セルフデベロップメントとプログラミング

和田英一
IIJ イノベーションインスティテュート
技術研究所

尾見半左右氏(富士通研究所初代所長 1901-1985)
の情報処理学会のコンピュータ博物館パイオニアのページ
http://museum.ipsj.or.jp/pioneer/omi.html

少にして学ばば即ち壮にして為すこと有り,
壮にして学ばば即ち老いて衰へず,
老いて学ばば即ち死して朽ちず　　(佐藤一斎 言志四録)

少而学 即壮而有為
壮而学 即老而不衰
老而学 即死而不朽

パソコンが手元にあると, 面白そうな話題が試せる.

Knuth の The Art of Computer Programming を読み, 訳し,
多くのアルゴリズムと問題をテストした.

お詫び

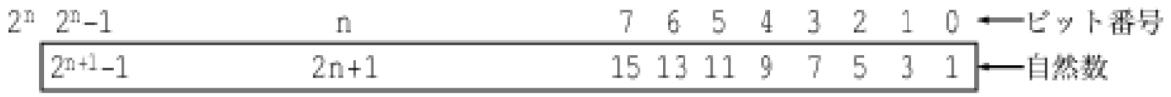
以下のプログラムはLisp言語の1つ, Schemeで書いてある.
私はプログラムのほとんどをLispとPostScriptでしか書かないから
なので, 済みません.

The Art of Computer Programmingは, Stanford大学の
Donald Knuthが1968年から書き続けているアルゴリズムの大著で,
この春4A巻が出た.

そのこだわりから, アルゴリズムは自然言語で書いていて,
曖昧なところがあり, 多くのことを書こうとするあまり,
説明をはしょっていて, 実装して走らせてみてやっと理解出来たりする.

Eratosthenes の篩

昔から知られている素数表の作り方

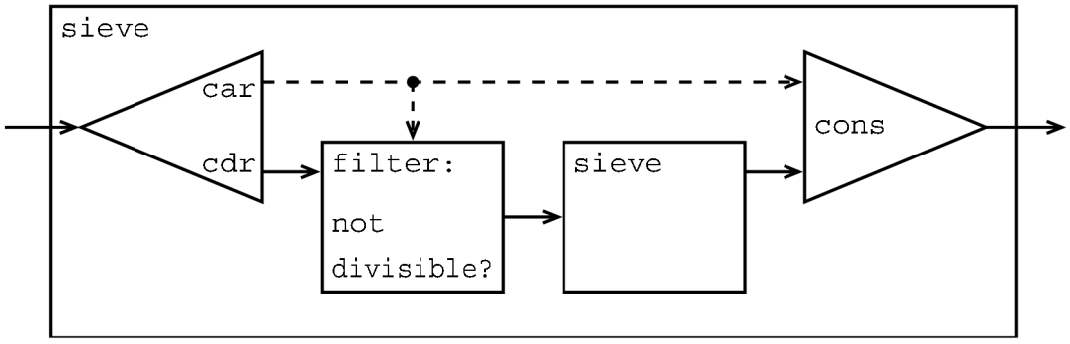


計算機で Eratosthenes の篩を計算するには, 0 から $2^n - 1$ の, 長さ 2^n で, 各ビットが 1 のビット列を用意し, 1 から $2^{n+1} - 1$ の奇数に対応させる.
1 は素数ではないので, 1 に対応するビットを 0 にする.
番号の小さい方から 1 のビットを探し, そのビット番号が n なら $p = 2n + 1$ が次の素数. そこから p 置きにビットを 0 にしていく.
p^2 より小さいビットは, $< p$ の素数で篩われているから, p^2 に対応するビット位置 $(p^2 - 1)/2$ 番目から p 置きに篩う.

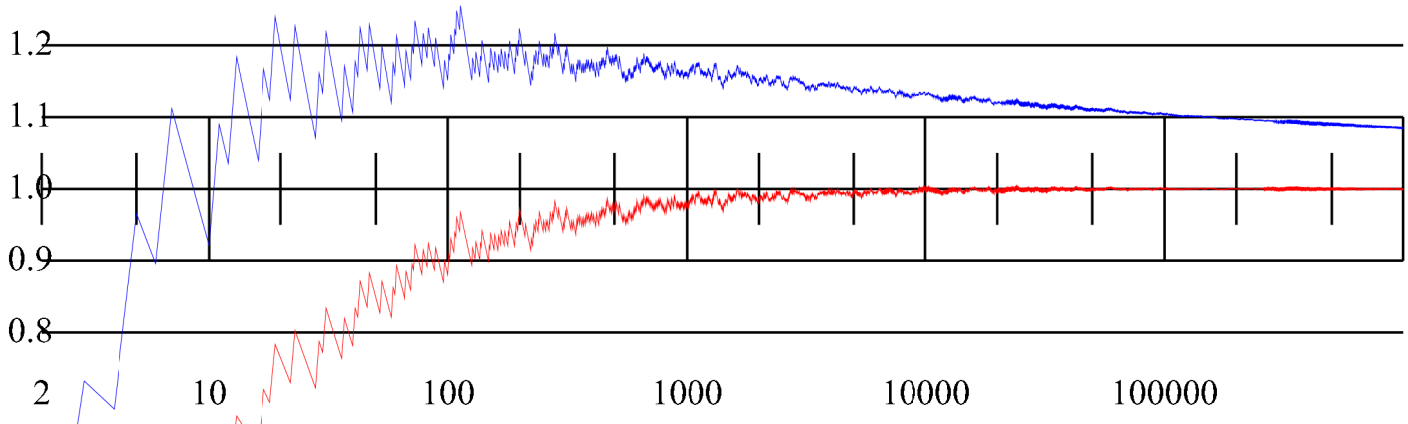
手作業では100くらいまで．パソコンでは，100万くらいまで出来る．

```
(define length 524288)  ;2^19
(define sieve (make-bit-string length #t))  ;篩を作る
(bit-string-clear! sieve 0)       ;1の場所をクリアする
(define k 0)
(define (next k)
 (let ((n (bit-substring-find-next-set-bit sieve
        (+ k 1) length)))
  (if n (let ((p (+ n n 1))) ;nの場所が1なら2n+1が素数
   (do ((q (/ (- (* p p) 1) 2) (+ q p))) ((>= q length))
    (bit-string-clear! sieve q)) (next n))
 'ok)))       ;(p*p-1)/2からp置きにクリア
(next 0)      ;起動
```

上の篩の実行に，MacBook Airで2.8秒くらいかかる．



ストリームの信号処理系の入れ子によるEratosthenesの篩
一番外の処理系には左から2,3,4,5,...が入る
右から2,3,5,7,11,..が出る



100万までの素数表があると, Gaussと Legendreの素数定理の図を描くことが出来る.

前のページの図は $\pi(n)/(n/\ln(n))$ をプロットしてある.
Gaussの式の値は

2の時 $1/(2/\ln(2))$=.34657359027997264
3の時 $2/(3/\ln(3))$=.73240819244540 64
4の時 $2/(4/\ln(4))$=.69314718055994 53
5の時 $3/(5/\ln(5))$=.96566274746046 01
6の時 $3/(6/\ln(6))$=.89587973461402 75
...
1000000の時 $78498/(1000000/\ln(1000000))$=1.08448994777907 95

Wilson の定理

$((x-1)!+1)/x$ は, x が素数か 1 の時に限り整数

Scheme には bignum があるのでやってみる.

```
(define (wilson x)
 (/ (+ (factorial (- x 1)) 1) x))
        ;Scheme の整数除算は整除できないと分数にする
(map wilson (a2b 2 20))
 ;(a2b a b) は a から b-1 までのリスト
(1 1 7/4 5 121/6 103 5041/8 40321/9 362881/10 329891
 39916801/12 36846277 6227020801/14 87178291201/15
 1307674368001/16 1230752346353 355687428096001/18
 336967037143579)
```


Carmichael 数 Fermat のテスト 素数 p で $a < p$ について $a^p \bmod p = p$.

Carmichael 数: 合成数なのに, Fermat テストを通過する

```
 4  1  0  1
 6  1  4  3  4  1
 8  1  0  1  0  1  0  1
 9  1  8  0  1  8  0  1  8
10  1  4  9  6  5  6  9  4  1
12  1  4  9  4  1  0  1  4  9  4  1
14  1  4  9  2 11  8  7  8 11  2  9  4  1
15  1  8 12  4  5  6 13  2  9 10 11  3  7 14
16  1  0  1  0  1  0  1  0  1  0  1  0  1  0  1
18  1 10  9 10  1  0  1 10  9 10  1  0  1 10  9 10  1
20  1 16  1 16  5 16  1 16  1  0  1 16  1 16  5 16  1 16  1
21  1  8  6  1 20  6  7  8 15 13  8  6 13 14 15  1 20 15 13 20
22  1  4  9 16  3 14  5 20 15 12 11 12 15 20  5 14  3 16  9  4  1
24  1 16  9 16  1  0  1 16  9 16  1  0  1 16  9 16  1  0  1 16  9 16  1
25  1  7 18 24  0  1  7 18 24  0  1  7 18 24  0  1  7 18 24  0  1  7 18 24
26  1  4  9 16 25 10 23 12  3 22 17 14 13 14 17 22  3 12 23 10 25 16  9  4  1
27  1 26  0  1 26  0  1 26  0  1 26  0  1 26  0  1 26  0  1 26  0  1 26  0  1 26
28  1 16 25  4  9  8 21  8  9  4 25 16  1  0  1 16 25  4  9  8 21  8  9  4 25 16  1
```

前ページの表は下のような latex の原稿で構成した

```
\verb+ 4 +\color{red}\verb+ 1+\color{black}\verb+  0  1 +\\
\verb+ 6 +\color{red}\verb+ 1+\color{black}\verb+  4 +
\color{red}\verb+ 3+\color{black}\verb+ +\color{red}\verb+ 4+
\color{black}\verb+  1 +\\
\verb+ 8 +\color{red}\verb+ 1+\color{black}\verb+  0  1  0  1  0  1 +\\
```

この原稿は次のページの Scheme のプログラムで作る.


```
(define (fermattest n a)
 (modulo (expt a (- n 0)) n))    ;a^(n-1)mod nはもう1つのFermatテスト

(define (disp2 n)
 (if (< n 10) (display " ")) (display n))

(define (dispred n)
 (display "+\\color\{red\}\\verb+")
 (disp2 n)
 (display "+\\color\{black\}\\verb+"))

(define (foo n)
 (if (> (length (factorize n)) 1)
  (begin (display "\\verb+")
   (disp2 n) (display " ")
   (do ((a 1 (+ a 1))) ((= a n))
    (let ((f (fermattest n a)))
     (if (= f a) (dispred f) (disp2 f)) (display " ")))
   (display "+\\\\") (newline))))

(do ((n 2 (+ n 1))) ((= n 30)) (foo n))
```

```
(map (lambda (a) (modulo (expt a 561) 561)) (a2b 1 561))
=> (1 2 3 4 5 6 7 8 9 10 ... 560)
```

Carmichael数には, 1105, 1729, 2465, 2821, 6601, ...
などがあるらしい.

このうち, 1729 は別のことで有名

Hardy の Ramanujan 追悼文 (Hardy 1921) から引用すると「『すべての正の整数は彼の友人であった』といったのは Littlewood だった (と思う.) 彼がパトニーで療養していた時, 見舞いにいったのを覚えている. 1729 番というタクシーに乗ったので, その数には何の面白味もないといい, それが悪い予兆でないことを望むといった. 『いや』彼は答えた『それは非常に興味のある数だ. それは二通りで二つの立方数の和で表現出来る最小の数だ』」.

$9^3 + 10^3 = 1^3 + 12^3 = 1729 \qquad 9^3 = 729, \quad 12^3 = 1728$

$x^2 + x + 41$ **の式と同じように気になる数である.**

Fourier 変換

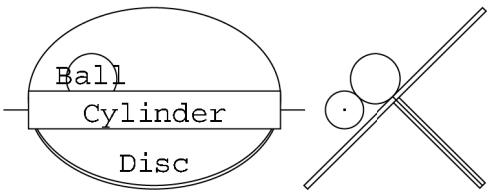
2π を周期とする周期関数 f を

$f(x) = a_0 + \Sigma_{n=1}^{N} a_n \cos nx + \Sigma_{n=1}^{N} b_n \sin nx$

のように sin や cos の和と表現したい時, 係数 a_n, b_n を計算する.

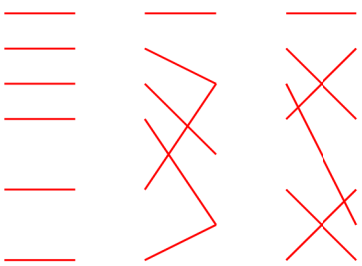
$$a_n = \frac{1}{N} \Sigma_m f\left(\frac{m\pi}{N}\right) \cos \frac{mn\pi}{N}$$

$$b_n = \frac{1}{N} \Sigma_m f\left(\frac{m\pi}{N}\right) \sin \frac{mn\pi}{N}$$



人力計算の場合

f(0)	cos(0π/N)
f(1)	cos(2π/N)
f(2)	cos(4π/N)
f(3)	cos(6π/N)
f(m)	cos(2mπ/N)
f(N-1)	cos(2(N-1)π/N)

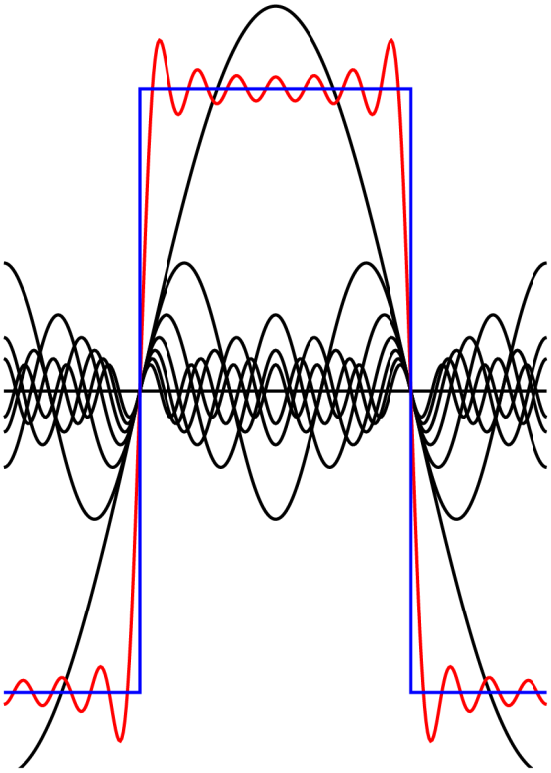


手で計算する時は, このような表を作る.
a_1 の計算は, 0 行目同士, 1 行目同士, ... を掛け合わせて足す.
a_2 の計算は, 0 行目と 0 行目, 1 行目と 2 行目, ... k 行目と 2k 行目を掛け合わせて足す.
a_3 の計算は, 0 行目と 0 行目, 1 行目と 3 行目, ... k 行目と 3k 行目を掛け合わせて足す.
のようにするが, 結構めんどうくさい.

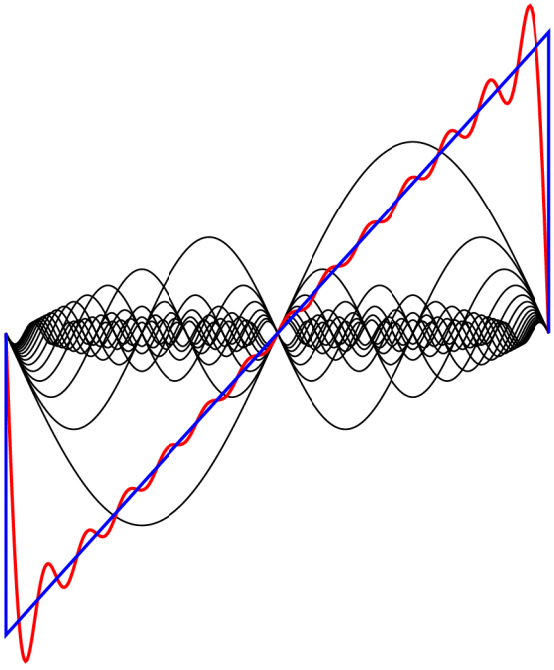
計算機があれば, すぐ試すことが出来る.

```
(define nn 64)

(define (a n)
 (let ((an 0))
  (do ((m 0 (+ m 1))) ((= m nn))
    (set! an (+ (* (f m) (cos (/ (* 2 pi (+ (* m n) 0.5)) nn))) an)))
   (/ an nn)))

(define (f n)     ; ステップ関数の時
 (cond ((< n (/ nn 4)) -1) ;1/4 より小さい時 -1
       ((>= n (/ (* nn 3) 4)) -1) ;3/4 より大きい時 -1
       (else 1)))          ; それ以外は 1

(define (f n)     ; 鋸歯状波関数
  (+ (/ (* 2 (+ n 0.5)) nn) -1))
```

ステップ関数の場合



鋸歯状波関数の場合



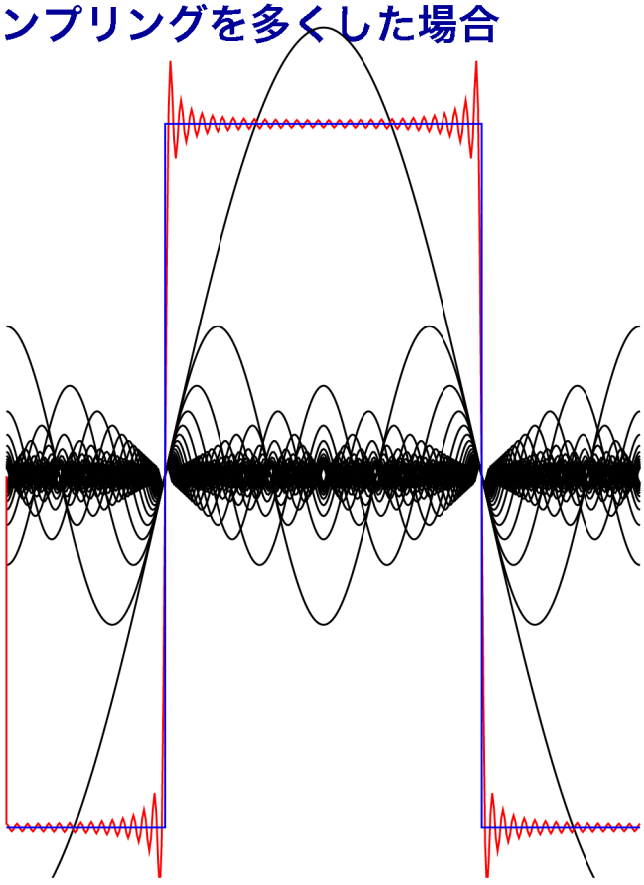
ステップ関数の係数

0
-.6368755077217536
0
.21194999078372936
0
-.12614008280347488
0
.08876606163859643
0
-.06752635118750022
0
.05360803812501698
0
-.04361835776624559
0
.03597089502453032

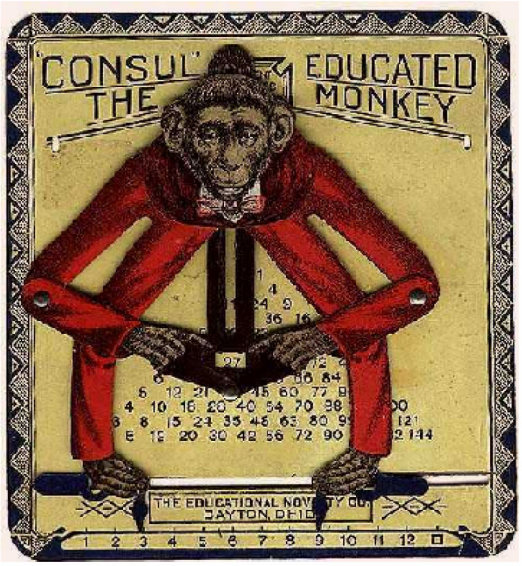
鋸歯状波関数の係数

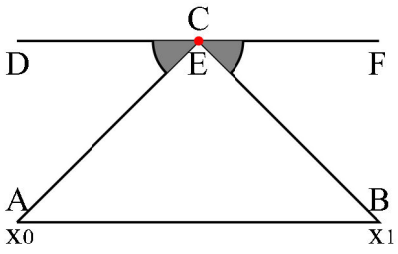
-.3184377538608768 6
-.1592188769304384 3
-.1059749953918646 5
-.0792242424353677 4
-.063070041401737 5
-.0522133599145449 1
-.0443830308192982 5
-.0384433314177275 5
-.0337631755937502 2
-.0299637897371944 56
-.0268040190625084 62
-.0241229798407529 3
-.0218091788831225 15
-.0197828665477294 76
-.0179854475122653 08

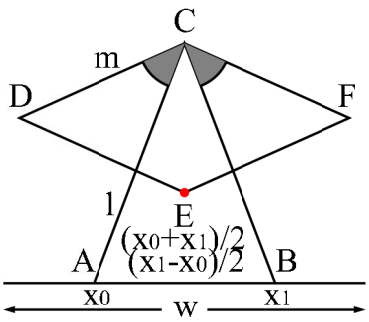
ステップ関数でサンプリングを多くした場合

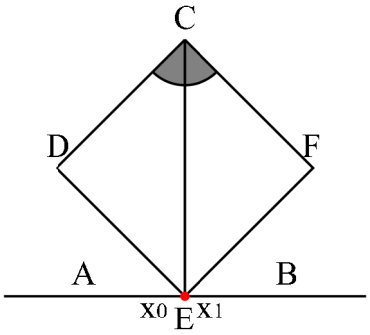


学者猿コンサル Consul the Educated Monkey

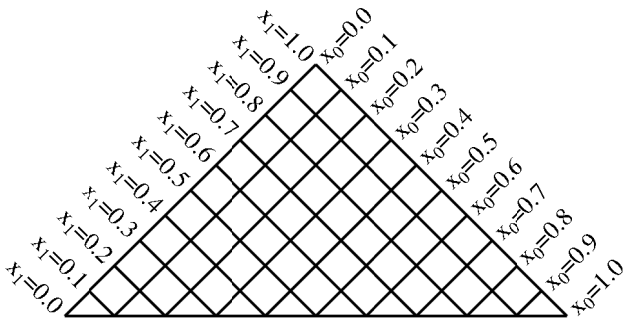




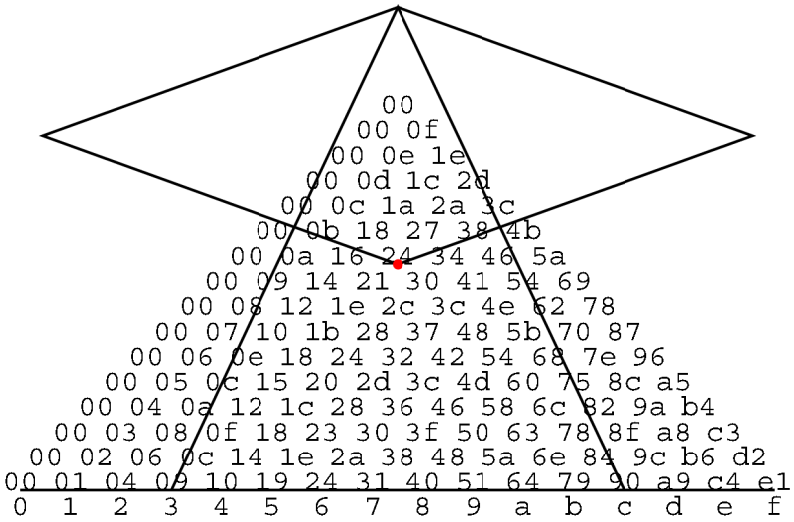




これは一種の計算図表



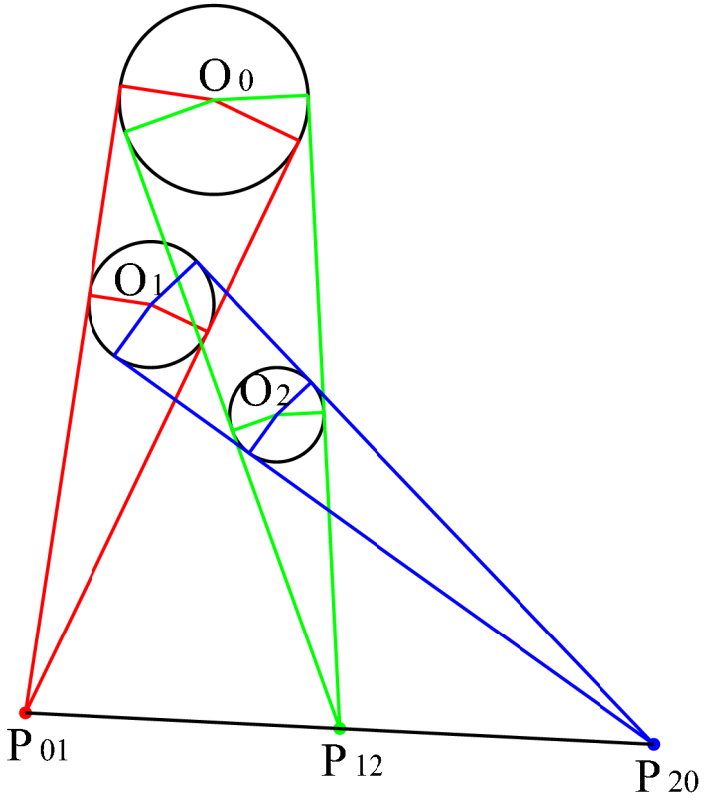
十六進の乗算表を作ることが出来る.

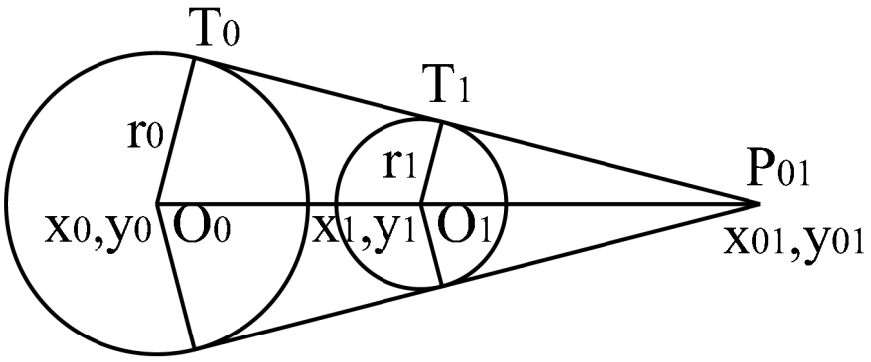


三つの円

平面上の２つの円の共通接線の交点を, それらの円の焦点という.

３つの円で出来る３つの焦点は一直線上にある.





$x_{01} = (r_0x_1 - r_1x_0)/(r_0 - r_1),\ y_{01} = (r_0y_1 - r_1y_0)/(r_0 - r_1)$

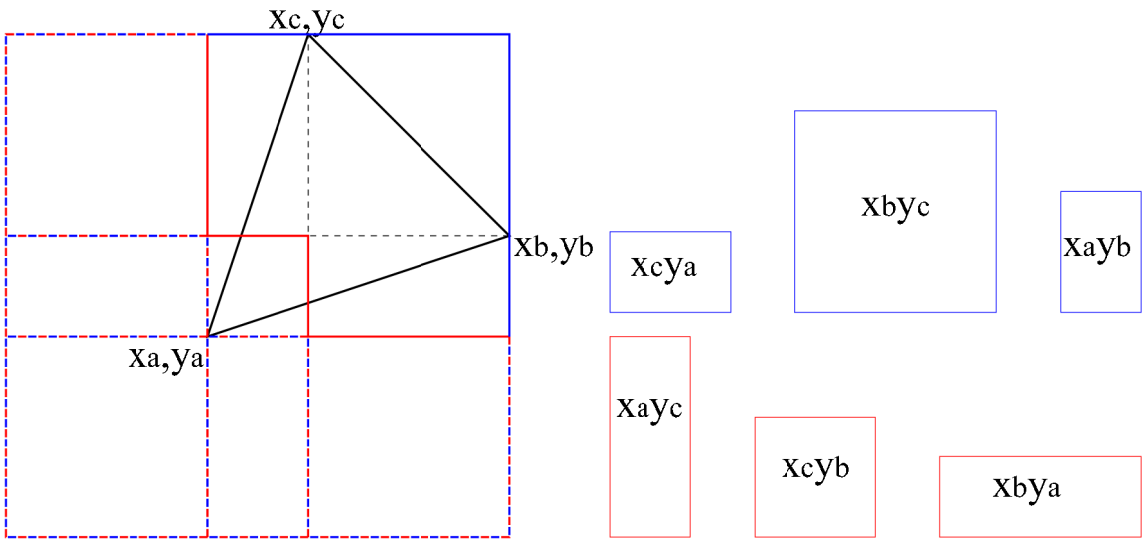
$x_{12} = (r_1x_2 - r_2x_1)/(r_1 - r_2),\ y_{12} = (r_1y_2 - r_2y_1)/(r_1 - r_2)$

$x_{20} = (r_2x_0 - r_0x_2)/(r_2 - r_0),\ y_{20} = (r_2y_0 - r_0y_2)/(r_2 - r_0)$

3点 (x_a, y_a), (x_b, y_b), (x_c, y_c) で構成する三角形の面積 S は

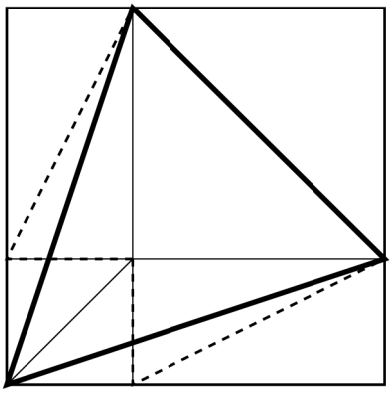
$$S = \frac{1}{2}\begin{vmatrix} x_a, y_a, 1 \\ x_b, y_b, 1 \\ x_c, y_c, 1 \end{vmatrix} \qquad \text{又は } S = \frac{1}{2}((x_b - x_a)(y_c - y_a) - (x_c - x_a)(y_b - y_a))$$

上の行列式の値は



右上の三角形に外接する欠けた四角の面積で,
その1/2が三角形の面積になる.

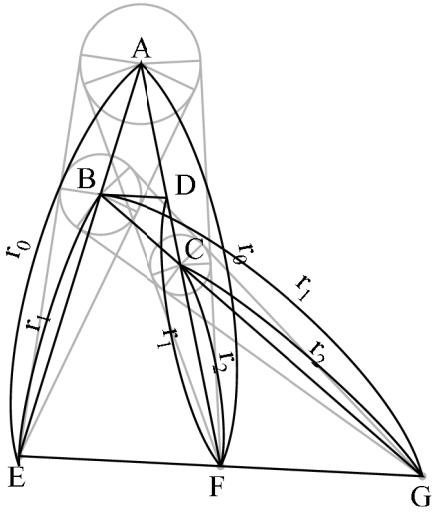
四角形と三角形の面積の関係



risa/asir では

```
X01=(r0*x1-r1*x0)/(r0-r1);
X12=(r1*x2-r2*x1)/(r1-r2);
X20=(r2*x0-r0*x2)/(r2-r0);

Y01=(r0*y1-r1*y0)/(r0-r1);
Y12=(r1*y2-r2*y1)/(r1-r2);
Y20=(r2*y0-r0*y2)/(r2-r0);

X01*Y12+X12*Y20+X20*Y01-X12*Y01-X20*Y12-X01*Y20;
```


幾何学的証明



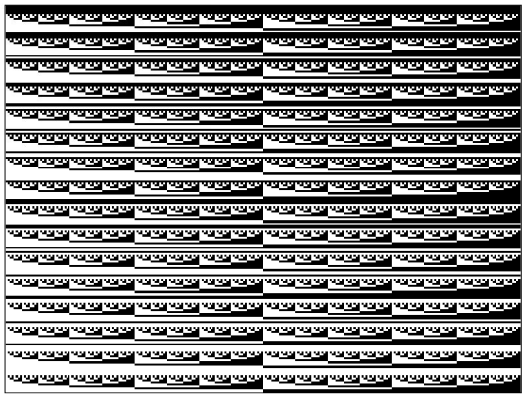
A,B,C の半径を r_0, r_1, r_2 とする.
$\frac{\mathrm{BE}}{\mathrm{AE}} = \frac{r_1}{r_0}$, $\frac{\mathrm{CF}}{\mathrm{AF}} = \frac{r_2}{r_0}$, $\frac{\mathrm{CG}}{\mathrm{BG}} = \frac{r_2}{r_1}$.
AF 上に D をとり, BD//EF とすると $\frac{\mathrm{DF}}{\mathrm{AF}} = \frac{r_1}{r_0}$.

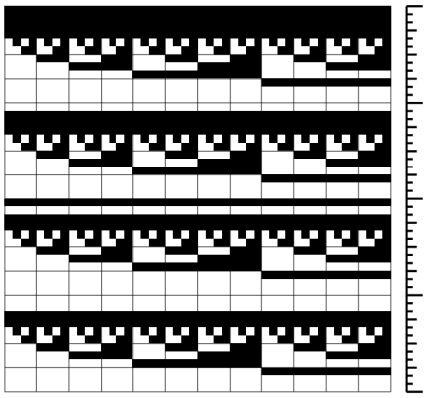
従って BD//FG QED.

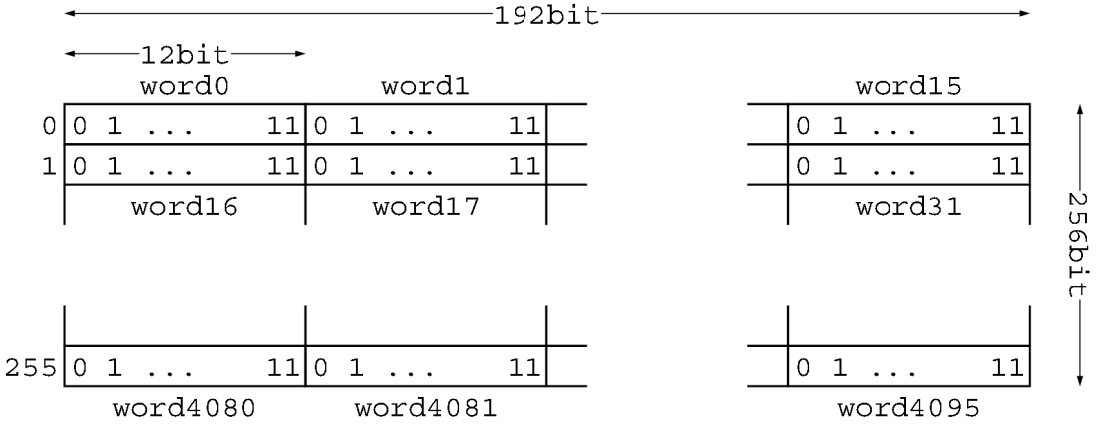
シミュレータ

PDP-8 のメモリーを表示するシミュレータ

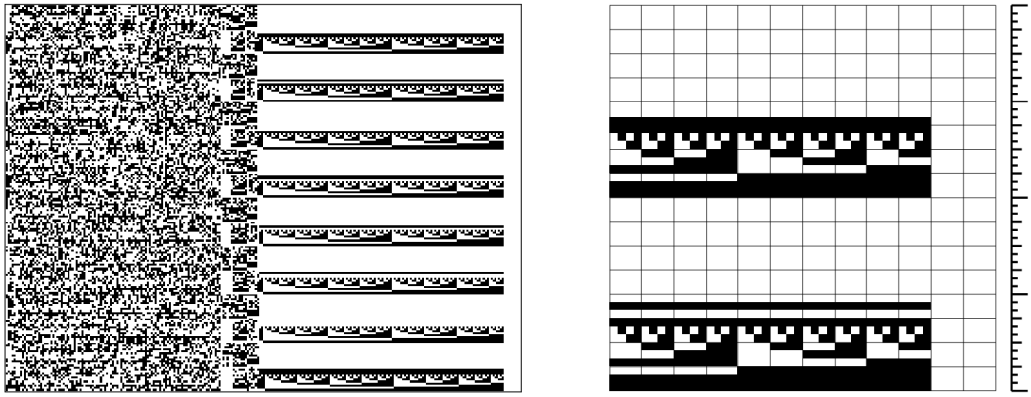
$0 \leq n < 4096$ の n 番地に n を書き込んだところ







van der Poel先生のPDP-8 Lispを読み込んだところ
左がプログラム領域, 右がヒープ領域



Programming should be fun.
Programs should be beautiful.