefore the difference

$$\left| \text{Fib}(n) - \frac{1}{\sqrt{5}}\phi^n \right|$$

is always less than $1/2$. It follows that $\text{Fib}(n)$ is the closest integer to

$$\frac{1}{\sqrt{5}}\phi^n$$

and so is given, using the greatest integer function, as

$$\text{Fib}(n) = \left[\frac{1}{\sqrt{5}}\phi^n + \frac{1}{2} \right].$$

From this formula it follows that the n th Fibonacci number increases exponentially with n .

4 The Golden Section Search

The golden section search is an iterative technique for isolating the extreme value (minimum or maximum) of a unimodal function. A unimodal function is a function defined on an interval that has only one relative extreme point. In the case of the search for a minimum, one is given two end points that bracket the extremum and a third point in the bracketing interval where the function value at the internal point is smaller than at the interval end points. The task is to select a new point and to select a new set of three points from the four that improve the bracketing. It turns out that the golden ration ϕ plays a role in the optimal selection of these points. This algorithm is often discussed in books on optimization theory. The "Numerical Recipes" books, give functions for this calculation. For example see "Numerical Recipes in C." We give the essence of the algorithm below.

Given a point c on a line, points x_1 and x_2 are said to be symmetric with respect to c if the distance from x_1 to c equals the distance from x_2 to c , and x_1 and x_2 are on opposite sides of c . That is,

$$d(x_1, c) = d(x_2, c).$$

This is equivalent to the condition that the center point between x_1 and x_2 is equal to c . That is,

$$\frac{x_1 + x_2}{2} = c.$$

Given an interval $[a, b]$, points x_1 and x_2 are symmetric with respect to this interval, if they are symmetric with respect to the center of the interval. This means that

$$x_1 + x_2 = a + b.$$

So given a point, a symmetric partner can be found from this equation.

Suppose a point x_1 divides the interval $[a, b]$ in mean and extreme proportion. Then clearly the symmetric point

$$x_2 = a + b - x_1,$$

also divides the interval $[a, b]$ in mean and extreme proportion.

Proposition Suppose the symmetric points x_1 and x_2 divide the interval $[a, b]$ in mean and extreme proportion, and suppose

$$a < x_1 < x_2 < b.$$

Then point x_1 divides the interval $[a, x_2]$ in mean and extreme proportion. And also point x_2 divides the interval $[x_1, b]$ in mean and extreme proportion.

Proof. Point x_2 divides the interval $[a, b]$ in mean and extreme proportion. This means that

$$\frac{d(a, b)}{d(a, x_2)} = \phi = \frac{1 + \sqrt{5}}{2}.$$

Then

$$d(a, x_2) = \frac{1}{\phi}d(a, b).$$

Also

$$d(a, x_1) = \left(1 - \frac{1}{\phi}\right)d(a, b).$$

So

$$\frac{d(a, x_2)}{d(a, x_1)} = \frac{1/\phi}{1 - 1/\phi} = \frac{1}{\phi - 1}.$$

The Golden Ratio ϕ satisfies

$$\phi^2 - \phi - 1 = 0,$$

so

$$\phi(\phi - 1) = 1,$$

or

$$\phi = \frac{1}{\phi - 1}.$$

So

$$\frac{d(a, x_2)}{d(a, x_1)} = \phi.$$

This means that point x_1 divides interval $[a, x_2]$ in mean and extreme proportion. Similarly, point x_2 divides interval $[x_1, b]$ in mean and extreme proportion.

Now we are ready to describe the Golden Section optimization algorithm. Let us suppose that we are to find the minimum value of a unimodal function $f(x)$. Suppose we have an interval $[a, b]$ that brackets the minimum value. Suppose that we have, or that we select an internal point x_1 in the interval. Then by the definition of a unimodal function we must have the value $f(x_1)$ less than both $f(a)$ and $f(b)$. Further suppose that x_1 divides the interval in mean and extreme ratio. Now select the point x_2 that is symmetric to x_1 . Recall that this point will be

$$x_2 = a + b - x_1.$$

If

$$x_1 > x_2,$$

then interchange the points. If

$$f(x_1) < f(x_2)$$

then clearly the interval $[a, x_2]$ brackets the minimum and x_1 is the new internal point. If

$$f(x_2) < f(x_1)$$

then clearly $[x_1, b]$ brackets the minimum and x_2 is the new internal point. So select the interval that brackets the minimum. This becomes the new interval replacing $[a, b]$, and either x_1 or x_2 becomes the new value of x_1 . The length of the new bracketing interval has been reduced by the factor $\phi - 1 = .618033988749895$. This process is repeated until the bracketing interval is sufficiently small. This bracketing is similar to the bracketing of a root in locating a root of an equation by the bisection method. In that case

the interval length is reduced by $1/2$ at each step. So convergence for the Golden Section Search is somewhat slower. Note for example that

$$.618033988749895^{50} = 1.9728 \times 10^{-11},$$

so 50 steps gives quite a bit of accuracy.

5 Phyllotaxis, the Golden Rectangle, Numerology, and Pseudoscience

It is said that Fibonacci created his numbers in formulating a problem about rabbits. So at time one there was one pair of rabbits. After a month passes, at time 2, these rabbits are able to mate. There is still 1 pair of rabbits. Rabbits take one month to develop. After another month at time 3, they have given birth to a second pair of rabbits, so there are now 2 pair. These rabbits live forever. So if there are $P(n)$ rabbits at time n , this is the rabbit pairs that were alive at time $n - 1$, which is $P(n - 1)$, and the new rabbit pairs that were just born. Every pair of rabbits produces a new pair at time n , when they are two or more months old. This set of rabbit pairs is exactly the number of rabbit pairs alive at time $n - 2$, namely $P(n - 2)$. Hence $P(n) = P(n - 1) + P(n - 2)$. So the number of rabbit pairs at time n is the Fibonacci number $\text{Fib}(n)$.

As given above the Golden Ratio results from dividing an interval in mean and extreme proportion as discussed in Euclid's Elements. The Golden Rectangle has this ratio of length to width. It is said to be the most beautiful rectangle. However, referring to figure 1, the Parthenon may be in the Golden Rectangle form simple because the proportions are easy to construct by using a compass. Such geometrical constructions were favored in the time of the ancient Greeks, when Mathematics was Geometry. This beautiful rectangle idea seems to me to be a bit mystical and pseudoscientific.

Phyllotaxis is the classification of plants by the arrangement of their leaves, and by say their seed arrangement, for example in the case of the sunflower. It is said that many of these arrangements involve Fibonacci numbers. For example leaves on a plant might be arranged in a spiral on a stem where each new leaf occurs at an angle that is $2/5$ of the full rotation angle. And 2 and 5 are Fibonacci numbers. A certain pine cone might

have 8 spirals in one direction and 15 in another. The numbers 8 and 15 are Fibonacci numbers. There are growth theories purporting to explain such occurrences. See for example the books **The Algorithmic Beauty of Plants** and Coxiter's classic *Introduction to Geometry*.

However this phyllotaxis connection might just be numerology and pseudoscience where certain examples are picked out that happen to agree with this Fibonacci hypotheses. Suppose that in the census some town is found to have a population of 75025. Should we remark on the amazing fact that this is a Fibonacci number? There is much material written about this sort of thing and it can be found on the Internet. For myself I have better things to do than pursue such possible mysticism.

6 The Pentagon and the Golden Ratio

Let a be the length of a side of a pentagon.

Let b be the length of one of the five chords of this pentagon. These chords form a five pointed star inside the pentagon. By drawing lines from the center of the pentagon to each of the five vertices of the pentagon. We have five equal isosceles triangles inside the pentagon. Let α be the measure of the two equal angles of each triangle. Then the sum of the angles of the five triangles is

$$5\pi = 2\pi + 10\alpha.$$

So

$$\alpha = \frac{3\pi}{10}$$

Let a side of the pentagon be AB . Let a chord be drawn from vertex A to the vertex after B . Let C be the center point of this chord. Then triangle ABC is a right triangle with AB having length a and side AC having length $b/2$. Angle CAB is

$$\frac{\pi}{2} - \alpha = \frac{\pi}{5}$$

Then

$$\frac{b}{2} = a \cos(\pi/5).$$

Now letting $\theta = \pi/5$, we have

$$\cos(5\theta) = \cos(\pi) = -1.$$

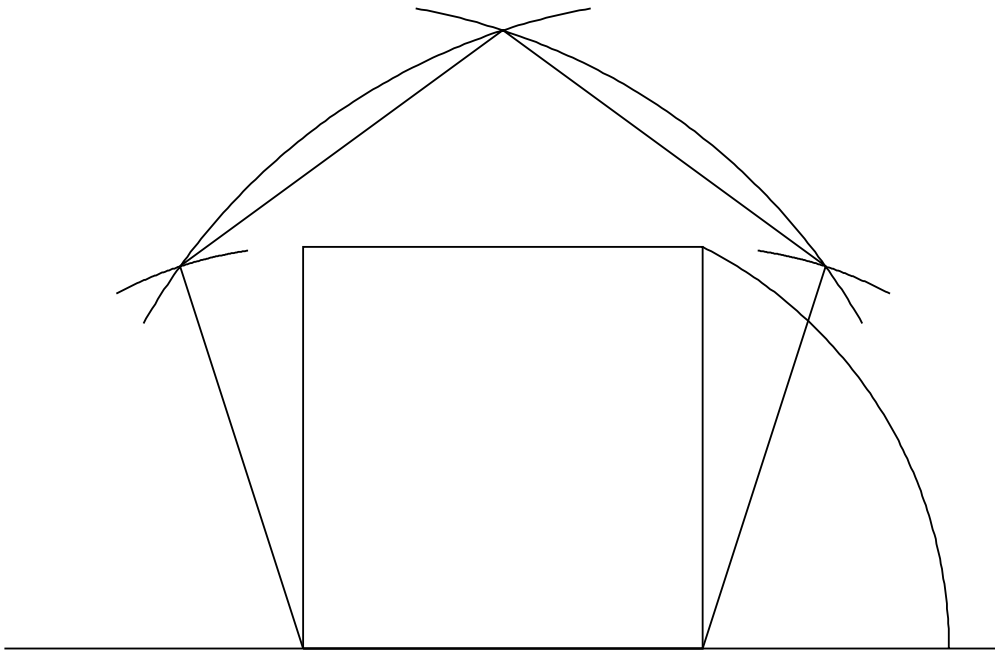


Figure 4: The construction of a pentagon with ruler and compass using the Golden Ratio. First we draw a square with a side length equal to the desired side length of the pentagon. We take the lower left corner of the square as the first vertex of the pentagon. Next we draw the small arc, using the midpoint of the bottom side of the square as center. We obtain the base length r of a golden rectangle, relative to the length of the side of the square. Using this r as radius, and bottom left vertex of the square as center, we draw an arc that passes through the third and fourth vertex of the pentagon. The fourth vertex is directly above the midpoint of the base of the square.

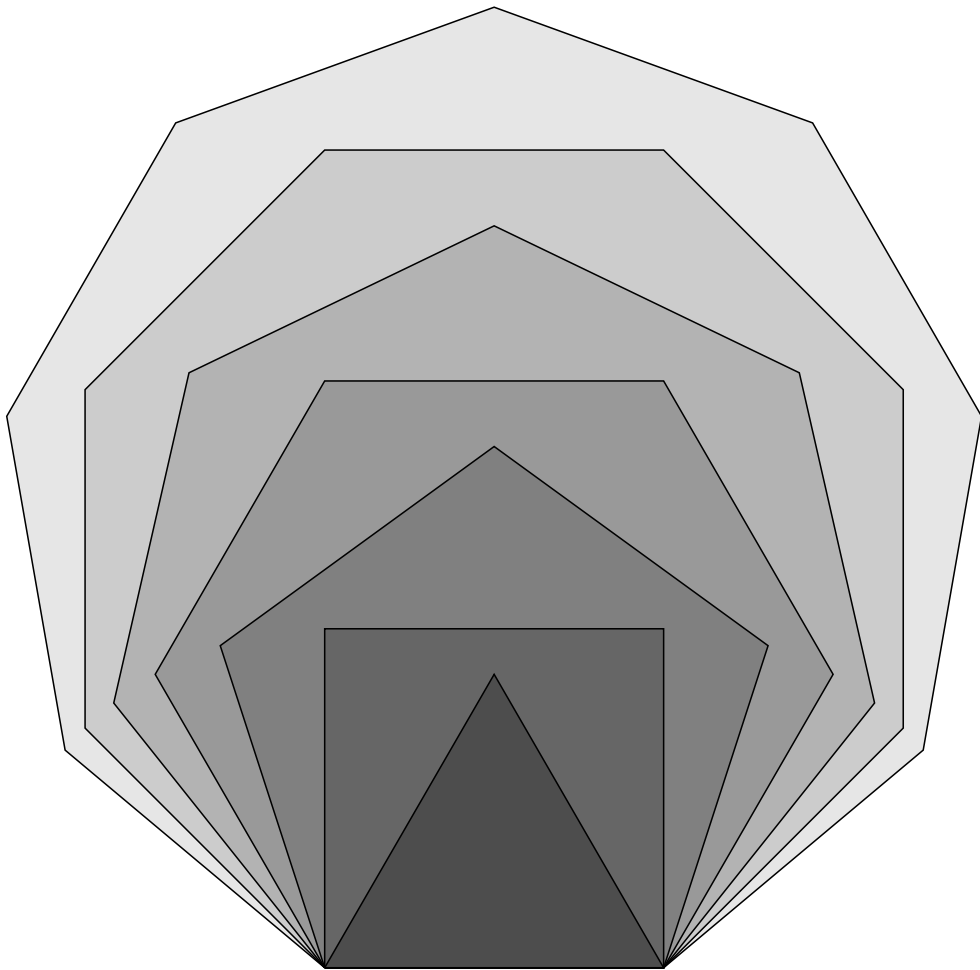


Figure 5: Polygons with common edge from program *fibfig.ftn*.

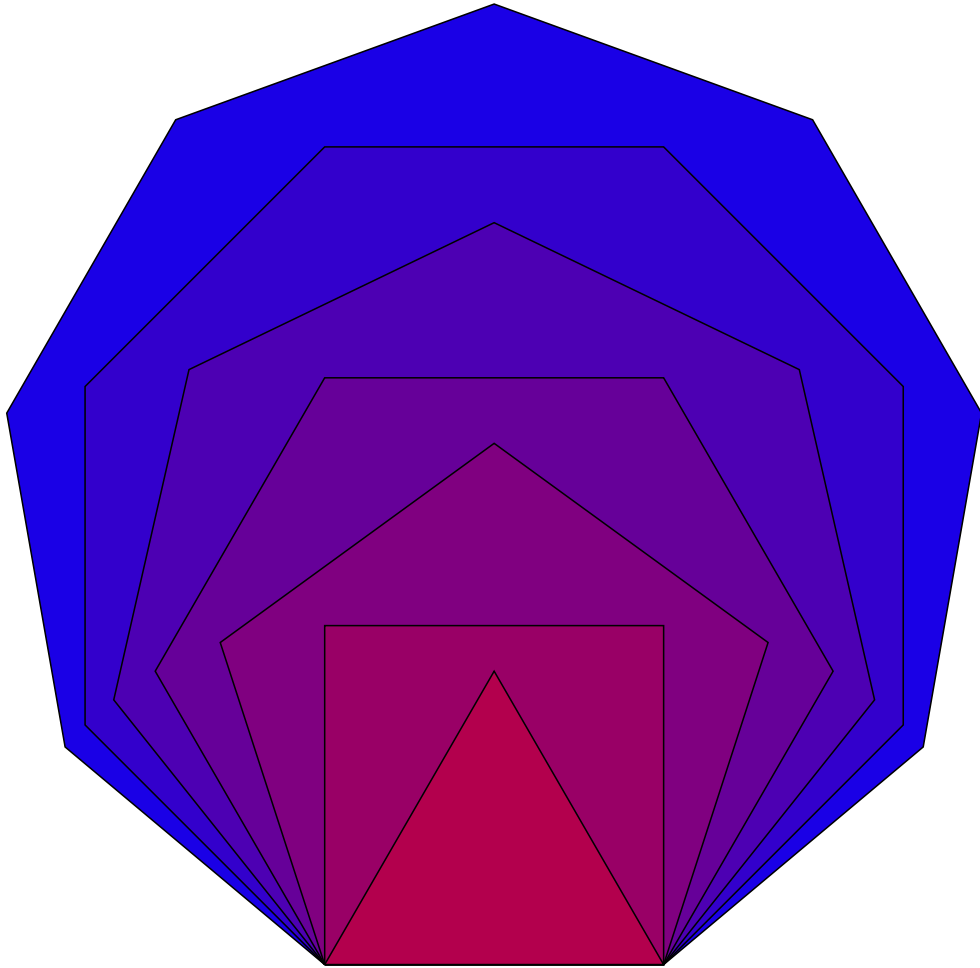


Figure 6: Polygons with common edge from program *fillcurves.ftn* (polygons filled with color).

Letting $x = \cos(\theta)$, and expanding $\cos(5\theta)$ using a multiple angle formula, the equation becomes

$$16x^5 - 20x^3 + 5x + 1 = 0.$$

This polynomial factors as

$$(x + 1)(4x^2 - 2x - 1)^2 = 0.$$

The roots are

$$-1, \frac{1 + \sqrt{5}}{4}, \frac{1 - \sqrt{5}}{4},$$

where the last two roots are repeated. The value we are looking for, $x = \cos(\pi/5)$, is positive, so must be

$$\cos(\pi/5) = \frac{1 + \sqrt{5}}{4}.$$

Thus

$$\frac{b}{a} = \frac{1 + \sqrt{5}}{2},$$

which is the Golden Ratio.

This gives a way of drawing a pentagon with ruler and compass. Vertices are located as intersections of various arcs, each with radii equal to either a or b . Given side a , b may be found geometrically as usual by inscribing a square of side a in a semicircle. See the figure.

7 Appendix: How the Figures in This Document Were Drawn

Figures are included in Latex documents like this as Postscript files. Most of the figures were created with a Fortran program that created a file of type "eg." This is a graphics file of my design. It may contain lines, Bezier curves, circles and arcs, text, and filled curves. The eg file is translated to Postscript with my C program called **eg2ps.c**. Additional text and symbols are sometimes added to the postscript file using Ghostview (a program using the Ghostscript engine). Ghostview will display a postscript file and

print it to various printers. It also displays a cursor position so that one can determine where text or symbols are to be positioned. A small file is created with the additional text and the coordinates of its location. A set of automated programs works with Ghostview to merge the new information into the document. Although this is quite simple to do, the method is probably too technical to be communicated to others who have only a casual interest.

Now we list the program that produced the eg files for the figures 2 through 5. It relies on various subroutines that reside in a source library called emerylib.ftn, and which are not special to this particular program. Here is the listing, which contains the source of the subroutines as well as the main program:

```
c creates figures for fibonaccitex
c 10/8/2010
  implicit real*8(a-h,o-z)
  dimension a(10),b(4)
  dimension p1(2),p2(2)
  dimension sp(2)
  dimension v(2,20)
  dimension cn(2)
  nf1=1
  nf2=2
  nf3=3
  nf4=4
  open(nf1,file='fibonaccif2.eg',status='unknown')
  open(nf2,file='fibonaccif3.eg',status='unknown')
  open(nf3,file='fibonaccif4.eg',status='unknown')
  open(nf4,file='fibonaccif5.eg',status='unknown')
  write(nf1,'(a)')'v -1 1 -1 1'
  write(nf1,'(a)')'w -1.1 1.1 0 2.2'
  write(nf2,'(a)')'v -1 1 -1 1'
  write(nf2,'(a)')'w -1.1 1.1 0 2.2'
  write(nf3,'(a)')'v -1 1 -1 1'
  write(nf3,'(a)')'w -3.1 3.1 0 6.2'
  write(nf4,'(a)')'v -1 1 -1 1'
  write(nf4,'(a)')'w-3.02295 3.02295 -0.18745 5.85845'
  xc=0.
  yc=0.
  r=1.
  one=1.
  pi=4.*atan(one)
  a1=0.
  a2=pi
  a3=100.*pi/180.
  a4=90.*pi/180.
  a5=135.*pi/180.
  a6= 45.*pi/180.
  npts=100
  call xdarcc(nf1,xc,yc,r,a1,a2,npts)
  x=1.
```

```

y=0.
call xmove(nf1,x,y)
y=2.0
call xdraw(nf1,x,y)
x=-1.0
call xdraw(nf1,x,y)
y=0.0
call xdraw(nf1,x,y)
x=1.0
call xdraw(nf1,x,y)
x=0.0
call xmove(nf1,x,y)
x=-1.
y=2.0
call xdraw(nf1,x,y)
x=0.
y=0.
call xmove(nf1,x,y)
x=1.
y=2.0
call xdraw(nf1,x,y)
a(1)=0.
a(2)=0.
a(3)=r
a(4)=0.
a(5)=pi
b(1)=0.
b(2)=0.
b(3)=-1.
b(4)=2.
call intla(a,b,n,p1,p2,ier)
write(*,*)' n= ',n
write(*,*)' x= ',p1(1)
write(*,*)' y= ',p1(2)

x=p1(1)
y=0.
call xmove(nf1,x,y)
x=p1(1)
y=p1(2)
call xdraw(nf1,x,y)
x=-p1(1)
call xdraw(nf1,x,y)
y=0.
call xdraw(nf1,x,y)
c draw fig 3
call xdarc(nf2,xc,yc,r,a1,a2,npts)
x=-1.
y=0.
call xmove(nf2,x,y)
x=1.
call xdraw(nf2,x,y)
c draw inscribed square
x=p1(1)
y=0.

```

```

    call xmove(nf2,x,y)
    x=p1(1)
    y=p1(2)
    call xdraw(nf2,x,y)
    x=-p1(1)
    call xdraw(nf2,x,y)
    y=0.
    call xdraw(nf2,x,y)
c draw triangle
    x=-1.
    y=0.
    call xmove(nf2,x,y)
    x=-p1(1)
    y=p1(2)
    call xdraw(nf2,x,y)
    x=1.0
    y=0.
    call xdraw(nf2,x,y)

c draw fig 4
c make pentagon
    n=5
    sp(1)=-1.
    sp(2)=0.
    sl=2.
    sa=0.
    call polygon(n,sp,sl,sa,v,cn,r,d)
    chordd=sqrt((v(1,3)-v(1,1))**2+(v(2,3)-v(2,1))**2)
    iflag=1
    call drawcurve(nf3,n,iflag,v)
    a1=30.*pi/180.
    a2=80.*pi/180.
    xc=v(1,1)
    yc=v(2,1)
    r1=chordd
    call xdarcm(nf3,xc,yc,r1,a1,a2,npts)
    a1=100.*pi/180.
    a2=150.*pi/180.
    xc=v(1,2)
    yc=v(2,2)
    r1=chordd
    call xdarcm(nf3,xc,yc,r1,a1,a2,npts)
c make square
    n=4
    call polygon(n,sp,sl,sa,v,cn,r,d)
    call drawcurve(nf3,n,iflag,v)
c make arc with radius sl center at first vertex
    a1=98.*pi/180.
    a2=118.*pi/180.
c
    a1=0.
c
    a2=pi
    xc=v(1,1)
    yc=v(2,1)
    r2=sl
    call xdarcm(nf3,xc,yc,r2,a1,a2,npts)

```

```

c make arc with radius s1 center at second vertex
  a1=62.*pi/180.
  a2=82.*pi/180.
c   a1=0.
c   a2=pi/2
  xc=v(1,2)
  yc=v(2,2)
  r2=s1
  call xdarc(nf3,xc,yc,r2,a1,a2,npts)

c make arc from midpoint of base to get golden ratio
  a1=62.*pi/180.
  a2=82.*pi/180.
  a1=0.
  a2=atan(2.*one)
  xc=(v(1,2)+v(1,1))/2.
  yc=v(2,2)
  r3=sqrt(5.)*s1/2.
  call xdarc(nf3,xc,yc,r3,a1,a2,npts)
  x=-2.5
  y=0.
  call xmove(nf3,x,y)
  x=2.5
  call xdraw(nf3,x,y)
c draw some polygons
c draw some polygons
  do i=9,3,-1
    n=i
    call polygon(n,sp,s1,sa,v,cn,r,d)
    call drawcurve(nf4,n,iflag,v)
    g=i/10.
    write(nf4,'(a)')'p gsave'
    write(nf4,'(a,g15.8,a)')'p ',g,' setgray'
    write(nf4,'(a)')'p fill'
    write(nf4,'(a)')'p grestore'
    write(nf4,'(a)')'p stroke'
  end do
end

c
c+ xdarc arc into external plot file
  subroutine xdarc(nfile,xc,yc,r,a1,a2,npts)
  implicit real*8(a-h,o-z)
c nfile-unit number for output file
c xc,yc-arc center
c   r-arc radius
c a1,a2-start and end angles
c npts-number of points in arc
  character s*25,t*80
c   write(*,*)' xdarc'
  t='a'
  n=2
  call str(xc,s)
  l=lenstr(s)
  t(n:80)=s

```

```

n=n+1+1
call str(yc,s)
l=lenstr(s)
t(n:80)=s
n=n+1+1
call str(r,s)
l=lenstr(s)
t(n:80)=s
n=n+1+1
call str(a1,s)
l=lenstr(s)
t(n:80)=s
n=n+1+1
call str(a2,s)
l=lenstr(s)
t(n:80)=s
n=n+1+1
an=npts
call str(an,s)
l=lenstr(s)
t(n:80)=s
write(nfile,'(a)')t(1:(n+1-1))
return
end

c+ lenstr  nonblank length of string
function lenstr(s)
c length of the substring of s obtained by deleting all
c trailing blanks from s. thus the length of a string
c containing only blanks will be 0.
character s*(*)
lenstr=0
n=len(s)
do 10 i=n,1,-1
if(s(i:i) .ne. ' ')then
lenstr=i
return
endif
10 continue
return
end

c+ str floating point number to string
subroutine str(x,s)
implicit real*8(a-h,o-z)
character s*25,c*25,b*25,e*25
zero=0.
if(x.eq.zero)then
s='0'
return
endif
write(c,'(g11.4)')x
read(c,'(a25)')b
l=lenstr(b)

```

```

do 10 i=1,1
n1=i
if(b(i:i).ne.' ')go to 20
10 continue
20 continue
if(b(n1:n1).eq.'0')n1=n1+1
b=b(n1:1)
l=l+1-n1
k=index(b,'E')
if(k.gt.0)e=b(k:1)
if(k.gt.0)then
s=b(1:(k-1))
k1=index(b,'E+0')
if(k1.gt.0)then
e='E'//b((k1+3):1)
else
k1=index(b,'E+')
if(k1.gt.0)e='E'//b((k1+2):1)
endif
k1=index(b,'E-0')
if(k1.gt.0)e='E-'//b((k1+3):1)
l=k-1
else
s=b
endif
j=index(s,'.')
n2=1
if(j.ne.0)then
do 30 i=1,1
n2=1+1-i
if(s(n2:n2).ne.'0')go to 40
30 continue
endif
40 continue
s=s(1:n2)
if(s(n2:n2).eq.'.')then
s=s(1:(n2-1))
n2=n2-1
endif
if(k.gt.0)s=s(1:n2)//e
return
end
c+ xdrws put line segment in gi file
subroutine xdrws(nfile,a)
implicit real*8(a-h,o-z)
logical tk,fx
common tk,fx
dimension a(4)
x=a(1)
y=a(2)
call xmove(nfile,x,y)
x=a(3)
y=a(4)
call xdraw(nfile,x,y)
return

```

```

        end
c+ xdraw draw parameters to external plot file
      subroutine xdraw(nfile,x,y)
        implicit real*8(a-h,o-z)
        character s*25,t*80
        t='d'
        n=2
        call str(x,s)
        l=lenstr(s)
        t(n:80)=s
        n=n+l+1
        call str(y,s)
        l=lenstr(s)
        t(n:80)=s
        write(nfile,'(a)')t(1:(n+l-1))
        return
      end

c
c+ xmove move parameters to external plot file
      subroutine xmove(nfile,x,y)
        implicit real*8(a-h,o-z)
        character s*25,t*80
        t='m'
        n=2
        call str(x,s)
        l=lenstr(s)
        t(n:80)=s
        n=n+l+1
        call str(y,s)
        l=lenstr(s)
        t(n:80)=s
        write(nfile,'(a)')t(1:(n+l-1))
        return
      end

c+ ionarc test for point on arc
      function ionarc(c,r,s,b,e,p)
c      5/27/96
c input:
c c center of arc
c r radius
c s start point
c b between point
c e end point
c p point to be tested
c output:
c returns 0 if not on arc
c returns 1 if on arc
c is considered on the arc if the distance to
c the arc is less than epsilon = 1.e-8*r
        implicit real*8 (a-h,o-z)
        dimension c(*),s(*),b(*),e(*),p(*)
        dimension v1(3),v2(3),v3(3),v4(3)
        zero=0.
        m=0

```

```

epsilon=1.e-8*r
write(*,*)' start point= ',(s(i),i=1,2)
write(*,*)' mid point= ',(b(i),i=1,2)
write(*,*)' end point= ',(e(i),i=1,2)
v1n=0.
v2n=0.
v3n=0.
v4n=0.
do i=1,2
  v1(i)=e(i)-s(i)
  v2(i)=p(i)-s(i)
  v3(i)=p(i)-e(i)
  v4(i)=p(i)-c(i)
  v1n= v1n + v1(i)**2
  v2n= v2n + v2(i)**2
  v3n= v3n + v3(i)**2
  v4n= v4n + v4(i)**2
enddo
v1n=sqrt(v1n)
v2n=sqrt(v2n)
v3n=sqrt(v3n)
v4n=sqrt(v4n)
if(abs(v4n - r) .gt. epsilon)then
  ionarc = 0
  return
endif
if(v2n .lt. epsilon)then
  ionarc=1
  return
endif
if(v3n .lt. epsilon)then
  ionarc=1
  return
endif
c
if the chord is very short, use the circle tangent
if(v1n .lt. epsilon)then
  v1n=0.
  do i=1,2
    v1(i)=(s(1)+e(i))/2. - c(i)
    v1n= v1n + v1(i)**2
  enddo
  v1n=sqrt(v1n)
  tmp=v1(1)
  v1(1)=v1(2)
  v1(2)=-tmp
endif
do i=1,2
  v1(i) = v1(i)/v1n
  v2(i) = v2(i)/v2n
enddo
v1(3)=1.
v2(3)=1.
call crsspr(v1,v2,v3)
d1=v3(3)
write(*,*)' d1 = ',d1

```

```

        if(d1 .ne. zero)then
            do i=1,2
                v4(i)=b(i)-s(i)
            enddo
            v4(3)=1.
            call crsspr(v1,v4,v3)
            d2=v3(3)
            write(*,*)' d2 = ',d2
            if(d1*d2 .gt. zero)then
                m=1
            endif
        endif
        ionarc=m
        write(*,*)' ionarc= ',ionarc
        return
    end

c+ intlc intersection of line and circle
    subroutine intlc(p,q,c,r,e,f,ier)
        implicit real*8 (a-h,o-z)
c 5/24/96
c Note: all vectors have projective space dimension 2,
c that is each vector should have a dimensioned size of 3.
c This subroutine sets each third coordinates of
c p,q,c,e,f, to 1
c (see geometry.tex for the algorithm)
c input:
c   p,q   points on line
c   c     center of circle
c   r     radius of circle
c output:
c   e     first intersection point
c   f     second intersection point
c   ier   ier=0, intersection points
c         ier=1, no intersection points, or input data error

c   dimension p(3),q(3),c(3),e(3),f(3)
c   dimension p(*),q(*),c(*),e(*),f(*)
c   dimension a(3),b(3),d(3)
c   real*8 l1(3),l2(3)
c   zero=0.
c   p(3)=1.
c   q(3)=1.
c   c(3)=1.
c   an=0.
c   do i=1,2
c       a(i)=p(i)-q(i)
c       an=an+a(i)*a(i)
c   enddo
c   an=sqrt(an)
c   if(an .eq. zero)then
c       ier=1
c       return
c   endif
c   do i=1,2
c       a(i) = a(i)/an

```

```

        enddo
        call crsspr(p,q,l1)
        b(1)=a(2)
        b(2)=-a(1)
        b(3)=0.
        call crsspr(c,b,l2)
        call crsspr(l1,l2,d)
        d(1)=d(1)/d(3)
        d(2)=d(2)/d(3)
        alpha=sqrt((d(1)-c(1))*(d(1)-c(1))+(d(2)-c(2))*(d(2)-c(2)))
        if( alpha .le. r)then
            beta = sqrt(r*r - alpha*alpha)
        else
            ier=1
            return
        endif
        do i=1,2
            e(i) = d(i) - beta*a(i)
            f(i) = d(i) + beta*a(i)
        enddo
        e(3)=1.
        f(3)=1.
        ier=0
        return
    end

c+ crsspr vector cross product.
    subroutine crsspr(a,b,c)
        implicit real*8(a-h,o-z)
c      c=product of a and b
        dimension a(3),b(3),c(3)
        c(1)=a(2)*b(3)-a(3)*b(2)
        c(2)=a(3)*b(1)-a(1)*b(3)
        c(3)=a(1)*b(2)-a(2)*b(1)
        return
    end

c+ intl intersection of line and arc
    subroutine intl(a,b,n,p1,p2,ier)
        implicit real*8 (a-h,o-z)
c 5/24/96
c      input:
c      a arc parameters: center (a(1),a(2)), radius a(3),
c      angle1=a(4), angle2=a(5)
c      b line parameters: x1=b(1),y1=b(2),x2=b(3),y2=b(4)
c      n number of intersection points
c      output:
c      p1 first intersection point
c      p2 second intersection point
c      ier 0, no error
        dimension a(*),b(*),p1(*),p2(*)
        dimension p(3),q(3),c(3)
        dimension s(3),se(3),e(3)
        p(1)=b(1)
        p(2)=b(2)
        q(1)=b(3)

```

```

q(2)=b(4)
c(1)=a(1)
c(2)=a(2)
r=a(3)
call intl(c,p,q,c,r,p1,p2,ier)
if(ier .eq. 0)then
  a1=a(4)
  a2=a(5)
  am=(a1+a2)/2.
  s(1)= c(1)+r*cos(a1)
  s(2)= c(2)+r*sin(a1)
  se(1)= c(1)+r*cos(am)
  se(2)= c(2)+r*sin(am)
  e(1)= c(1)+r*cos(a2)
  e(2)= c(2)+r*sin(a2)
  n=0
  m= ionarc(c,r,s,se,e,p1)
  n=n+m
  m= ionarc(c,r,s,se,e,p2)
  if( m .eq. 1)then
    if(n .eq. 0)then
      p1(1)=p2(1)
      p1(2)=p2(2)
    endif
    n=n+m
  endif
endif
return
end

c+ arg argument of point (x,y), -pi <= arg <= pi
real*8 function arg(x,y)
c revised 6/4/93
implicit real*8(a-h,o-z)
t=atan2(y,x)
one=1.
pi=4.*atan(one)
if(t .gt. pi)then
  t=t-2.*pi
endif
arg=t
return
end

c+ polygon create vertices of polygon
subroutine polygon(n,sp,sl,sa,v,c,r,d)
implicit real*8 (a-h,o-z)
dimension sp(*)
dimension v(2,*)
dimension c(2)

c input:
c      n  number of sides

```

```

c    sp  starting point coordinates (x=sp(1),y=sp(2))
c    sl  side length
c    sa  first side angle direction
c output:
c    v   coordinates of vertices (x1=v(1,i), y1=v(2,i)), i=1 to n
c    c   center (c(1),c(2))
c    r   radius
c    d   normal distance from side to center
c    va  angle of the vector from center to first vertex
v(1,1)=sp(1)
v(2,1)=sp(2)
v(1,2)=sp(1)+sl*cos(sa)
v(2,2)=sp(2)+sl*sin(sa)
one=1.
pi=4.*atan(one)
a=pi/n
r=sl/(2.*sin(a))
d=r*cos(a)
xm=(v(1,1)+v(1,2))/2.
ym=(v(2,1)+v(2,2))/2.
b=sa+pi/2.
c(1)=xm+d*cos(b)
c(2)=ym+d*sin(b)
x=v(1,1)-c(1)
y=v(2,1)-c(2)
va=arg(x,y)
c    write(*,*)' arg= ',va*180./pi, 'degrees'

do i=3,n
a=va+(i-1)*(2.*pi)/n
c    write(*,*)' i,ang= ',i,a*180./pi
v(1,i)=c(1)+r*cos(a)
v(2,i)=c(2)+r*sin(a)
enddo
return
end

c+ drawcurve draw a curve defined by n points to an eg file
subroutine drawcurve(nf,n,iflag,p)
implicit real*8 (a-h,o-z)
dimension p(2,*)
c Input:
c    nf  file number where the points are written
c    n   number of points
c    iflag if iflag=1, draw a line from the last point to the first point
c Output:
c    p   points, ith point x=p(1,i), y= p(2,i)
do i=1,n
x=p(1,i)
y=p(2,i)
if(i .eq. 1)then
call xmove(nf,x,y)
else
call xdraw(nf,x,y)
endif
endif

```

```

enddo
if(iflag .eq. 1)then
  x=p(1,1)
  y=p(2,1)
  call xdraw(nf,x,y)
end if
return
end

```

Here is the resulting eg file for figure 4 of this document is called fibonaccif4.eg. It shows the construction of a pentagon with ruler and compass, and contains only moves, draws, and arcs.

```

v -1 1 -1 1
w -3.1 3.1 0 6.2
m-1 0
d1 0
d1.618 1.902
d.1884E-15 3.078
d-1.618 1.902
d-1 0
a-1 0 3.236 .5236 1.396 100
a1 0 3.236 1.745 2.618 100
m-1 0
d1 0
d1 2
d-1 2
d-1 0
a-1 0 2 1.71 2.059 100
a1 0 2 1.082 1.431 100
a0 0 2.236 0 1.107 100
m-2.5 0
d2.5 0

```

The postscript file fibonaccif4.ps is produced from this file with the following command

```
eg2ps fibonaccif4.eg fibonaccif4.ps
```

And finally here is a listing of the postscript file fibonaccif4.ps.

```

%!PS
%%BoundingBox: 0 50 450 500
%%Creator: eg2ps.c by Jim Emery
%%EndComments
72 300 div 72 300 div scale
100 100 translate
3 setlinewidth

```

```
newpath
677 0 moveto
1323 0 lineto
1522 614 lineto
1000 993 lineto
478 614 lineto
677 0 lineto
1581 522 moveto
1577 530 lineto
1572 538 lineto
1567 546 lineto
1562 553 lineto
1558 561 lineto
1553 569 lineto
1548 577 lineto
1542 584 lineto
1537 592 lineto
1532 599 lineto
1527 607 lineto
1521 614 lineto
1516 622 lineto
1510 629 lineto
1505 637 lineto
1499 644 lineto
1493 651 lineto
1488 658 lineto
1482 665 lineto
1476 672 lineto
1470 679 lineto
1464 686 lineto
1458 693 lineto
1452 700 lineto
1446 707 lineto
1439 714 lineto
1433 720 lineto
1427 727 lineto
1420 734 lineto
1414 740 lineto
1407 746 lineto
1400 753 lineto
1394 759 lineto
1387 766 lineto
1372 618 lineto
1365 622 lineto
1358 626 lineto
1351 630 lineto
1344 634 lineto
1337 638 lineto
1330 641 lineto
1323 645 lineto
.....
.....
(the file has 533 lines, most of which are not displayed)
.....
stroke
```

showpage

This file is longer than the eg file, because each one line arc command has been rendered into a sequence of approximating short line segments.

The postscript figure is included in the document with the embedded commands:

```
\begin{figure}
\psfig{figure=fibonaccif4.ps,height=5in}
\caption{ Construction of a pentagon with ruler and compass using the Golden Ratio.}
\end{figure}
```

To demonstrate how postscript shades the interior of closed curves, here is the postscript file for figure 5.

```
%!PS
%%BoundingBox: 0 50 450 500
%%Creator: eg2ps.c by Jim Emery
%%EndComments
72 300 div 72 300 div scale
100 100 translate
3 setlinewidth
newpath
669 62 moveto
1331 62 lineto
1838 487 lineto
1952 1139 lineto
1622 1712 lineto
1000 1938 lineto
378 1712 lineto
48 1139 lineto
162 487 lineto
669 62 lineto
gsave
.90000000 setgray
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1799 530 lineto
1799 1191 lineto
1331 1659 lineto
669 1659 lineto
201 1191 lineto
201 530 lineto
669 62 lineto
gsave
.80000000 setgray
fill
grestore
```

```

stroke
669 62 moveto
1331 62 lineto
1743 579 lineto
1596 1224 lineto
1000 1511 lineto
404 1224 lineto
257 579 lineto
669 62 lineto
gsave
  .70000000    setgray
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1662 635 lineto
1331 1208 lineto
669 1208 lineto
338 635 lineto
669 62 lineto
gsave
  .60000000    setgray
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1535 691 lineto
1000 1080 lineto
465 691 lineto
669 62 lineto
gsave
  .50000000    setgray
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1331 724 lineto
669 724 lineto
669 62 lineto
gsave
  .40000000    setgray
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1000 635 lineto
669 62 lineto
gsave
  .30000000    setgray
fill
grestore

```

```
stroke
stroke
showpage
```

So each of the closed polygons is drawn and then filled with gray with the fill command. In postscript each object is drawn on top of any previous objects and may cover all or part of the previous objects.

Finally, the same thing can be done with color. In place of the setgray command is a setrgb command, setting the color with values for red green and blue. Here is the postscript file for figure 6, that was drawn with program fillcurves.ftn:

```
%!PS
%%BoundingBox: 0 50 450 500
%%Creator: eg2ps.c by Jim Emery
%%EndComments
72 300 div 72 300 div scale
100 100 translate
3 setlinewidth
newpath
669 62 moveto
1331 62 lineto
1838 487 lineto
1952 1139 lineto
1622 1712 lineto
1000 1938 lineto
378 1712 lineto
48 1139 lineto
162 487 lineto
669 62 lineto
gsave
.100000E+00 .000000 .900000 setrgbcolor
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1799 530 lineto
1799 1191 lineto
1331 1659 lineto
669 1659 lineto
201 1191 lineto
201 530 lineto
669 62 lineto
gsave
.200000 .000000 .800000 setrgbcolor
fill
grestore
stroke
669 62 moveto
1331 62 lineto
```

```

1743 579 lineto
1596 1224 lineto
1000 1511 lineto
404 1224 lineto
257 579 lineto
669 62 lineto
gsave
  .300000      .000000      .700000      setrgbcolor
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1662 635 lineto
1331 1208 lineto
669 1208 lineto
338 635 lineto
669 62 lineto
gsave
  .400000      .000000      .600000      setrgbcolor
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1535 691 lineto
1000 1080 lineto
465 691 lineto
669 62 lineto
gsave
  .500000      .000000      .500000      setrgbcolor
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1331 724 lineto
669 724 lineto
669 62 lineto
gsave
  .600000      .000000      .400000      setrgbcolor
fill
grestore
stroke
669 62 moveto
1331 62 lineto
1000 635 lineto
669 62 lineto
gsave
  .700000      .000000      .300000      setrgbcolor
fill
grestore
stroke
stroke
showpage

```

8 Appendix: Continued Fractions

This appendix is a copy of the document `continuedfractions.tex`.

8.1 The Continued Fraction Representation of a Rational Number

A continued fraction is a fraction written in a form like

$$\frac{p}{q} = a_1 + \frac{b_2}{a_2 + \frac{b_3}{a_3 + \frac{b_4}{a_4 + \frac{b_5}{a_5}}}},$$

where the a_i 's and the b_i 's are integers.

Any rational number $p/q > 1$ can be written in a form like

$$\frac{p}{q} = a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4 + \frac{1}{a_5}}}},$$

where the b 's are all equal to 1. This is called a simple continued fraction. This is the same as

$$\frac{p}{q} = (a_1 + 1/(a_2 + 1/(a_3 + 1/(a_4 + 1/(a_5))))).$$

We write this in abbreviated form as

$$\frac{p}{q} = a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4 + \frac{1}{a_5}}}},$$

and in an even more abbreviated form as

$$[a_1, a_2, a_3, a_4, a_5].$$

For example consider finding the continued fraction representation of the fraction

$$\frac{22}{9}.$$

Dividing 22 by 9 we get 2 with a remainder of 4, so

$$\frac{22}{9} = 2 + \frac{4}{9} = 2 + \frac{1}{9/4}.$$

Dividing 9 by 4 we get

$$\frac{9}{4} = 2 + \frac{1}{4}.$$

So

$$\frac{22}{9} = 2 + \frac{4}{9} = 2 + \frac{1}{9/4} = 2 + \frac{1}{2 + \frac{1}{4}} = [2, 2, 4].$$

With the same repeated division we find for example that

$$\frac{75948}{24175} = [3, 7, 15, 1, 212, 1].$$

An infinite continued fraction goes on forever,

$$[a_1, a_2, a_3, a_4, a_5, \dots].$$

In that case the sequence of continued fractions

$$\{[a_1], [a_1, a_2], [a_1, a_2, a_3], [a_1, a_2, a_3, a_4], \dots\}$$

is called the sequence of partial quotients. The partial quotients are also called convergents. So the convergents are

$$\begin{aligned} [a_1] &= a_1, \\ [a_1, a_2] &= a_1 + \frac{1}{a_2}, \\ [a_1, a_2, a_3] &= a_1 + \frac{1}{a_2 + \frac{1}{a_3}}, \end{aligned}$$

and so on.

It turns out that such an infinite simple continued fraction converges to some real number, which is the value of the infinite continued fraction. This value is defined as the limit of the sequence of convergents.

In the case of expanding an irrational number into an infinite continued fraction

$$[a_1, a_2, a_3, a_4, a_5, \dots],$$

the convergents can be used as rational approximations to the irrational number. For example π can be written as a partial fraction as

$$\pi = [3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 2, 1, 1, 2, 2, 2, 2, 1, 84, 2, \dots]$$

We will show that the convergents will converge to the value of the infinite continued fraction. Note that the second partial quotient for π is

$$[3, 7] = \frac{22}{7},$$

which is often used as an approximation to π .

We now show how to compute the partial quotients in an iterative way. For arbitrary integers p and q , let us write

$$p = a_1q + r_1,$$

where $a_1 = [p/q]$ and r_1 is less than q . The brackets represent the greatest integer function. Then

$$\frac{p}{q} = a_1 + \frac{r_1}{q} = a_1 + \frac{1}{\frac{q}{r_1}},$$

where

$$\frac{q}{r_1} > 1.$$

This can be continued as follows

$$\begin{aligned} q &= a_2r_1 + r_2, \frac{q}{r_1} = a_2 + \frac{r_2}{r_1}, r_2 < q \\ r_1 &= a_3r_2 + r_3, \frac{r_1}{r_2} = a_2 + \frac{r_3}{r_2}, r_3 < r_2 \\ &\dots \\ r_n &= a_{n+2}r_{n+1} + r_{n+2}, \frac{r_n}{r_{n+1}} = a_{n+2} + \frac{r_{n+2}}{r_{n+1}}, r_{n+2} < r_{n+1}, \end{aligned}$$

until finally the remainder is zero. This algorithm for computing the partial quotients of a continued fraction is equivalent to the Euclidian algorithm for finding the GCD of integers p and q . The program **cf.java** computes the quotients of the continued expansion of a rational number.

```
//cf.java, continued fraction expansion of a rational number. 12/6/07
//
class cf {
    public static void main(String args[]) {
        int cnv[];
```

```

int p;
int q;
int r;
int a;
int b;
int c;
int n;
int gcd;
int sign;
cnv = new int[50];
n=0;
if(args.length <= 1){
    System.out.println(" Continued fraction expansion of  p/q ");
    System.out.println(" Usage: java cv p q ");
    return;
}
if(args.length > 1){
    //System.out.println(" The first argument is " + args[0]);
    p=Integer.parseInt(args[0]);
    //System.out.println(" The second argument is " + args[1]);
    q=Integer.parseInt(args[1]);
    sign=1;
    if(q < 0){
        sign=sign*(-1);
        q=-q;
    }
    if(p < 0){
        sign=sign*(-1);
        p=-p;
    }
    System.out.println(" The partial quotients of " + sign*p + "/" + q + " are:");
    //p = 101;
    //q = 59;
    a=p;
    b=q;
    gcd=q;
    // p/q = a + r/q
    for(int i=1;i<50;i++){
        c = a/b;
        r= a-c*b;
        // a/b = c + r/b
        //System.out.println(" c = " + c + " r = " + r);
        a=b;
        b=r;
        cnv[i]=c;
        n++;
        if(r == 0)break;
        gcd=r;
    }
    System.out.print(" [");
    for(int i=0;i<n;i++){
        if(i != (n-1)){
            System.out.print(sign*cnv[i] + " ");
        }
        else{

```

```

        System.out.println(sign*cnv[i] + "");
    }
}
if(cnv[n-1] > 1 || n > 1){
    System.out.println(" or");
    System.out.print(" ");
    if(cnv[n-1] > 1){
        cnv[n-1]=cnv[n-1]-1;
        cnv[n]=1;
        n++;
    }
    else{
        n--;
        cnv[n-1]=cnv[n-1]+1;
    }
    for(int i=0;i<n;i++){
        if(i != (n-1)){
            System.out.print(sign*cnv[i] + " ");
        }
        else{
            System.out.println(sign*cnv[i] + "");
        }
    }
}
System.out.println(" gcd=" + gcd);
}
}
}

```

8.2 The Continued Fraction Representation of a Real Number

Suppose x is a real positive number. We shall compute the continued fraction for x . We shall later show that the convergents of the continued fraction converge to x ,

$$x = [a_1, a_2, a_3, \dots]$$

Let us write

$$x_0 = x$$

$$x_1 = 1/(x_0 - a_1),$$

where

$$a_1 = [x_0],$$

is the greatest integer in x_0 . So that

$$x_0 = a_1 + 1/x_1.$$

Next we expand x_1 in the same way, and continuing in this way, we find

$$x_n = 1/(x_{n-1} - a_n)$$

So that

$$x_{n-1} = a_n + 1/x_n.$$

If we ignore

$$1/x_n,$$

then we get the n th convergent of the continued fraction representation of x ,

$$[a_1, a_2, a_3, \dots, a_n],$$

which in general is an approximation to x . If x is irrational, the continued fraction will be infinite, for otherwise x would be rational.

Examples.

$$\pi = 3.1415926535897932384626433832795\dots$$

$$\pi = [3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 2, 1, 1, 2, 2, 2, 2, 1, 84, 2, \dots]$$

$$e = 2.7182818284590452353602874713527\dots$$

$$e = [2, 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, 10, 1, 1, 12, \dots]$$

The program **cfxr.java** computes the quotients of the continued fraction expansion of a real number.

```
//cfxr.java, continued fraction expansion of a real number. 12/9/07
//
class cfxr {
    public static void main(String args[]) {
        int cnv[];

        int a1;
        int n;
        double x,x0,x1;
        cnv = new int[50];
        n=0;
        if(args.length <= 1){
            System.out.println(" Continued fraction expansion of a real number  x " );
            System.out.println(" x is the real number > 0, n is the number of desired quotients. " );
            System.out.println(" Usage: java cv x n " );
            return;
        }
        if(args.length > 1){
            //System.out.println(" The first argument is " + args[0]);

```

```

x=Double.parseDouble(args[0]);
//System.out.println(" The second argument is " + args[1]);
n=Integer.parseInt(args[1]);

System.out.println(" The partial quotients of " + x + " are:");
// algorithm:
// a_{i+1} = [x_i], i=0,1,2,3,4,..., where [x_i] is the greatest integer in x_i
// x0=x
// x_1 = 1/(x_0-a_1)
// So x_0 = a_1 + 1/x_1
// .....
// x_{n+1} = 1/(x_n - a_{n+1})
// So x_n = a_{n+1} + 1/x_{n+1}
//
// pi=3.1415926535897932384626433832795
// quotients of pi= 3 7 15 1 292 1 1 1 2 1 3 1 14 2 1 1 2 2 2 1 84 2
// e=2.7182818284590452353602874713527
// quotients of e= 2 1 2 1 1 4 1 1 6 1 1 8 1 1 10 1 1 12 .....
x0=x;

for(int i=0;i<n;i++){
    a1=(int)x0;
    x1=1/(x0-a1);
    cnv[i]=a1;
    x0=x1;
}
System.out.print(" [");
for(int i=0;i<n;i++){
    if(i != (n-1)){
        System.out.print(cnv[i] + " ");
    }
    else{
        System.out.println(cnv[i] + "]");
    }
}
}
}
}

```

8.3 Computing the Convergents

Define

$$p_0 = 1, q_0 = 0, p_1 = a_1, q_1 = 1.$$

The first convergent is

$$c_1 = a_1 = \frac{p_1}{q_1}.$$

The second convergent is

$$c_2 = a_1 + \frac{1}{a_2} = \frac{a_2 a_1 + 1}{a_2} = \frac{a_2 p_1 + p_0}{a_2 q_1 + q_0} = \frac{p_2}{q_2},$$

where

$$p_2 = a_2 a_1 + 1, q_2 = a_2.$$

The third convergent is

$$\begin{aligned} \frac{p_3}{q_3} = c_3 &= a_1 + \frac{1}{a_2 + \frac{1}{a_3}} \\ &= a_1 + \frac{a_3}{a_2 a_3 + 1} \\ &= \frac{a_1(a_2 a_3 + 1) + a_3}{a_2 a_3 + 1} \\ &= \frac{a_3(a_2 a_1 + 1) + a_1}{a_2 a_3 + 1} \\ &= \frac{a_3 p_2 + p_1}{a_3 q_2 + q_1}. \end{aligned}$$

Theorem. The n th convergent c_n of a simple continued fraction is given by

$$c_n = \frac{p_n}{q_n},$$

where p_n and q_n are defined iteratively by

$$p_n = a_n p_{n-1} + p_{n-2},$$

and

$$q_n = a_n q_{n-1} + q_{n-2},$$

with initial values

$$p_0 = 1, q_0 = 0, p_1 = a_1, q_1 = 1.$$

Proof.

Assume that for all $k < n$ the k th convergent is

$$c_k = \frac{p_k}{q_k},$$

where

$$p_k = a_k p_{k-1} + p_{k-2},$$

and

$$q_k = a_k q_{k-1} + q_{k-2}.$$

Then write

$$c_n = [a_1, a_2, \dots, a_n] = [a_1, a_2, \dots, a_{n-1} + (1/a_n)] = c'_{n-1},$$

which is the $n - 1$ convergent of the partial fraction with only the $n - 1$ quotient changed to

$$a'_{n-1} = a_{n-1} + (1/a_n).$$

So we can write

$$\begin{aligned} \frac{p'_{n-1}}{q'_{n-1}} &= \frac{a'_{n-1} p_{n-2} + p_{n-3}}{a'_{n-1} q_{n-2} + q_{n-3}} \\ &= \frac{(a_{n-1} + 1/a_n) p_{n-2} + p_{n-3}}{(a_{n-1} + 1/a_n) q_{n-2} + q_{n-3}} \\ &= \frac{(a_n a_{n-1} + 1) p_{n-2} + a_n p_{n-3}}{(a_n a_{n-1} + 1) q_{n-2} + a_n q_{n-3}}. \end{aligned}$$

The numerator is

$$\begin{aligned} &(a_n a_{n-1} + 1) p_{n-2} + a_n p_{n-3} \\ &= a_n (a_{n-1} p_{n-2} + p_{n-3}) + p_{n-3} \\ &= a_n p_{n-1} + p_{n-2}. \end{aligned}$$

Similarly, the denominator is

$$a_n q_{n-1} + q_{n-2}.$$

Hence

$$p_n = a_n p_{n-1} + p_{n-2},$$

and

$$q_n = a_n q_{n-1} + q_{n-2}.$$

The program **convergents.java** compute the convergents of a continued fraction.

```

//convergents.java, convergents of a continued fraction. 12/7/07
class convergents {
public static void main(String args[]) {
    int cnv[];
    int a;
    int a1=1;
    int a2=1;
    int p1=1;
    int p2=1;
    int p3=1;
    int q1=1;
    int q2=1;
    int q3=1;
    int n;
    int k;
    double v;
    cnv = new int[50];
    n=args.length;
    if(args.length < 1){
        System.out.println(" Convergents of a continued fraction " );
        System.out.println(" Usage: java convergents a1 a2 a3 a4 .... " );
        return;
    }
    for(int i=0;i<n;i++){
        cnv[i]=Integer.parseInt(args[i]);
    }
    for(int i=0;i<n;i++){
        a=cnv[i];
        k=i+1;
        if(i == 0){
            a1=a;
            p1=a1;
            q1=1;
            v=a1;
            System.out.print(" Convergent " + k);
            System.out.print(" [");
            for(int j=0;j<k;j++){
                if(j != (k-1)){
                    System.out.print(cnv[j] + " ");
                }
                else{
                    System.out.print(cnv[j] + "] ");
                }
            }
            System.out.println("= " + p1 + "/" + q1+ " = " + v);
        }
        if(i == 1){
            a2=a;
            p2=a2*a1+1;
            q2=a2;
            v=p2;
            v=v/q2;
            System.out.print(" Convergent " + k);
            System.out.print(" [");
            for(int j=0;j<k;j++){

```

```

    if(j != (k-1)){
        System.out.print(cnv[j] + " ");
    }
    else{
        System.out.print(cnv[j] + "]" );
    }
}
System.out.println(" = " + p2 + "/" + q2 + " = " + v);
}
if(i > 1){
    p3=a*p2+p1;
    q3=a*q2+q1;
    v=p3;
    v=v/q3;
    System.out.print(" Convergent " + k);
    System.out.print(" [");
    for(int j=0;j<k;j++){
        if(j != (k-1)){
            System.out.print(cnv[j] + " ");
        }
        else{
            System.out.print(cnv[j] + "]" );
        }
    }
    System.out.println(" = " + p3 + "/" + q3 + " = " + v);
    p1=p2;
    p2=p3;
    q1=q2;
    q2=q3;
}
}
}
}
}

```

8.4 Properties of the Convergents

Theorem. For all n

$$p_n q_{n-1} - q_n p_{n-1} = (-1)^n.$$

Proof. We shall prove this by induction. So first we have

$$p_1 = a_1, q_1 = 1,$$

$$p_2 = a_2 a_1 + 1, q_2 = a_2.$$

Thus

$$p_2 q_1 - q_2 p_1 = a_2 a_1 + 1 - a_2 a_1 = 1 = (-1)^2.$$

Now

$$p_n q_{n-1} - q_n p_{n-1}$$

$$\begin{aligned}
&= (a_n p_{n-1} + p_{n-2})q_{n-1} - (a_n q_{n-1} + q_{n-2})p_{n-1} \\
& p_{n-2}q_{n-1} - q_{n-2}p_{n-1} = -(-1)^{n-1} = (-1)^n.
\end{aligned}$$

Theorem. Convergents are in lowest terms.

Proof. We have

$$p_n q_{n-1} - q_n p_{n-1} = (-1)^n.$$

So if there is an integer $k > 1$ that divides both p_n and q_n , then it must divide $(-1)^n$, which it can't do.

Theorem. The convergents c_n and c_{n-1} satisfy

$$c_n - c_{n-1} = \frac{(-1)^n}{q_n q_{n-1}}.$$

Proof.

$$\begin{aligned}
c_n - c_{n-1} &= \frac{p_n}{q_n} - \frac{p_{n-1}}{q_{n-1}} \\
&= \frac{p_n q_{n-1} - q_n p_{n-1}}{q_n q_{n-1}} \\
&= \frac{(-1)^n}{q_n q_{n-1}}.
\end{aligned}$$

Theorem. The convergents c_n and c_{n-2} satisfy

$$c_n - c_{n-2} = \frac{a_n (-1)^{n-1}}{q_n q_{n-2}}.$$

Proof.

$$\begin{aligned}
c_n - c_{n-2} &= \frac{p_n}{q_n} - \frac{p_{n-2}}{q_{n-2}} \\
&= \frac{p_n q_{n-2} - q_n p_{n-2}}{q_n q_{n-2}} \\
&= \frac{(a_n p_{n-1} - p_{n-2})q_{n-2} - (a_n q_{n-1} - q_{n-2})q_{n-2}}{q_n q_{n-2}} \\
&= \frac{a_n (p_{n-1} q_{n-2} - q_{n-1} p_{n-2})}{q_n q_{n-2}}
\end{aligned}$$

$$= \frac{a_n(-1)^{n-1}}{q_n q_{n-2}}.$$

Theorem. Every even convergent is greater than its preceding convergent.

$$c_{2n} > c_{2n-1}.$$

Proof. We have

$$c_{2n} - c_{2n-1} = \frac{(-1)^{2n}}{q_{2n} q_{2n-1}} > 0.$$

Theorem. The even convergents form a decreasing sequence.

$$c_{2n} < c_{2n-2}.$$

Proof. We have

$$c_{2n} - c_{2n-2} = \frac{a_n(-1)^{2n-1}}{q_{2n} q_{2n-2}} = \frac{-1}{q_{2n} q_{2n-2}} < 0.$$

Theorem. The odd convergents form an increasing sequence.

$$c_{2n+1} > c_{2n-1}.$$

Proof. We have

$$c_{2n+1} - c_{2n-1} = \frac{a_n(-1)^{2n}}{q_{2n+1} q_{2n-1}} = \frac{1}{q_{2n+1} q_{2n-1}} > 0.$$

Theorem. Each odd convergent is less than each even convergent.

Proof. Let c_{2k-1} be an odd convergent and c_{2j} be an even convergent. Let n be greater than both k and j . We have

$$c_{2n-1} < c_{2n}.$$

Because the odd convergents are increasing

$$c_{2k-1} < c_{2n-1}.$$

Because the even convergents are decreasing

$$c_{2n} < c_{2j-1}.$$

So

$$c_{2k-1} < c_{2n-1} < c_{2n} < c_{2j}.$$

The odd convergents are an increasing sequence bounded above, so they converge to some number ℓ_o . Likewise the even convergents are a decreasing sequence bounded below so converge to a number ℓ_e .

Theorem. The sequence of convergents of an infinite simple continued fraction converges.

Proof. We have

$$c_{2n} - c_{2n-1} = \frac{(-1)^2}{q_{2n}q_{2n-1}} = \frac{1}{q_{2n}q_{2n-1}}.$$

The iterative formula for the q_k show that they increase without limit. Therefore

$$\lim_{n \rightarrow \infty} (c_{2n} - c_{2n-1}) = 0.$$

So

$$\ell_o = \ell_e.$$

Theorem. If $[a_1, a_2, \dots]$ is the continued fraction expansion for x , then the convergents of $[a_1, a_2, \dots]$ converge to x .

Proof. See Olds.

Theorem. A continued fraction has repeating quotients iff it is a quadratic surd (a root of a quadratic equation with integer coefficients).

Proof. See Olds.

8.5 Continued Fractions Bibliography

- [1]Olds C. D., **Continued Fractions**, Random House, New Mathematical Library, 1963.
- [2]Chrystal, G. **Algebra** ,Volume II, Dover reprint.
- [3]Wall H. S., **Analytic Theory of Continued Fractions**, Van Nostrand, 1948.
- [4]Hardy G. H., Wright E. M., **An Introduction to the Theory of Numbers**, 4th ed, Oxford, 1960.
- [5]Perron, Oskar, **Die Lehrer von den KettenBrüchen**, Teubner, 1963.
- [6]Davenport H., **The Higher Arithmetic**, Dover reprint.

9 Notes

See the following Wikipedia articles concerning phyllotaxis:

Alan Turing

Fibonacci Phyllotaxis

ReactionDiffusion Equations

The following is a quote from the Wikipedia Alan Turing article:

”Alan Turing worked from 1952 until his death in 1954 on mathematical biology, specifically morphogenesis. He published one paper on the subject called The Chemical Basis of Morphogenesis in 1952, putting forth the Turing hypothesis of pattern formation. His central interest in the field was understanding Fibonacci phyllotaxis, the existence of Fibonacci numbers in plant structures. He used reaction diffusion equations which are now central to the field of pattern formation. Later papers went unpublished until 1992 when Collected Works of A.M. Turing was published. His contribution is now considered to be a seminal piece of work in this field.”

10 Bibliography

[1]Huntley H. E., **The Devine Proportion**, Dover.

[2]Coxeter H. S. M., **Introduction to Geometry**, 2nd ed, John Wiley.

[3]Olds C. D., **Continued Fractions**, Random House, New Mathematical Library, 1963.

[4]Emery James D., **Continued Fractions**, Emery University Press, 2008.

<http://stem2.org/je/continuedfractions.pdf>

[5]Abelson, Harold and Sussman, Gerald J., **Structure and Interpretation of Computer Programs**, The MIT Press, McGraw-Hill, 1985.

[6] Press William H, Flannery Brian P, Teukolsky Saul A, Vetterling William T, **Numerical Recipes in C**, Cambridge University Press, 1988.

[7] Prusinkiewicz Przemyslaw, Lindenmayer Aristid, **The Algorithmic Beauty of Plants** Springer-Verlag, 1990. (available online free as pdf file, my copy is stored as *algorithmicbeautyofplants.pdf*)