```
////////////////////////////////////////////////////////////
// CoCoA - Introduction


-- Semicolon at the end of the line!!
-- Everything but ring identerimetes starts with a  CAPITAL LETTER

-- The most used command: Quit;

-- Help  ?

?min

??min

////////////// CoCoA as a pocket computer
// Numbers and Polynomials


1+2+3+4+5+6+7+8+9;

9*10/2;

x^2+2xy+y^2;

(x+y)^2*(x-y)^2;

(2x+3y+4z)^15;

Bin(5,1);

Bin(3,0)+Bin(3,1)+Bin(3,2)+Bin(3,3)=2^3;

23/4;

Div(23,4);

Mod(23,4);

Div(23,4)*4+Mod(23,4);


P:=(x+y)^5;
Q:=(x-y)^4;
P+2Q^3;
```

```
E:=It;// It is the result of the last command


LT(E);
LC(E);
LM(E);
Time E:=(3x^2+2yz^2+x^2z)^2;
LT(E);
LC(E);
LM(E);

Log(xy^2z^5);
LogToTerm([1, 2, 5]);

xy^2z^5=LogToTerm(Log(xy^2z^5));

NR(x^10-1,[x^2-1]); // Poly division - reminder
NR(x^10-1,[x^3-1]);

DivAlg(x^10-1,[x^3-1]);// Poly division - all of it

DD:=DivAlg(x^10-1,[x^3-1]);DD;
Head(DD.Quotients)*(x^3-1)+DD.Remainder;

// Factorization - integers and polynomials

Fact(100);

Time Fact(100000);

Time A:=Fact(100000);

Time A:=Fact(500000);

ILogBase(A,10);

Time FloatStr(A);// more than two million digits



Factor(x^10-1);

Factor(x^248-1);


F:=4x^36y^26 - 4x^35y^27 + x^34y^28 + 20x^36y^25z - 20x^35y^26z +
```

```
5x^34y^27z - 4x^3y^2z^12 + 4x^2y^3z^12 - xy^4z^12 - 20x^3yz^13 +
20x^2y^2z^13 - 5xy^3z^13;

Factor(F);

LF:=Factor(F);

///////////////// Lists


NewList(10,3);

1..10;


L:=[1,2,3,4,5,6,7,8,9,10];L;

L1:=[x^I|I In 1..20];L1;

L2:=[[x^Iy^J|J In 1..5]|I In 1..10];// ERROR - Iy ambiguous

L2:=[[x^I*y^J|J In 1..5]|I In 1..10];L2; // Correct form

Flatten(L2);

L3:=[1,x,[1,5],"Pippo"];// Heterogeneous Lists

Len(L);
Head(L);
Tail(L);
First(L,5);
Last(L,5);
Max(L);
Min(L);
Reversed(L);
Reverse(L);L;
Sum(L);
Product(L);

CartesianProduct([1,2,3],[-1,1]); Also [1,2,3]><[-1,1];


Concat(First(L,2),Last(L,2));
```

```
Append(L,11);L;

Remove(L,Len(L));L;  // rpt


L:=[1,2,3,4,5,6,7,8,9,10];
WithoutNth(L,5);
L;


Permutations([1,2,3]);


[1,2,3,4,5]=[5,4,3,2,1];

EqSet([1,2,3,4,5],[5,4,3,2,1]);

MakeSet([1,2,3,2,3,2,3,2,1,2,3]);


[P| P In 1..100 And IsPrime(P)];

[P^2| P In 1..100 And IsPrime(P)];

LF;

[K[1]|K In LF];// avoiding multiplicities


Product(It); // Computing the sqfr part of F


LF:=Factor(y^255-1);


F:=Factor(y^253-1);
[Deg(T[1])|T In F];
[Len(T[1])|T In F];

F:=(x+3y-z^12xyz)^10(x-y+4)^7;
C:=Coefficients(F);C;
S:=Support(F);S;

ScalarProduct(S,C);

ScalarProduct(S,C)=F;
```

```
[FloatStr(K)|K In C];

Set Indentation;
[FloatStr(K)|K In C];

[FloatStr(K)|K In Sorted(C)];

Unset Indentation;


// Shape and Type

[IsPrime(K)|K In C]; // ERROR

Shape(C);

Type(1);

1+x-x;

Type(1+x-x);

[IsPrime(Cast(K,INT))|K In C];// ERROR - INT TOO BIG

[IsPPrime(Cast(K,INT))|K In C];// Correct - ISPPrime


// Programming Language

[Bin(5,K)|K In 0..5];  // By List constructor

[[Bin(5,K)|K In 0..5]| T In 0..6];

Set Indentation;

[[Bin(5,K)|K In 0..5]| T In 0..6];

Unset Indentation;

// By programming language
For N:=0 To 5 Do
  PrintLn [Bin(N,K)|K In 0..N];
EndFor;

Define  Khayyam(D) -- Pascal, Tartaglia, Yang Hui
```

```
  For N:=0 To 10 Do
    PrintLn [Bin(N,K)|K In 0..N];
  EndFor;
EndDefine;

Khayyam(10);

/*
Builds the Khayyam triangle
Example
D:=5;
Khayyam(D);
*/
Define  Khayyam(D)
  For N:=0 To 10 Do
    PrintLn [Bin(N,K)|K In 0..N];
  EndFor;
EndDefine;--Khayyam



Define Factorial(N)
  If N IsIn [0,1] Then
    Return 1
  Else
    Return N*Factorial(N-1);
  Endif;
EndDefine;

Factorial(5);


// Write a procedure for computing Euler Phi


// We need integer factorization


?factor

??factor

// Rough and ready procedure

??sqrt
```

```
??prime

/*
Factorize N=P1^A1*..*PN^AN as [[P1,A1],..,[PN,AN]]
Example
N:=5*49;
IFactor(N);
*/
Define IFactor(N)
  L:=[[I,FactorMultiplicity(N,I)]| I In [J|J In 2..N And IsPrime(J)] ];
  Return [K IN L|K[2]<>0];
EndDefine;


N:=987876565;
IFactor(N);

// Finding the problem by using the interpreter


// Solution
/*
Factorize N=P1^A1*..*PN^AN as [[P1,A1],..,[PN,AN]]
Example
N:=12;
IFactor(N);
//[[3, 1], [2, 2]]
N:=16;
IFactor(N);
//[[2, 4]]
N:=7;
IFactor(N);
//[[7, 1]]
*/
Define IFactor(N)
  P:=2;
  L:=[];
  Repeat
    If Mod(N,P)<>0 Then // DOES NOT DIVIDES
      P:=NextPPrime(P);
    Else // DIVIDES
     MP:=FactorMultiplicity(N,P);
     Append(L,[P,MP]);
     N:=N/P^MP;
    EndIf;
  Until N=1;
```

```
  Return L;
EndDefine;--IFactor



/*
Returns Phi(N)
Example
N:=7;
EulerPhi(N);
//6
N:=9;
EulerPhi(N);
//6
N:=15;
EulerPhi(N);
//8
N:=36;
EulerPhi(N);
//12
N:=8756385736;
*/
Define EulerPhi(N)
  L:=IFactor(N);
  L1:=[K[1]^K[2]-K[1]^(K[2]-1)| K In L];
  Return Product(L1);
EndDefine;

// Exercise: find how much time is spent in the procedure
//           FactorMultiplicity during the computation of
//           EulerPhi(362275200045)


/////// Scope

X:=1;

Define F(A)
  Return A+X;
EndDefine;


F(1);// Error!!!

Define F(A)
  X:=12;// Local X, not Global X
```

```
  Return A+X;
EndDefine;

F(1);
X;

Define Change(Var A,B)
  A:=B;
EndDefine;

X;
Change(X,2);// Complex way to assign 2 to X
X;


MEMORY.Counter:=1;

Define NextCounter()
  MEMORY.Counter:=MEMORY.Counter+1;
EndDefine;

NextCounter();
MEMORY.Counter;

// When MEMORY is useful - NOT OFTEN

IsDefined(MEMORY.VERBOSE);  // Can I use this name

MEMORY.VERBOSE:=True;


Define MyPrintLn(S)
  If MEMORY.VERBOSE Then
    PrintLn S;
  EndIf;
EndDefine;

Define IFactor(N)
  P:=2;
  L:=[];
  Repeat
    MyPrintLn(P);
    If Mod(N,P)<>0 Then // DOES NOT DIVIDES
      P:=NextPPrime(P);
    Else // DIVIDES
     MP:=FactorMultiplicity(N,P);
```

```
     Append(L,[P,MP]);
     N:=N/P^MP;
    EndIf;
  Until N=1;
  Return L;
EndDefine;--FactorInteger

IFactor(5675654);

MEMORY.VERBOSE:=False;

Time IFactor(5675654);

//Matrices - linear system solving

I:=Ideal(3x-2y-z+1,4x-2y-2z+5,4x-2y-6z-2);

A:=Mat([[3,-2,-1],[4,-2,-2],[4,-2,-6]]);
B:=Mat([[3,-2,-1,1],[4,-2,5,5],[4,-2,-6,-2]]);

Det(A);// One solution
Rank(A)=Rank(B);// One solution

C:=Mat([[-1],[-5],[2]]);
C:=ColMat([-1,-5,2]);C;// Easy way

Inverse(A)*C;// Finding the solution


Subst(I,[[x,-23/4],[y,-29/4],[z,-7/4]]);// Checking the solution

A^(-1);

3A^(-3)+2A-Identity(3);

A*B;

??mat

A;
B;
MatConcatHor(A,B);

Submat(It,[2],2..5);

////////////// Eigenvalues computations
```

```
Use R::=QQ[x];

A:=Mat([[3,-2,-1],[4,-2,-2],[4,-2,-6]]);
A-x*Identity(3);

P:=Det(It);
Factor(P);

RealRoots(P);// One real eigenvalue

RealRoots(P, 10^(-5));

DecimalStr(-3171043119/536870912);
DecimalStr(-6342086237/1073741824);

DecimalStr(-3171043119/536870912,20);
DecimalStr(-6342086237/1073741824,20);

Subst(P,[[x,-3171043119/536870912]]);
DecimalStr(It,20);


////////////// Rings declaration
Use R1::=QQ[l];// Rings

A-l*Identity(3);
F:=Det(It);
Factor(F);


Use R3::=ZZ/(3)[l];
F;
F:=BringIn(F);// Quick and easy - check Image for more
Factor(F);

// Find where -l^3 + l^2 + l + 1 is reducible
[P|P In 5..20 And IsPrime(P)];

Foreach P In [P|P In 5..20 And IsPrime(P)] Do
  RR::=ZZ/(P)[l];
  Using RR Do
    F1:=BringIn(F);
    PrintLn Factor(F1);
  EndUsing;
EndForeach;
```

```
// Problem with finites chars - bringingin in from Q

Use R::=QQ[l];
F:=-l^3 + l^2 + l + 1;
Foreach P In [P|P In 5..20 And IsPrime(P)] Do
  RR::=ZZ/(P)[l];
  Using RR Do
    F1:=BringIn(F);
    PrintLn [P,Factor(F1)];
  EndUsing;
EndForeach;


// Deg by Deg decomposition
Use R::=ZZ/(101)[x];
P:=x^33 - 2x^32 - 5x^31 + 5x^30 - 4x^29 + 4x^28 + x^27
   - 2x^26 - 5x^25 + x^24 + 3x^23 - 4x^21 - x^20 - 4x^19
   - x^18 - 4x^17 + 4x^16 + x^15 + 4x^14 + x^13 + 4x^12
   - 3x^10 - x^9 + 5x^8 + 2x^7 - x^6 - 4x^5 + 4x^4 - 5x^3
   + 5x^2 + 2x - 1;

Time GCD(P,x^(101^2)-x);
Time GCD(P,x^(101^3)-x);// Too slow


// Trick GCD(P,x^a-x) can be computed as
//       GCD(P,(x^a) mod P)-x);
Use R::=ZZ/(101)[x];
P:=x^33 - 2x^32 - 5x^31 + 5x^30 - 4x^29 + 4x^28 + x^27
   - 2x^26 - 5x^25 + x^24 + 3x^23 - 4x^21 - x^20 - 4x^19
   - x^18 - 4x^17 + 4x^16 + x^15 + 4x^14 + x^13 + 4x^12
   - 3x^10 - x^9 + 5x^8 + 2x^7 - x^6 - 4x^5 + 4x^4 - 5x^3
   + 5x^2 + 2x - 1;


/*
(Elem^Power) mod P
Example
Use R::=ZZ/(11)[x];
Elem:=x;
Power:=Characteristic()^4;
P:=x^7+x+1;
NR(Elem^Power,[P])=PowerQuot(Elem,Power,P);
//TRUE
*/
```

```
Define PowerQuot(Elem,Power,P)
  Y:=1;
  N:=Power;
  Z:=Elem;
  Repeat
    If IsOdd(N) Then
      Y:=NR(Y*Z,[P]);
    EndIf;
    N:=Div(N,2);
    If N<>0 Then
      Z:=NR(Z*Z,[P]);
    EndIf;
  Until N=0;
  Return Y;
EndDefine;


Time GCD(P,PowerQuot(x,101^2,P)-x);

Time GCD(P,PowerQuot(x,101^3,P)-x);

Time GCD(P,PowerQuot(x,101^6,P)-x);

Time GCD(P,PowerQuot(x,101^116,P)-x);


Time GCD(P,PowerQuot(x,101^999,P)-x);

// Exercise: find the deg by deg decomposition of P

// Exercise: find the deg by deg decomposition of
F:=x^23 + 2x^22 + 9x^21 + 4x^20 + 6x^19 - 9x^18 + 2x^17 + x^16 - 2x^15 + 11x^14
+ 5x^13 + 6x^12 + 3x^11 - 8x^10 - 6x^9 + 10x^8 + 4x^7 + 10x^6 - 2x^5 - x^4 -
10x^3 - 6x^2 + 7x - 3;
//in  ZZ/(23)[x];

// More about rings
Use R::=QQ[x,y,z];

Indets();
NumIndets();
Indet(2);
IndetIndex(y);

Use R::=QQ[x[0..4],a[-2..2]];
Indets();
```

```
Indet(6);
IndetIndex(a[-1]);

Use R::=QQ[x[0..2,0..4]];

M:=Mat([[x[I,J]|J In 0..4]|I In 0..2]);M;

Minors(3,M);
Minors(2,M);
Minors(1,M);
Rank(M);

M:=Mat([[x[I,J]|J In 0..2]|I In 0..2]);M;
Inverse(M);

Rank(M);


// Division - not canonical
Use R::=QQ[x,y,z];

L:=[x^2y-y^2,xy^2-x+1];

P:=x^7y^4-xyz-3;

NR(P,L);

NR(P,Reversed(L));



G:=GBasis(Ideal(L));G;


NR(P,G);

NR(P,Reversed(G));

NF(P,Ideal(L));


// Different term order, different GBasis
Use R::=QQ[x,y,z],Lex;
P:=x^7y^4-xyz-3;

L:=[x^2y-y^2,xy^2-x+1];
```

```
NF(P,Ideal(L));



// Exercise
Use R::=QQ[x,y,z];
P:=x^17y^4-xyz-3;
Q:=x^3y^2-x^3+y-2;
// compute NR(P,[Q]) WITHOUT using NR or NF or any GB command;



/////////////// GBasis
Use R::=QQ[x,y,z,t];

I:=Ideal(x^7-t^7,y^5-t^5,z^3-t^3);
GBasis(I);

Use R1::=QQ[x,y,z,t],Xel;
I1:=Ideal([BringIn(P)|P In Gens(I)]);// BringIn works with poly
GBasis(I1);

M:=Mat([[0,0,0,1],[1,1,1,0],[0,0,-1,0],[0,-1,0,0]]);M;
Use R2::=QQ[x,y,z,t],Ord(M);
I2:=Ideal([BringIn(P)|P In Gens(I)]);
GBasis(I2);

Use R;
Elim(t,I);


// tests
Use R::=QQ[t,x,y,z],Lex;

I:=Ideal(x^7-t^7,y^5-t^5,z^3-t^3);
ReducedGBasis(I);

J:=Ideal(ty^5 - z^6, -tz^6 + x^7, tx^7 - y^5z^3, x^7z^3 -y^10);
ReducedGBasis(J);


// Ideal Membership

NF(-t^5x + t^5z - t^3xy + xy^5 + xyz^3 - y^5z,I);
```

```
NF(-t^5x + t^5z - t^3xy + xy^5 + xyz^3 - y^5z,J);

// Ideal inclusion and equality
I<J;

I>J;

I=J;

// Reduction - easy case
//

//
// System solving by GB
// Linear systems
Use R::=QQ[x,y,z];
I:=Ideal(3x-2y-z+1,4x-2y-2z+5,4x-2y-6z-2);
ReducedGBasis(I);

Use R::=QQ[x,y,z,t];
I:=Ideal(x - 7/5y + 6/5z - t + 3/5, t^2 + 12/5y - 6/5z - t - 18/5, zt
+ 4/5y - 7/5z - t - 1/5, yt - 13/5y + 4/5z - 2t + 22/5, z^2 - 2/5y -
19/5z + 18/5, yz - 11/5y - 12/5z + 24/5, y^2 - 13/5y - 6/5z + 12/5);
ReducedGBasis(I);
LT(I);
QuotientBasis(I);


Use R3::=QQ[x,y,z,t],Lex;
I3:=Ideal([BringIn(P)|P In Gens(I)]);

ReducedGBasis(I3);

LT(I3);

QuotientBasis(I3);

Factor(t^3 - 4t^2 + 3t);G;
//[[t, 1], [t - 1, 1], [t - 3, 1]]

G:=ReducedGBasis(I3);
Subst(G,[[t,3]]);

// Exercise: find all the solutions
```

```
//// Ideal, modules and ideal ops



Use R::=QQ[x,y,z];
I:=Ideal(y-x^2);
J:=Ideal(x-y,x-z);
K:=Ideal(x-1,y-2,z-3)^2;

I+K;
I*K;
I*K:K;

IJK:=Intersection(I,J,K);IJK;
IJK:Ideal(x-1,y-2,z-3);

Colon(IJK,Ideal(x-1,y-2,z-3));

Saturation(IJK,Ideal(x-1,y-2,z-3));

IJK:Ideal(x-1,y-2,z-3)^2=Intersection(I,J);

// Intersection

Use R::=QQ[x,y,z,t],PosTo;

I:=Ideal(xy,yz,tx);
J:=Ideal(xy^2,z^2y);
Intersection(I,J);
I:Ideal(x);

// By Module
M:=Module([xy,0],[yz,0],[tx,0],[xy^2,xy^2],[z^2y,z^2y]);M;
ReducedGBasis(M);

/*
The part with 0 in the first coordinate gives the intersection
[Vector(xt, 0),
 Vector(yz, 0),
 Vector(xy, 0),
 Vector(0, yz^2),
 Vector(0, xy^2)]
*/

M:=Module([xy,0],[yz,0],[tx,0],[x,1]);M;
```

```
ReducedGBasis(M);
/*
// The part with 0 in the first coordinate gives the colon
[Vector(x, 1),
 Vector(yz, 0),
 Vector(0, t),
 Vector(0, y)]
*/

// By Elim
Use R1::=QQ[x,y,z,t,h];
I1:=Ideal([BringIn(P)|P In Gens(I)]);
J1:=Ideal([BringIn(P)|P In Gens(J)]);

// Intersection
H:=hI1+(1-h)J1;
Elim(h,H);

// Saturation
K:=I1+Ideal(1-xh);
Elim(h,K);

Saturation(I1,Ideal(x));

// GBasis - some classical benchmarks

Define Cycle(Var L)
  If L.CurrPos<Len(L.Inds) Then
    L.CurrPos:=L.CurrPos+1;
  Else
   L.CurrPos:=1
  EndIf;
  L.LastPos:=L.LastPos+1;
EndDefine;
Define Cyclic(N)
  L:=Record[Inds:=Tail(First(Indets(),N+1)),CurrPos:=1,LastPos:=1];
  GenIdeal:=[];
  For I:=1 To N-1 Do
    L.CurrPos:=I+1;
    L.LastPos:=1;
    NewTerm:=Product(First(L.Inds,I));
    StartingTerm:=NewTerm;
    GenDegI:=[];
    Repeat
      Append(GenDegI,NewTerm);
      NewTerm:=NewTerm*L.Inds[L.CurrPos]/L.Inds[L.LastPos];
```

```
      Cycle(L);
    Until NewTerm = StartingTerm;
    Append(GenIdeal,Sum(GenDegI));
  EndFor;
  Append(GenIdeal,1-Product(L.Inds));
  Return Ideal(GenIdeal);
EndDefine;

Use R::=ZZ/(32003)[k,a,b,c,d,e,f,g,h];

For N:=5 To 7 Do
  PrintLn "Cyclic  - hom",N;
PrintLn Time E1:=ReducedGBasis(Ideal([Homogenized(k,P)| P In Gens(Cyclic(N))]));
  PrintLn ;
EndFor;

// PrintLn Time E1:=ReducedGBasis(Ideal([Homogenized(k,P)| P In Gens(Cyclic(8))]));
// Too slow


Use R::=ZZ/(32003)[k,a,b,c,d,e,f,g,h];
Set Verbose;
PrintLn Time E1:=ReducedGBasis(Ideal([Homogenized(k,P)| P In Gens(Cyclic(8))]));

//Cpu time = 324.36, User time = 325

UnSet Verbose;

Use R::=QQ[k,a,b,c,d,e,f,g,h];
For N:=5 To 7 Do
  PrintLn "Cyclic - hom",N;
PrintLn Time E1:=ReducedGBasis(Ideal([Homogenized(k,P)| P In Gens(Cyclic(N))]));
  PrintLn ;
EndFor;


//Packages - how to use them

Aliases();


// Monomial Ideals
Use R::=ZZ/(2)[k,a,b,c,d,e,f,g,h];

I:=Ideal(abc,cde,fgh,hka);
```

```
Functions("$monomial_ideals");
Functions("$list");

$monomial_ideals.SqFrPrimaryDecomposition(I);

Aliases();

MonId.SqFrPrimaryDecomposition(I);

MonId.About();

MonId.Man();// Not all packages have the Man

$contrib/control.Man();// Some does


// The Latex Package
Use R::=QQ[x,y];
Latex((2x+4y^2-y^6)^3);
Latex(Ideal(2x+4y^2-y^6)^3);
M:=Mat([[1,2,3],[4,5,6],[7,8,9]]);M;
Latex(M);
M:=Mat([[1,2x,y3],[4,5y,6],[7x,8y,9]])^3;M;
Latex(M);

// Exercise: solve the diophantine equation with F,G poly of QQ[x,y,z]
// F*(x^2y-yz)+G*(x^2+y^2-3)=x^5 + x^3y^2 + x^3y - x^2y^2 -
//                           3x^3 - x^2y - y^3 - xyz + y^2z
//      - x^2 - y^2 + 3y + 3;
// using the package $contrib/control
// Hint: execute $contrib/control.Man(); and read
```