Számítógépes számelmélet

Járai Antal

Ezek a programok csak szemléltetésre szolgálnak

▼ 14. A szita módszerek alapjai

```
> restart; with(numtheory);
[GIgcd, bigomega, cfrac, cfracpol, cyclotomic, divisors, factorEQ, factorset,      (14.1)
```

fermat, *imagunit*, *index*, *integral_basis*, *invcfrac*, *invphi*, *issqrfree*, *jacobi*, *kronecker*, λ, *legendre*, *mcombine*, *mersenne*, *migcdex*, *minkowski*, *mipolys*, *mlog*, *mobius*, *mroot*, *msqrt*, *nearestp*, *nthconver*, *nthdenom*, *nthnumer*, *nthpow*, *order*, *pdexpand*, ϕ, π, *pprimroot*, *primroot*, *quadres*, *rootsunity*, *safeprime*, σ, *sq2factor*, *sum2sqr*, τ, *thue*]

14.1. Dixon véletlen négyzet módszere.

```
> n:=nextprime(7*10^2)*prevprime(14*10^2);
```

$$n := 980699 \tag{14.1.1}$$

```
> B:=20; F:=[];
  for j from 2 while j<B do
    if isprime(j) then F:=[op(F),j]; fi;
  od;
  F; nops(F);
```

$$B := 20$$

$$F := [\,]$$

$$[2, 3, 5, 7, 11, 13, 17, 19]$$

$$8 \tag{14.1.2}$$

```
> rnd:=rand(2..n-2); rnd(); ifactors(%^2 mod n);
```

$$rnd := \mathbf{proc}()$$

$$\quad \mathbf{proc}()$$

$$\quad\quad \mathbf{option}\ builtin;$$

$$\quad\quad 393$$

$$\quad \mathbf{end\ proc}(6, 980696, 20) + 2$$

$$\mathbf{end\ proc}$$

$$113502$$

$$[1, [[2, 2], [5, 1], [12097, 1]]] \tag{14.1.3}$$

```
> R:=[];
  while nops(R)<nops(F)+5 do
    x:=rnd();
    y:=ifactors(modp(x^2,n))[2];
    if y[nops(y)][1]>=B then next else R:=[op(R),[x,y]] fi;
  od:
```

$$R := [\,] \tag{14.1.4}$$

```
> R;
```

$$[[844857, [[2, 2], [5, 1], [7, 2], [13, 1], [17, 1]]], [384555, [[2, 1], [11, \tag{14.1.5}$$

1], [13, 2]]], [750169, [[2, 1], [5, 1], [11, 1], [19, 1]]], [257195, [[2, 9], [3, 1], [7, 1], [13, 1]]], [877424, [[2, 7], [3, 1], [5, 3], [13, 1]]], [220, [[2, 4], [5, 2], [11, 2]]], [908747, [[2, 1], [11, 3], [19, 2]]], [12778, [[2, 1], [5, 5], [7, 1], [11, 1]]], [121626, [[2, 6], [3, 2], [5, 1], [7, 1]]], [531172, [[2, 5], [3, 3], [5, 1], [7, 1], [17, 1]]], [453471, [[2, 9], [7, 1], [11, 1]]], [21475, [[3, 2], [5, 1], [17, 2], [19, 1]]], [855183, [[2, 4], [3, 4], [5, 1], [7, 2]]]]

```
> Rc:=[[21475, [[3, 2], [5, 1], [17, 2], [19, 1]]], [855183, [
  [2, 4], [3, 4], [5, 1], [7, 2]]], [164912, [[3, 1], [5, 2],
  [11, 1], [13, 1], [19, 1]]], [728436, [[2, 1], [3, 2], [11,
  1], [13, 1], [17, 1]]], [362222, [[3, 2], [19, 1]]], [297430,
  [[5, 1], [19, 4]]], [744161, [[3, 3], [5, 1], [11, 1], [13,
  1], [19, 1]]], [495370, [[2, 8], [3, 6], [5, 1]]], [577106, [
  [2, 1], [11, 1], [13, 2], [19, 1]]], [699549, [[7, 4]]],
  [689811, [[5, 4], [13, 1], [17, 1]]], [695704, [[3, 2], [5,
  1], [11, 4]]], [315384, [[2, 4], [3, 1], [5, 1], [11, 1],
  [13, 1], [19, 1]]]];
```

$$Rc := [[21475, [[3, 2], [5, 1], [17, 2], [19, 1]]], [855183, [[2, 4], [3, 4], [5, 1], [7, 2]]], [164912, [[3, 1], [5, 2], [11, 1], [13, 1], [19, 1]]], [728436, [[2, 1], [3, 2], [11, 1], [13, 1], [17, 1]]], [362222, [[3, 2], [19, 1]]], [297430, [[5, 1], [19, 4]]], [744161, [[3, 3], [5, 1], [11, 1], [13, 1], [19, 1]]], [495370, [[2, 8], [3, 6], [5, 1]]], [577106, [[2, 1], [11, 1], [13, 2], [19, 1]]], [699549, [[7, 4]]], [689811, [[5, 4], [13, 1], [17, 1]]], [695704, [[3, 2], [5, 1], [11, 4]]], [315384, [[2, 4], [3, 1], [5, 1], [11, 1], [13, 1], [19, 1]]]] \quad (14.1.6)$$

```
> with(linalg);
```

[*BlockDiagonal*, *GramSchmidt*, *JordanBlock*, *LUdecomp*, *QRdecomp*, *Wronskian*, *addcol*, *addrow*, *adj*, *adjoint*, *angle*, *augment*, *backsub*, *band*, *basis*, *bezout*, *blockmatrix*, *charmat*, *charpoly*, *cholesky*, *col*, *coldim*, *colspace*, *colspan*, *companion*, *concat*, *cond*, *copyinto*, *crossprod*, *curl*, *definite*, *delcols*, *delrows*, *det*, *diag*, *diverge*, *dotprod*, *eigenvals*, *eigenvalues*, *eigenvectors*, *eigenvects*, *entermatrix*, *equal*, *exponential*, *extend*, *ffgausselim*, *fibonacci*, *forwardsub*, *frobenius*, *gausselim*, *gaussjord*, *geneqns*, *genmatrix*, *grad*, *hadamard*, *hermite*, *hessian*, *hilbert*, *htranspose*, *ihermite*, *indexfunc*, *innerprod*, *intbasis*, *inverse*, *ismith*, *issimilar*, *iszero*, *jacobian*, *jordan*, *kernel*, *laplacian*, (14.1.7)

leastsqrs, linsolve, matadd, matrix, minor, minpoly, mulcol, mulrow, multiply, norm, normalize, nullspace, orthog, permanent, pivot, potential, randmatrix, randvector, rank, ratform, row, rowdim, rowspace, rowspan, rref, scalarmul, singularvals, smith, stackmatrix, submatrix, subvector, sumbasis, swapcol, swaprow, sylvester, toeplitz, trace, transpose, vandermonde, vecpotent, vectdim, vector, wronskian]

```
> RM:=matrix(nops(R),nops(F),0);
```

$$RM := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (14.1.8)$$

```
> for i to nops(Rc) do
    v:=Rc[i][2];
    for j to nops(v) do
      vv:=v[j];
      for k to nops(F) do if vv[1]=F[k] then RM[i,k]:=vv[2] fi
  od;
    od;
  od:
> print(RM);
```

$$\begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 2 & 1 \\ 4 & 4 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 1 & 0 & 1 \\ 1 & 2 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 4 \\ 0 & 3 & 1 & 0 & 1 & 1 & 0 & 1 \\ 8 & 6 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 2 & 0 & 1 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 1 & 1 & 0 \\ 0 & 2 & 1 & 0 & 4 & 0 & 0 & 0 \\ 4 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \quad (14.1.9)$$

```
> x:=Rc[10][1]; y:=7^2; x^2-y^2 mod n;
```

$$x := 699549$$

$$y := 49$$

$$0 \quad (14.1.10)$$

```
> igcd(n,x-y);
```

$$1399 \quad (14.1.11)$$

```
> RM:=addrow(RM,4,9,1);
```

(14.1.12)

$$RM := \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 2 & 1 \\ 4 & 4 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 1 & 0 & 1 \\ 1 & 2 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 4 \\ 0 & 3 & 1 & 0 & 1 & 1 & 0 & 1 \\ 8 & 6 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 2 & 3 & 1 & 1 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 1 & 1 & 0 \\ 0 & 2 & 1 & 0 & 4 & 0 & 0 & 0 \\ 4 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \quad (14.1.12)$$

```
> RM:=addrow(RM,3,7,1): RM:=addrow(RM,3,13,1);
```

$$RM := \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 2 & 1 \\ 4 & 4 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 1 & 0 & 1 \\ 1 & 2 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 4 \\ 0 & 4 & 3 & 0 & 2 & 2 & 0 & 2 \\ 8 & 6 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 2 & 3 & 1 & 1 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 1 & 1 & 0 \\ 0 & 2 & 1 & 0 & 4 & 0 & 0 & 0 \\ 4 & 2 & 3 & 0 & 2 & 2 & 0 & 2 \end{bmatrix} \quad (14.1.13)$$

```
> RM:=addrow(RM,2,1,1): RM:=addrow(RM,2,6,1): RM:=addrow(RM,2,
  7,1):
  RM:=addrow(RM,2,8,1): RM:=addrow(RM,2,12,1):RM:=addrow(RM,2,
```

```
13,1);
```

$$RM := \begin{bmatrix} 4 & 6 & 2 & 2 & 0 & 0 & 2 & 1 \\ 4 & 4 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 1 & 0 & 1 \\ 1 & 2 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\ 4 & 4 & 2 & 2 & 0 & 0 & 0 & 4 \\ 4 & 8 & 4 & 2 & 2 & 2 & 0 & 2 \\ 12 & 10 & 2 & 2 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 2 & 3 & 1 & 1 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 1 & 1 & 0 \\ 4 & 6 & 2 & 2 & 4 & 0 & 0 & 0 \\ 8 & 6 & 4 & 2 & 2 & 2 & 0 & 2 \end{bmatrix} \quad (14.1.14)$$

```
> RM:=addrow(RM,1,5,1): RM:=addrow(RM,1,9,1);
```

$$RM := \begin{bmatrix} 4 & 6 & 2 & 2 & 0 & 0 & 2 & 1 \\ 4 & 4 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 1 & 0 & 1 \\ 1 & 2 & 0 & 0 & 1 & 1 & 1 & 0 \\ 4 & 8 & 2 & 2 & 0 & 0 & 2 & 2 \\ 4 & 4 & 2 & 2 & 0 & 0 & 0 & 4 \\ 4 & 8 & 4 & 2 & 2 & 2 & 0 & 2 \\ 12 & 10 & 2 & 2 & 0 & 0 & 0 & 0 \\ 6 & 8 & 2 & 2 & 2 & 3 & 3 & 2 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 1 & 1 & 0 \\ 4 & 6 & 2 & 2 & 4 & 0 & 0 & 0 \\ 8 & 6 & 4 & 2 & 2 & 2 & 0 & 2 \end{bmatrix} \quad (14.1.15)$$

```
> x:=Rc[5][1]*(Rc[1][1]*Rc[2][1]) mod n;
```

$$x := 720723 \quad (14.1.16)$$

```
> y:=2^2*3^4*5*7*17*19 mod n;
```

$$y := 720723 \tag{14.1.17}$$

```
> x^2-y^2 mod n;
```

$$0 \tag{14.1.18}$$

```
> igcd(n,x-y);
```

$$980699 \tag{14.1.19}$$

```
> interface(echo=3);
```

$$1 \tag{14.1.20}$$

```
> read("../../old/factor/factor");
> with(numtheory);
```

[*GIgcd*, *bigomega*, *cfrac*, *cfracpol*, *cyclotomic*, *divisors*, *factorEQ*, *factorset*, *fermat*, *imagunit*, *index*, *integral_basis*, *invcfrac*, *invphi*, *issqrfree*, *jacobi*, *kronecker*, λ, *legendre*, *mcombine*, *mersenne*, *migcdex*, *minkowski*, *mipolys*, *mlog*, *mobius*, *mroot*, *msqrt*, *nearestp*, *nthconver*, *nthdenom*, *nthnumer*, *nthpow*, *order*, *pdexpand*, ϕ, π, *pprimroot*, *primroot*, *quadres*, *rootsunity*, *safeprime*, σ, *sq2factor*, *sum2sqr*, τ, *thue*]

```
>
#
# This procedure set up the factor base.
# The factor base is a list which
# always start with -1, has length N,
# and contains in incerasing order
# the primes p for which (n|p)=1.
#
>
> setfactorbase:=proc(n,N) local F,p;
> F:=[-1];
> p:=2;
> while nops(F)<N do
>           if jacobi(n,p)=1 then F:=[op(F),p]; fi;
>           p:=nextprime(p);
> od;
> F;
> end;
```

$$setfactorbase := \mathbf{proc}(n, N)$$

$$\quad \mathbf{local}\ F, p;$$

$$\quad F := [-1];$$

$$\quad p := 2;$$

$$\quad \mathbf{while}\ nops(F) < N\ \mathbf{do}$$

$$\qquad \mathbf{if}\ numtheory:-jacobi(n, p) = 1\ \mathbf{then}$$

```
            F := [op(F), p]
        end if;
        p := nextprime(p)
    end do;
    F
end proc
```

```
>
#
# The following procedure is a naive way to
# look for a full over a given factor base F
# starting with -1. Simple the number x
# is trial divided with
# the members of the factor base up to
# the end of the factor base. The result
# is a list with two members. The first
# is the unfactored part, the other a list.
# The members of this list are two-member lists
# containing index into the factor base and
# exponent of the given factor base member
# in the factorization.
#
>
> trialfull:=proc(x,F) local y,p,e,L,i;
> L:=[];
> y:=x;
> if y<0 then L:=[[1,1]]; y:=-y; fi;
> for i from 2 to nops(F) while y>1 do
>           p:=F[i];
>           if y mod p = 0 then
>                     e:=0;
>                     while y mod p = 0 do
>                               e:=e+1;
>                               y:=y/p;
>                     od;
>                     L:=[op(L),[i,e]];
>           fi;
> od;
> [y,L];
> end;
```

$$trialfull := \mathbf{proc}(x, F)$$

```
    local y, p, e, L, i;
    L := [ ];
    y := x;
```

```
    if y < 0 then
        L := [[1, 1]];
        y := -y
    end if;
    for i from 2 to nops(F) while 1 < y do
        p := F[i];
        if mod(y, p) = 0 then
            e := 0;
            while mod(y, p) = 0 do
                e := e + 1;
                y := y / p
            end do;
            L := [op(L), [i, e]]
        end if
    end do;
    [y, L]
end proc
```

```
>
#
# The following procedure is a naive way to
# find full's for a given composite
# number n over a factor base F
# starting with -1. Simple the numbers
# around sqrt(n) trial divided with
# the members of the factor base until
# N full factorizations have been found.
#
>
> trialfulls:=proc(n,F,N) local x,i,R,L;
> R:={};
> x:=floor(evalf(sqrt(n)));
> L:=trialfull(x^2-n,F);
> if L[1]=1 then R:={[x,L[2]]} fi;
> i:=1;
> while nops(R)<N do
>           L:=trialfull((x+i)^2-n,F);
>           if L[1]=1 then R:=R union {[x+i,L[2]]} fi;
>           if i>0 then i:=-i else i:=-i+1; fi;
> od;
> R;
```

```
> end;
```

```
trialfulls := proc(n, F, N)
    local x, i, R, L;
    R := {};
    x := floor(evalf(sqrt(n)));
    L := trialfull(x^2 - n, F);
    if L[1] = 1 then
        R := {[x, L[2]]}
    end if;
    i := 1;
    while nops(R) < N do
        L := trialfull((x + i)^2 - n, F);
        if L[1] = 1 then
            R := union(R, {[x + i, L[2]]})
        end if;
        if 0 < i then
            i := -i
        else
            i := -i + 1
        end if
    end do;
    R
end proc
```

```
>
#
# This procedure set up the equation system
# from the fulls. The cardinality N of the
# factor base have to given. R is the set
# of full relations.
#
>
> setequations:=proc(R,N) local T,L,LL;
> T:=array(N);
> LL:=[op(L)];
> for i to nops(LL) do
>
Error, on line 90, unexpected end of input
```

```
Error, while reading ``../../old/factor/factor``
> #
  # This procedure set up the factor base.
  # The factor base is a list which
  # always start with -1, has length N,
  # and contains in incerasing order
  # the primes p for which (n|p)=1.
  #

  setfactorbase:=proc(n,N) local F,p;
  F:=[-1]; p:=2;
  while nops(F)<N do
    if jacobi(n,p)=1 then F:=[op(F),p]; fi;
    p:=nextprime(p);
  od; F; end;
```

$$setfactorbase := \mathbf{proc}(n, N) \tag{14.1.21}$$

$$\begin{aligned}
&\mathbf{local}\ F, p;\\
&F := [-1];\\
&p := 2;\\
&\mathbf{while}\ nops(F) < N\ \mathbf{do}\\
&\quad \mathbf{if}\ numtheory:-jacobi(n, p) = 1\ \mathbf{then}\\
&\qquad F := [op(F), p]\\
&\quad \mathbf{end\ if};\\
&\quad p := nextprime(p)\\
&\mathbf{end\ do};\\
&F\\
&\mathbf{end\ proc}
\end{aligned}$$

```
> #
  # The following procedure is a naive way to
  # look for a full over a given factor base F
  # starting with -1. Simple the number x
  # is trial divided with
  # the members of the factor base up to
  # the end of the factor base. The result
  # is a list with two members. The first
  # is the unfactored part, the other a list.
  # The members of this list are two-member lists
  # containing index into the factor base and
  # exponent of the given factor base member
  # in the factorization.
  #
```

```
trialfull:=proc(x,F) local y,p,e,L,i;
L:=[]; y:=x;
if y<0 then L:=[[1,1]]; y:=-y; fi;
for i from 2 to nops(F) while y>1 do
  p:=F[i];
  if y mod p = 0 then e:=0;
    while y mod p = 0 do e:=e+1; y:=y/p; od;
    L:=[op(L),[i,e]];
  fi;
od; [y,L]; end;
```

$$trialfull := \mathbf{proc}(x, F) \tag{14.1.22}$$

$\mathbf{local}\ y, p, e, L, i;$

$L := [\,];$

$y := x;$

$\mathbf{if}\ y < 0\ \mathbf{then}$

$L := [[1, 1]];$

$y := -y$

$\mathbf{end\ if};$

$\mathbf{for}\ i\ \mathbf{from}\ 2\ \mathbf{to}\ nops(F)\ \mathbf{while}\ 1 < y\ \mathbf{do}$

$p := F[i];$

$\mathbf{if}\ mod(y, p) = 0\ \mathbf{then}$

$e := 0;$

$\mathbf{while}\ mod(y, p) = 0\ \mathbf{do}$

$e := e + 1;$

$y := y / p$

$\mathbf{end\ do};$

$L := [op(L), [i, e]]$

$\mathbf{end\ if}$

$\mathbf{end\ do};$

$[y, L]$

$\mathbf{end\ proc}$

```
> #
  # The following procedure is a naive way to
  # find full's for a given composite
  # number n over a factor base F
  # starting with -1. Simple the numbers
  # around sqrt(n) trial divided with
  # the members of the factor base until
```

```
   # N full factorizations have been found.
   #

   trialfulls:=proc(n,F,N) local x,i,R,L;
   R:={}; x:=floor(evalf(sqrt(n)));
   L:=trialfull(x^2-n,F);
   if L[1]=1 then R:={[x,L[2]]} fi;
   i:=1;
   while nops(R)<N do
     L:=trialfull((x+i)^2-n,F);
     if L[1]=1 then R:=R union {[x+i,L[2]]} fi;
     if i>0 then i:=-i else i:=-i+1; fi;
   od; R; end;
```

$$\begin{aligned}
&\mathit{trialfulls} := \mathbf{proc}(n, F, N) \\
&\quad \mathbf{local}\, x, i, R, L; \\
&\quad R := \{\}; \\
&\quad x := \mathrm{floor}(\mathit{evalf}(\mathrm{sqrt}(n))); \\
&\quad L := \mathit{trialfull}(x\text{^}2 - n, F); \\
&\quad \mathbf{if}\, L[1] = 1\, \mathbf{then} \\
&\qquad R := \{[x, L[2]]\} \\
&\quad \mathbf{end\ if}; \\
&\quad i := 1; \\
&\quad \mathbf{while}\, \mathit{nops}(R) < N \mathbf{do} \\
&\qquad L := \mathit{trialfull}((x+i)\text{^}2 - n, F); \\
&\qquad \mathbf{if}\, L[1] = 1\, \mathbf{then} \\
&\qquad\quad R := \mathit{union}(R, \{[x+i, L[2]]\}) \\
&\qquad \mathbf{end\ if}; \\
&\qquad \mathbf{if}\, 0 < i\, \mathbf{then} \\
&\qquad\quad i := -i \\
&\qquad \mathbf{else} \\
&\qquad\quad i := -i + 1 \\
&\qquad \mathbf{end\ if} \\
&\quad \mathbf{end\ do}; \\
&\quad R \\
&\mathbf{end\ proc}
\end{aligned} \tag{14.1.23}$$

```
> n; F:=setfactorbase(n,10);
```

$$980699$$

$$F := [-1, 5, 11, 23, 29, 31, 41, 43, 53, 59] \tag{14.1.24}$$

```
> trialfulls(n,F,11);
```

$$\{[1168, [[2, 2], [4, 2], [5, 1]]], [1532, [[2, 2], [6, 1], [7, 1], [8, 1]]], [532, [[1, 1], [2, 2], [3, 1], [8, 1], [10, 1]]], [928, [[1, 1], [2, 1], [3, 1], [7, 1], [9, 1]]], [768, [[1, 1], [2, 3], [9, 1], [10, 1]]], [1010, [[6, 2], [7, 1]]], [876, [[1, 1], [3, 2], [7, 1], [8, 1]]], [994, [[3, 1], [4, 1], [5, 1]]], [1122, [[2, 1], [4, 1], [7, 1], [10, 1]]], [912, [[1, 1], [2, 1], [6, 3]]], [948, [[1, 1], [2, 1], [4, 2], [6, 1]]]\} \tag{14.1.25}$$

▶ 14.2. Lánctörtek.

▶ 14.3. Kvadratikus irracionális számok lánctört alakja.

▼ 14.4. Faktorizálás lánctörtekkel.

```
>
```

▼ 14.5. Négyzetes szita.

```
> n:=nextprime(7*10^5)*prevprime(14*10^5);
```

$$n := 980000699999 \tag{14.5.1}$$

```
> n mod 8; n:=23*n; n mod 8;
```

$$7$$

$$n := 22540016099977$$

$$1 \tag{14.5.2}$$

```
> m:=2000; B:=50; T:=2.; b:=ceil(sqrt(n));
  F:=[[2,floor(0.5+2.*log[2.](2)),1]];
  for j from 3 while j<B do
    if isprime(j)=false then next fi;
    if jacobi(n,j)=1 then
      F:=[op(F),[j,floor(0.5+j/(j-1)*log[2.](j)),msqrt(n,j)]];
    fi;
  od;
  F; nops(F);
```

$$m := 2000$$

$$B := 50$$

$$T := 2.$$

$$b := 4747633$$

$$F := [[2, 2, 1]]$$

[[2, 2, 1], [3, 2, 1], [13, 4, 5], [19, 4, 6], [29, 5, 12], [31, 5, 12], [37, 5, 10]]

7 (14.5.3)

```
> S:= rtable(0..2*m,0);
```

$$S := \begin{bmatrix} 0 \,..\, 4000\ Array \\ Data\ Type: anything \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix} \quad (14.5.4)$$

```
> p:=F[1][1]; lp:=F[1][2]; x:=modp(m-b+F[1][3],p);
  while x<=2*m do S[x]:=S[x]+lp; x:=x+p; od:
```

$$p := 2$$
$$lp := 2$$
$$x := 0 \quad (14.5.5)$$

```
> for j from 2 to nops(F) do
    ppp:=F[j]; p:=ppp[1]; lp:=ppp[2];
    x:=modp(m-b+ppp[3],p);
    while x<=2*m do S[x]:=S[x]+lp; x:=x+p; od:
    x:=modp(m-b-ppp[3],p);
    while x<=2*m do S[x]:=S[x]+lp; x:=x+p; od:
  od:
> R:=[]; TT:=floor(log[2.](2*m*b)/T);
```

$$R := [\,]$$
$$TT := 17 \quad (14.5.6)$$

```
> for j from 0 to 2*m do if S[j]>=TT then R:=[op(R),j-m] fi od:
> R;
```

$$[-1800, -1104, -584, -432, -116, -2, 1366, 1506, 1714, 1734] \quad (14.5.7)$$

```
> map(y->ifactors((y+b)^2-n),R);
```

[[−1, [[2, 3], [3, 1], [19, 1], [29, 1], [31, 1], [71, 1], [587, 1]]], [−1, [[2, 3], [3, 1], [13, 1], [19, 1], [29, 1], [60953, 1]]], [−1, [[2, 3], [3, 1], [13, 1], [19, 1], [31, 1], [53, 1], [569, 1]]], [−1, [[2, 3], [3, 1], [19, 1], [29, 1], [37, 1], [8377, 1]]], [−1, [[2, 7], [3, 1], [13, 1], [19, 1], [37, 1], [313, 1]]], [−1, [[2, 3], [3, 2], [13, 1], [19, 1], [29, 1], [31, 1]]], [1, [[2, 3], [3, 2], [13, 1], [19, 1], [29, 1], [139, 1], [181, 1]]], [1, [[2, 5], [3, 1], [13, 1], [19, 1], [29, 1], [71, 1], [293, 1]]], [1, [[2, 6], [3, 1], [13, 1], [29, 1], [37, 1], [6079, 1]]], [1, [[2, 3], [3, 3], [19, 1], [31, 1], [37, 1], [3499, 1]]]] (14.5.8)

14.6. Többpolinomos négyzetes szita.

```
> n:=nextprime(7*10^7)*prevprime(14*10^7);
```

$$n := 9800003149999757 \tag{14.6.1}$$

```
> n mod 8; n:=37*n; n mod 8;
```

$$5$$

$$n := 362600116549991009$$

$$1 \tag{14.6.2}$$

```
> m:=10000; B:=100; T:=1.5;
  F:=[[2,floor(0.5+2.*log[2.](2)),1]];
  for j from 3 while j<B do
    if isprime(j)=false then next fi;
    if jacobi(n,j)=1 then
      F:=[op(F),[j,floor(0.5+j/(j-1)*log[2.](j)),msqrt(n,j)]];
    fi;
  od;
  F; nops(F);
```

$$m := 10000$$

$$B := 100$$

$$T := 1.5$$

$$F := [[2, 2, 1]]$$

$$[[2, 2, 1], [5, 3, 2], [7, 3, 2], [13, 4, 1], [19, 4, 7], [23, 5, 11], [29, 5, 11], [41, 5, 14], [47, 6, 4], [53, 6, 11], [59, 6, 22], [71, 6, 8], [97, 7, 67]]$$

$$13 \tag{14.6.3}$$

```
> dd:=floor((2*n/m^2)^(1/4.)): if type(dd,odd) then dd:=dd+1;
  fi:
  dd:=dd..dd;
```

$$dd := 292..292 \tag{14.6.4}$$

```
> if abs(op(1,dd)-(2*n/m^2)^(1/4.))<abs(op(2,dd)-(2*n/m^2)^
  (1/4.)) then
    d:=prevprime(op(1,dd));
    while modp(d,4)<>3 or jacobi(d,n)<>1 do
      d:=prevprime(d);
    od;
    dd:=d..op(2,dd);
  else
    d:=nextprime(op(2,dd));
    while modp(d,4)<>3 or jacobi(d,n)<>1 do
      d:=nextprime(d);
    od;
    dd:=op(1,dd)..d;
```

```
fi:
d; dd;
```

$$311$$

$$271..311 \tag{14.6.5}$$

```
> a:=d^2; h0:=n&^((d-3)/4) mod d;
```

$$a := 96721$$

$$h0 := 53 \tag{14.6.6}$$

```
> h1:=n*h0 mod d; (n-h1^2)/d; h2:=%*h0*((d+1)/2) mod d;
```

$$h1 := 223$$

$$1165916773472480$$

$$h2 := 25 \tag{14.6.7}$$

```
> b:=mods(h1+h2*d,a); b^2-n mod a; c:=(b^2-n)/a;
```

$$b := 7998$$

$$0$$

$$c := -3748928531405 \tag{14.6.8}$$

```
> S:= rtable(0..2*m,0); p:=F[1][1]; lp:=F[1][2]; x:=modp(m+(-b+
  F[1][3])/a,p);
  while x<=2*m do S[x]:=S[x]+lp; x:=x+p; od:
```

$$S := \begin{bmatrix} 0 .. 20000\ Array \\ Data\ Type: anything \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix}$$

$$p := 2$$

$$lp := 2$$

$$x := 1 \tag{14.6.9}$$

```
> for j from 2 to nops(F) do
    ppp:=F[j]; p:=ppp[1]; lp:=ppp[2];
    x:=modp(m+(-b+ppp[3])/a,p);
    while x<=2*m do S[x]:=S[x]+lp; x:=x+p; od:
    x:=modp(m+(-b-ppp[3])/a,p);
    while x<=2*m do S[x]:=S[x]+lp; x:=x+p; od:
  od:
> R:=[]; TT:=floor(log[2.](a*m^2/2.)/T);
```

$$R := [\,]$$

$$TT := 28 \tag{14.6.10}$$

```
> for j from 0 to 2*m do if S[j]>=TT then R:=[op(R),j-m] fi od:
> R;
```

(14.6.11)

[1215, 2625, 4024] (14.6.11)

```
> map(y->ifactors(a*y^2+2*b*y+c),R);
```

[[−1, [[2, 3], [5, 1], [7, 1], [19, 1], [41, 1], [47, 1], [53, 1], [6637, 1]]], (14.6.12)
[−1, [[2, 12], [5, 1], [23, 1], [37, 1], [47, 1], [53, 1], [71, 1]]], [−1,
[[5, 1], [7, 1], [19, 1], [23, 1], [47, 1], [53, 1], [59, 1], [971, 1]]]]

▶ 15. Számtest szita

▶ 16. Vegyes problémák