Grobner

RunJie Ma

September 11th 2018

1 Preparation

First we should try to do some preparations to finish our task. I will list something essential.

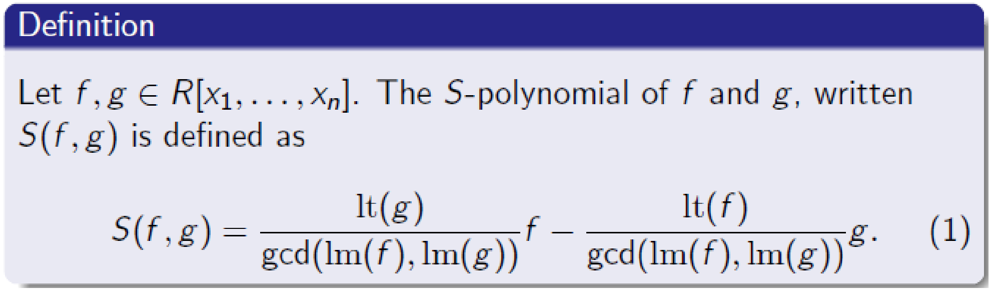
1.1 Factorize

Using factorize seems to be a most important thing in the work. Before using it, we should define a completely new ring in QQ(relational number). And when we define a polynomial in the ring, we should type the string type to avoid some conflicts.

Then we can normally get the result of factorization with the coefficient and the monomial.

1.2 S-Polynomials

I would like to give out the definition of S-polynomial again here, which is the same as the one in the slide. However, I think more detailed information seem to be better here.



Definition

Let $f, g \in R[x_1, \ldots, x_n]$. The S-polynomial of f and g, written $S(f, g)$ is defined as

$$S(f,g) = \frac{\mathrm{lt}(g)}{\gcd(\mathrm{lm}(f), \mathrm{lm}(g))} f - \frac{\mathrm{lt}(f)}{\gcd(\mathrm{lm}(f), \mathrm{lm}(g))} g. \quad (1)$$

The leading term and the leading monomial are almost the same, with the only difference is the coefficient. The only thing we should pay attention to is the order of variable will lead to the different lt and lm.

2 Exercise 1:

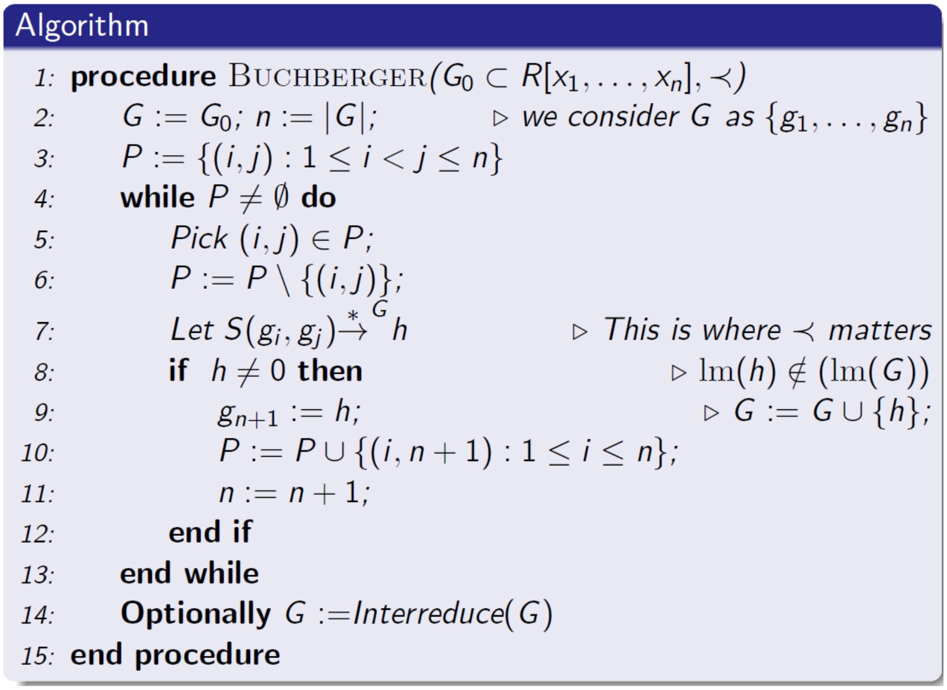
The Buchberger algorithm is designed to test if one polynomial is in the ideal I.

As the requirement, here we will emulate the process as using the sage to simulate what will happen when we call the Buchberger algorithm in the exercise 1.

The cyclic would be

$$\begin{cases} a+b+c \\ ab+bc+ac \\ abc-1 \end{cases}$$

Here I would like to use the order of $a > b > c$ and I will simulate the process.



Algorithm

```
procedure BUCHBERGER($G_0 \subset R[x_1, \dots, x_n], \prec$)
    $G := G_0$; $n := |G|$;            ▷ we consider $G$ as $\{g_1, \dots, g_n\}$
    $P := \{(i,j) : 1 \leq i < j \leq n\}$
    while $P \neq \emptyset$ do
        Pick $(i,j) \in P$;
        $P := P \setminus \{(i,j)\}$;
        Let $S(g_i, g_j) \overset{G}{\xrightarrow{*}} h$            ▷ This is where $\prec$ matters
        if $h \neq 0$ then            ▷ $\mathrm{lm}(h) \notin (\mathrm{lm}(G))$
            $g_{n+1} := h$;            ▷ $G := G \cup \{h\}$;
            $P := P \cup \{(i, n+1) : 1 \leq i \leq n\}$;
            $n := n+1$;
        end if
    end while
    Optionally $G :=$Interreduce($G$)
end procedure
```

First, we would pass the $a+b+c, ab+bc+ca, abc-1$, so the it is the G_0 and n = 3.
$P = (1,2),(1,3),(2,3)$
The first time we would like to pick (1,2), and then we would get $S(g_1, g_2)$ then. As we can see, the leading term of g_1 and g_2 are a and ab. And the same as the leading monomial. Here $S(g_1, g_2) = b^2 - ac$ but the leading monomial is divided by a so that we would continue the process to get the $h = b^2 + bc + c^2$ with leading term is b^2. Add (1,4),(2,4),(3,4) into P.
The second time we would like to pick (1,3), and then we would get $S(g_1, g_3)$ then. The leading term of g_1 and g_3 are a and abc. And the same as the leading monomial. Here $S(g_1, g_3) = c^3 - 1$with the reduce with the leading

term is c^3. Add (1,5),(2,5),(3,5),(4,5) into P.
The third time we would like to pick (2,3). However this time the S-Polynomial was reduced to 0 so that we would not add anything into the set.
The fourth time would like to pick (1,4). After long computation, we would get 1 finally. It can turn to zero finally.
The fifth time, pick (2,4), and we will get zero.
The sixth time, pick (3,4), and we will get zero.
The seventh time, pick (1,5), get zero.
The eighth time, pick (2,5), get zero.
The ninth time, pick (3,5), get zero.
The tenth time, pick (4,5), get zero.
Here the P is empty.

So now we should reduce the G by removing the g_2 and g_3 so we can get the result is $a + b + c, b^2 + bc + c^2, c^3 - 1$. And 6 of them has reduced to 0. It is the minimal base.

During the process, as you can see, we get 10 calculations of S-Polynomial eventually.

Using the criterion on slide 19, first we can see the gcd criterion, we can reduce 4 calculations which is the (1,4), (1,5), (2,5), (4,5). When we use the lcm criterion, the (2,3) is no more needed, so we can reduce one more. In total, the criterion helps us reduce five of them. A perfect job.

So gcd criterion has 4, and 1 for lcm criterion.

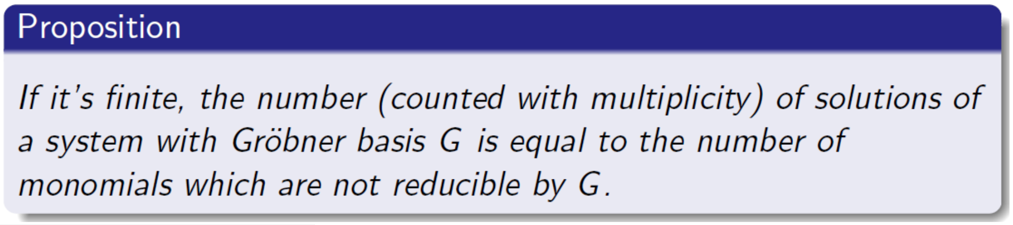
For the extra question, we can see that there should be at least three elements in the final set so that the example above is the least one. And here we can state that no matter what order we choose it is absolutely identical because the order doesn't matter. But the answer of the final Grobner Base is changed, however. But the amount stays unchanged. And in untitile5 I will show the result of changing the order and it would not change the amount.

3 Exercise 2:

$$L = \begin{cases} a + b + c + d + e \\ ab + bc + cd + de + ea \\ abc + bcd + cde + dea + eab \\ abcd + bcde + cdea + deab + eabc \\ abcde - 1 \end{cases}$$

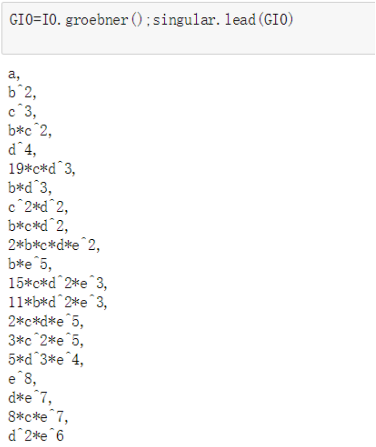
Above is the polynomial we would like to use and and we would like to show the statement we would like to use on the slide 19. And attention: the order we would like to use is $a < b < c < d < e$

3.1 Solutions Number



Proposition

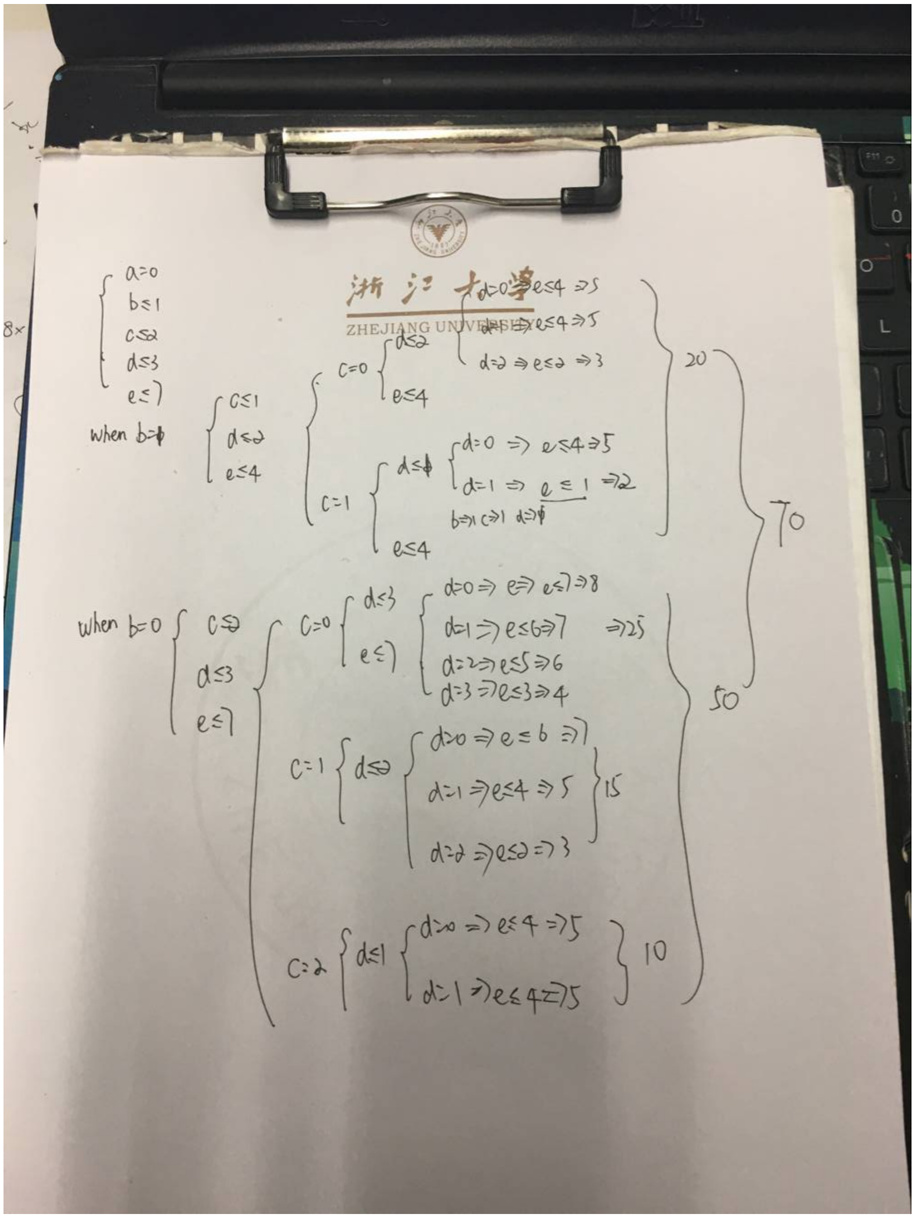
If it's finite, the number (counted with multiplicity) of solutions of a system with Gröbner basis G is equal to the number of monomials which are not reducible by G.



```
GI0=I0.groebner();singular.lead(GI0)
```

```
a,
b^2,
c^3,
b*c^2,
d^4,
19*c*d^3,
b*d^3,
c^2*d^2,
b*c*d^2,
2*b*c*d*e^2,
b*e^5,
15*c*d^2*e^3,
11*b*d^2*e^3,
2*c*d*e^5,
3*c^2*e^5,
5*d^3*e^4,
e^8,
d*e^7,
8*c*e^7,
d^2*e^6
```

Here for this question, we can discuss the situation for the different assignment of b,c,d,e. And we can use the adding theorem to get the total number of the solutions. And I will show the script I get on the answers. Here I will show the result of how dividing works.



浙江大学

ZHEJIANG UNIVERSITY

8×

$\begin{cases} a=0 \\ b\le 1 \\ c\le 2 \\ d\le 3 \\ e\le 7 \end{cases}$

when $b=\phi$ $\begin{cases} c\le 1 \\ d\le 2 \\ e\le 4 \end{cases}$

$c=0$ $\begin{cases} d\le 2 \\ e\le 4 \end{cases}$ $\begin{cases} d=0 \Rightarrow e\le 4 \Rightarrow 5 \\ d=1 \Rightarrow e\le 4 \Rightarrow 5 \\ d=2 \Rightarrow e\le 2 \Rightarrow 3 \end{cases}$

$c=1$ $\begin{cases} d\le \phi \\ e\le 4 \end{cases}$ $\begin{cases} d=0 \Rightarrow e\le 4 \Rightarrow 5 \\ d=1 \Rightarrow \underline{e\le 1} \Rightarrow 2 \end{cases}$

$b\Rightarrow 1$ $c\Rightarrow 1$ $d\Rightarrow \phi$

20

when $b=0$ $\begin{cases} c\le 2 \\ d\le 3 \\ e\le 7 \end{cases}$

$c=0$ $\begin{cases} d\le 3 \\ e\le 7 \end{cases}$ $\begin{cases} d=0 \Rightarrow e\Rightarrow e\le 7 \Rightarrow 8 \\ d=1 \Rightarrow e\le 6 \Rightarrow 7 \\ d=2 \Rightarrow e\le 5 \Rightarrow 6 \\ d=3 \Rightarrow e\le 3 \Rightarrow 4 \end{cases}$ $\Rightarrow 25$

$c=1$ $\{ d\le 2$ $\begin{cases} d=0 \Rightarrow e\le 6 \Rightarrow 7 \\ d=1 \Rightarrow e\le 4 \Rightarrow 5 \\ d=2 \Rightarrow e\le 2 \Rightarrow 3 \end{cases}$ 15

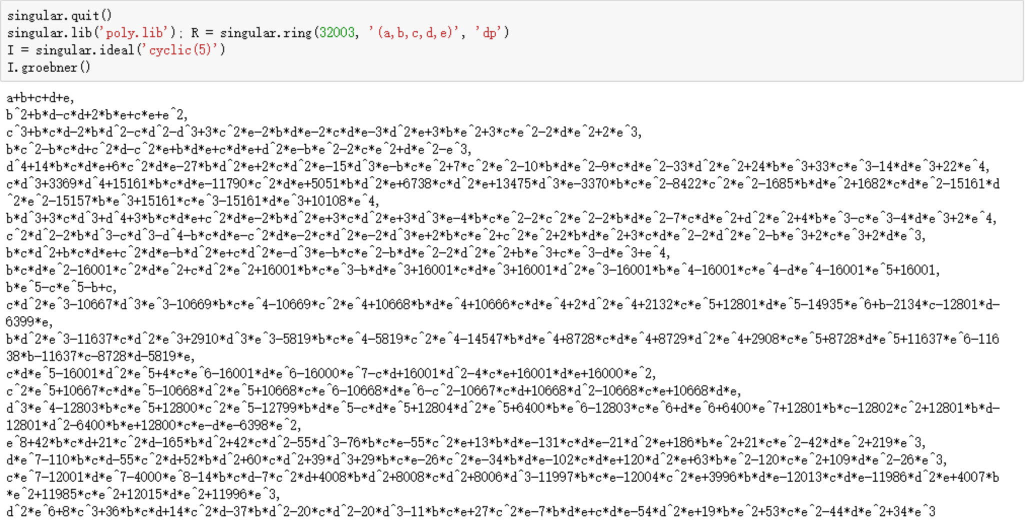
$c=2$ $\{ d\le 1$ $\begin{cases} d=0 \Rightarrow e\le 4 \Rightarrow 5 \\ d=1 \Rightarrow e\le 4 \Rightarrow 5 \end{cases}$ 10

50

70

So the total solutions of the polynomials are 70.

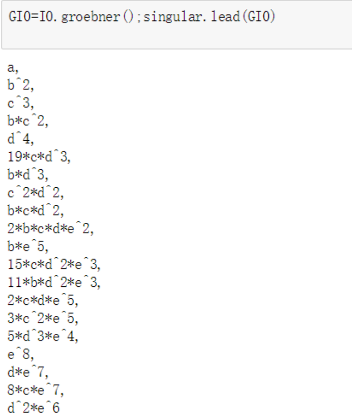
Here I would like to use the sage to compute the Grobner Base for the polynomials.



```
singular.quit()
singular.lib('poly.lib'); R = singular.ring(32003, '(a,b,c,d,e)', 'dp')
I = singular.ideal('cyclic(5)')
I.groebner()
```

```
a+b+c+d+e,
b^2+b*d-c*d+2*b*e+c*e+e^2,
c^3+b*c*d-2*b*d^2-c*d^2-d^3+3*c^2*e-2*b*d*e-2*c*d*e-3*d^2*e+3*b*e^2+3*c*e^2-2*d*e^2+2*e^3,
b*c^2-b*c*d+c^2*d-c^2*e+b*d*e+c*d*e+d^2*e-b*e^2-2*c*e^2+d*e^2-e^3,
d^4+14*b*c*d*e+6*c^2*d*e-27*b*d^2*e+2*c*d^2*e-15*d^3*e-b*c*e^2+7*c^2*e^2-10*b*d*e^2-9*c*d*e^2-33*d^2*e^2+24*b*e^3+33*c*e^3-14*d*e^3+22*e^4,
c*d^3+3369*d^4+15161*b*c*d*e-11790*c^2*d*e+5051*b*d^2*e+6738*c*d^2*e+13475*d^3*e-3370*b*c*e^2-8422*c^2*e^2-1685*b*d*e^2+1682*c*d*e^2-15161*d
^2*e^2-15157*b*e^3+15161*c*e^3-15161*d*e^3+10108*e^4,
b*d^3+3*c*d^3+d^4+3*b*c*d*e+c^2*d*e-2*b*d^2*e+3*c*d^2*e+3*d^3*e-4*b*c*e^2-2*c^2*e^2-2*b*d*e^2-7*c*d*e^2+d^2*e^2+4*b*e^3-c*e^3-4*d*e^3+2*e^4,
c^2*d^2-2*b*d^3-c*d^3-d^4-b*c*d*e-c^2*d*e-2*c*d^2*e-2*d^3*e+2*b*c*e^2+c^2*e^2+2*b*d*e^2+3*c*d*e^2-2*d^2*e^2-b*e^3+2*c*e^3+2*d*e^3,
b*c*d^2+b*c*d*e+c^2*d*e-b*d^2*e+c*d^2*e-d^3*e-b*c*e^2-b*d*e^2-2*d^2*e^2+b*e^3+c*e^3-d*e^3+e^4,
b*c*d*e^2-16001*c^2*d*e^2+c*d^2*e^2+16001*b*c*e^3-b*d*e^3+16001*c*d*e^3+16001*d^2*e^3-16001*b*e^4-16001*c*e^4-d*e^4-16001*e^5+16001,
b*e^5-c*e^5-b+c,
c*d^2*e^3-10667*d^3*e^3-10669*b*c*e^4-10669*c^2*e^4+10668*b*d*e^4+10666*c*d*e^4+2*d^2*e^4+2132*c*e^5+12801*d*e^5-14935*e^6+b-2134*c-12801*d-
6399*e,
b*d^2*e^3-11637*c*d^2*e^3+2910*d^3*e^3-5819*b*c*e^4-5819*c^2*e^4-14547*b*d*e^4+8728*c*d*e^4+8729*d^2*e^4+2908*c*e^5+8728*d*e^5+11637*e^6-116
38*b-11637*c-8728*d-5819*e,
c*d*e^5-16001*d^2*e^5+4*c*e^6-16001*d*e^6-16000*e^7-c*d+16001*d^2-4*c*e+16001*d*e+16000*e^2,
c^2*e^5+10667*c*d*e^5-10668*d^2*e^5+10668*c*e^6-10668*d*e^6-c^2-10667*c*d+10668*d^2-10668*c*e+10668*d*e,
d^3*e^4-12803*b*c*e^5+12800*c^2*e^5-12799*b*d*e^5-c*d*e^5+12804*d^2*e^5+6400*b*e^6-12803*c*e^6+d*e^6+6400*e^7+12801*b*c-12802*c^2+12801*b*d-
12801*d^2-6400*b*e+12800*c*e-d*e-6398*e^2,
e^8+42*b*c*d+21*c^2*d-165*b*d^2+42*c*d^2-55*d^3-76*b*c*e-55*c^2*e+13*b*d*e-131*c*d*e-21*d^2*e+186*b*e^2+21*c*e^2-42*d*e^2+219*e^3,
d*e^7-110*b*c*d-55*c^2*d+52*b*d^2+60*c*d^2+39*d^3+29*b*c*e-26*c^2*e-34*b*d*e-102*c*d*e+120*d^2*e+63*b*e^2-120*c*e^2+109*d*e^2-26*e^3,
c*e^7-12001*d*e^7-4000*e^8-14*b*c*d-7*c^2*d+4008*b*d^2+8008*c*d^2+8006*d^3-11997*b*c*e-12004*c^2*e+3996*b*d*e-12013*c*d*e-11986*d^2*e+4007*b
*e^2+11985*c*e^2+12015*d*e^2+11996*e^3,
d^2*e^6+8*c^3+36*b*c*d+14*c^2*d-37*b*d^2-20*c*d^2-20*d^3-11*b*c*e+27*c^2*e-7*b*d*e+c*d*e-54*d^2*e+19*b*e^2+53*c*e^2-44*d*e^2+34*e^3
```

So here what we should pay attention to how many monomials can not be divided by the Grobner Base. We should pay attention to the leading monomial and think of the monomial in the term like $a^f b^g c^h d^j e^k$. For the above Grobner list, the leading monomials are listed as below.



```
GI0=I0.groebner();singular.lead(GI0)
```

```
a,
b^2,
c^3,
b*c^2,
d^4,
19*c*d^3,
b*d^3,
c^2*d^2,
b*c*d^2,
2*b*c*d*e^2,
b*e^5,
15*c*d^2*e^3,
11*b*d^2*e^3,
2*c*d*e^5,
3*c^2*e^5,
5*d^3*e^4,
e^8,
d*e^7,
8*c*e^7,
d^2*e^6
```

So here what we should pay attention is the restriction adding to the f, g, h, j, k

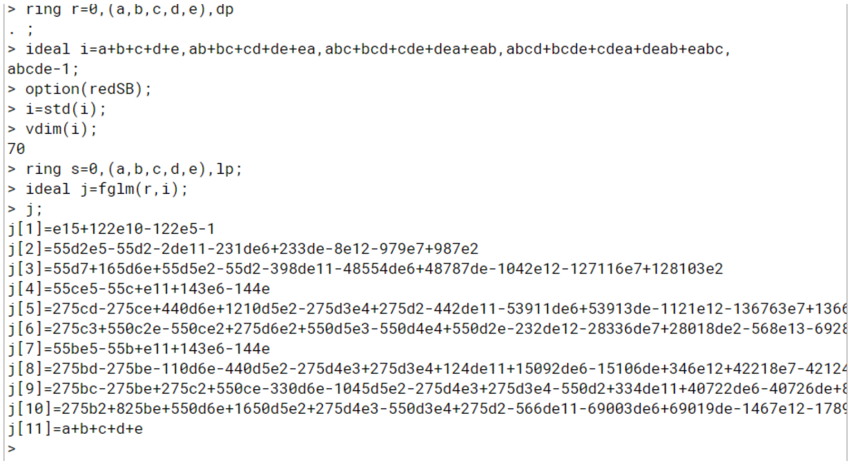
So here we can conclude that $f <= 0, g <= 1, h <= 2, j <= 3, k <= 1$ and lots of other restrictions. Here there are at least situations and we should check if state is available.
Once $b = 0$, then we can conclude that there are 3 doubles for the (c,d): (0,0-4), (1, 0-2), (2, 0-1) and plus the e there are 20 situations.

Once $b = 1 or 2$, then we can conclude that there are 15 situations by detailed discussing the situation about the assignment of d.

3.2 Faug‘ere–Gianni–Lazard–Mora algorithm

For this section, we would like to add an order into the Base. We have already mentioned above, that is $a > b > c > d > e$. For the request of the section, we have to use the purely lexicographical order, which is $a < b < c < d < e$

Here we can generally reproduce a ring in lp order so that we could get the answer of this question by inputting a new base and a new order so we can get the new basis. Here we would directly use the Singular Interface.



```
> ring r=0,(a,b,c,d,e),dp
. ;
> ideal i=a+b+c+d+e,ab+bc+cd+de+ea,abc+bcd+cde+dea+eab,abcd+bcde+cdea+deab+eabc,
abcde-1;
> option(redSB);
> i=std(i);
> vdim(i);
70
> ring s=0,(a,b,c,d,e),lp;
> ideal j=fglm(r,i);
> j;
j[1]=e15+122e10-122e5-1
j[2]=55d2e5-55d2-2de11-231de6+233de-8e12-979e7+987e2
j[3]=55d7+165d6e+55d5e2-55d2-398de11-48554de6+48787de-1042e12-127116e7+128103e2
j[4]=55ce5-55c+e11+143e6-144e
j[5]=275cd-275ce+440d6e+1210d5e2-275d3e4+275d2-442de11-53911de6+53913de-1121e12-136763e7+136
j[6]=275c3+550c2e-550ce2+275d6e2+550d5e3-550d4e4+550d2e-232de12-28336de7+28018de2-568e13-692
j[7]=55be5-55b+e11+143e6-144e
j[8]=275bd-275be-110d6e-440d5e2-275d4e3+275d3e4+124de11+15092de6-15106de+346e12+42218e7-4212
j[9]=275bc-275be+275c2+550ce-330d6e-1045d5e2-275d4e3+275d3e4-550d2+334de11+40722de6-40726de+
j[10]=275b2+825be+550d6e+1650d5e2+275d4e3-550d3e4+275d2-566de11-69003de6+69019de-1467e12-178
j[11]=a+b+c+d+e
>
```

And that is all.

3.3 Gianni–Kalkbrener theorem

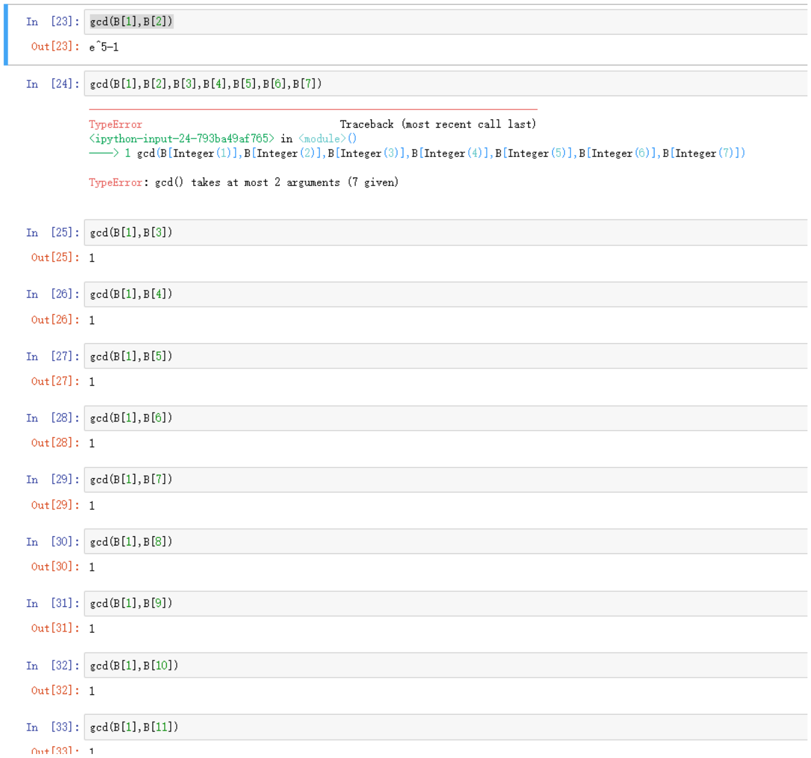
Attention: the actual work in this section is quite large.

The first thing we have to pay attention to is that to get the answer of the ideal is as same as the one to get the answer of the Grobner Base. So that to get the solution we can just focus on the Grobner Base. Then we can use the Gianni–Kalkbrener theorem to solve the resting question.

The aim of this section is to reduce the total answers by identifying some of the polynomials and then we can divide the total situation into different parts.

As we can see a only appears at only one polynomial so that we can deduce that once the other 4 vars have been identified, there can be no more situations.

Then we would start from e, after determine the e, we can take the answer to the one with d and e. (and there are 5 polynomials that can be divided by $e^5 - 1$)Then we can determine d. So we can find out all the answers. As the d is not only determined by only one polynomial so that what we should pay attention to find out the common root of these polynomials.



```
In  [23]: gcd(B[1],B[2])
 Out[23]: e^5-1

In  [24]: gcd(B[1],B[2],B[3],B[4],B[5],B[6],B[7])

          ---------------------------------------------------------------------------
          TypeError                                 Traceback (most recent call last)
          <ipython-input-24-793ba49af765> in <module>()
          ----> 1 gcd(B[Integer(1)],B[Integer(2)],B[Integer(3)],B[Integer(4)],B[Integer(5)],B[Integer(6)],B[Integer(7)])

          TypeError: gcd() takes at most 2 arguments (7 given)

In  [25]: gcd(B[1],B[3])
 Out[25]: 1

In  [26]: gcd(B[1],B[4])
 Out[26]: 1

In  [27]: gcd(B[1],B[5])
 Out[27]: 1

In  [28]: gcd(B[1],B[6])
 Out[28]: 1

In  [29]: gcd(B[1],B[7])
 Out[29]: 1

In  [30]: gcd(B[1],B[8])
 Out[30]: 1

In  [31]: gcd(B[1],B[9])
 Out[31]: 1

In  [32]: gcd(B[1],B[10])
 Out[32]: 1

In  [33]: gcd(B[1],B[11])
 Out[33]: 1
```

The graph above is about the initial process.
And we know the answer is that 70 in total. And in the worksheet I have factorized all the polynomials into different pieces. Here I would like to give

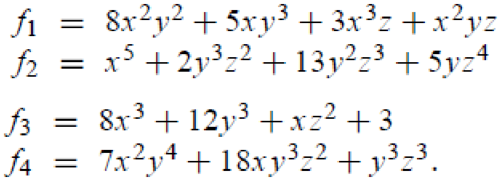
a solution. Case 1: if e is the solution of e^5-1, then first two polynomials can be eliminated. For d, the smallest degree is 7 and all 1 degree for the rest so that it is 5*7 = 35. Case 2: if e is the solution for the $e^4+e^3+6*e^2-4*e+1$, and then the min degree for d is 2. The other is still 1. So that 4*2 = 8. Case 3: Almost the same as above for the solution for the $e^4-4*e^3+6*e^2+e+1$ and the degree for c is 2 it is 4*2 = 8. Case 4: For the $e^2+3*e+1$, 2 more solutions and 2*2 = 4 here. Case 5: For the resting it is 3*5 = 15. So the total for this 35+8+8+4+15=70

I have shown some try in the untitled6.

4 Exercise 3:

4.1 Grobner Base

The polynomials that we could use is



$$\begin{aligned} f_1 &= 8x^2y^2 + 5xy^3 + 3x^3z + x^2yz \\ f_2 &= x^5 + 2y^3z^2 + 13y^2z^3 + 5yz^4 \end{aligned}$$

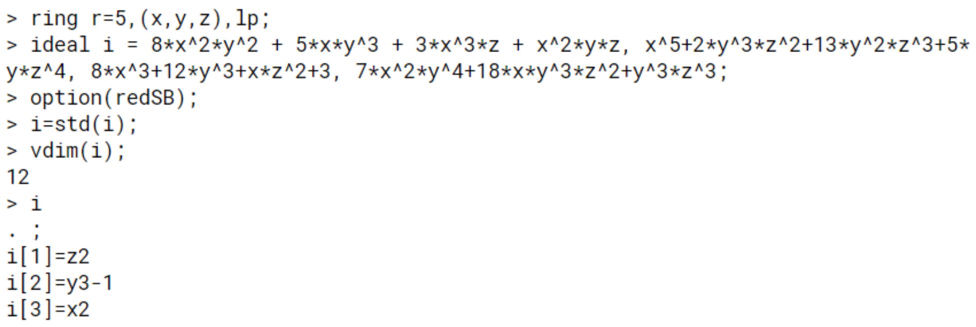
$$\begin{aligned} f_3 &= 8x^3 + 12y^3 + xz^2 + 3 \\ f_4 &= 7x^2y^4 + 18xy^3z^2 + y^3z^3. \end{aligned}$$

In the dp order, we could find out that the answer to this question is that (as shown in the paper)

$$\begin{cases} x \\ y^3 + 0.25 \\ z^2 \end{cases}$$

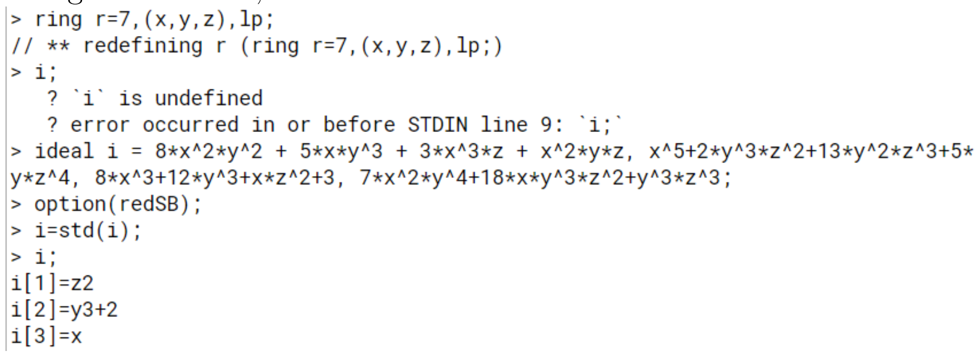
That is the total answer and here I would use some modular to run the theorem.

Using module of 5, I will show that



```
> ring r=5,(x,y,z),lp;
> ideal i = 8*x^2*y^2 + 5*x*y^3 + 3*x^3*z + x^2*y*z, x^5+2*y^3*z^2+13*y^2*z^3+5*
y*z^4, 8*x^3+12*y^3+x*z^2+3, 7*x^2*y^4+18*x*y^3*z^2+y^3*z^3;
> option(redSB);
> i=std(i);
> vdim(i);
12
> i
. ;
i[1]=z2
i[2]=y3-1
i[3]=x2
```

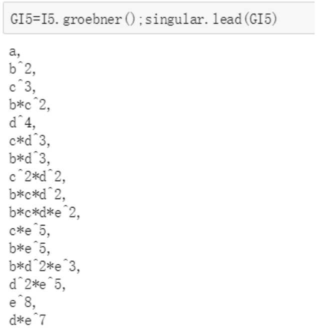
Using module of 7, I will show that



```
> ring r=7,(x,y,z),lp;
// ** redefining r (ring r=7,(x,y,z),lp;)
> i;
   ? `i` is undefined
   ? error occurred in or before STDIN line 9: `i;`
> ideal i = 8*x^2*y^2 + 5*x*y^3 + 3*x^3*z + x^2*y*z, x^5+2*y^3*z^2+13*y^2*z^3+5*
y*z^4, 8*x^3+12*y^3+x*z^2+3, 7*x^2*y^4+18*x*y^3*z^2+y^3*z^3;
> option(redSB);
> i=std(i);
> i;
i[1]=z2
i[2]=y3+2
i[3]=x
```

4.2 Cyclic-5

We can get the answer of the leading monomial sets by using the modular prime 2-13 easily using the sage.
For example, we would like to get the answer of the module 5 here.



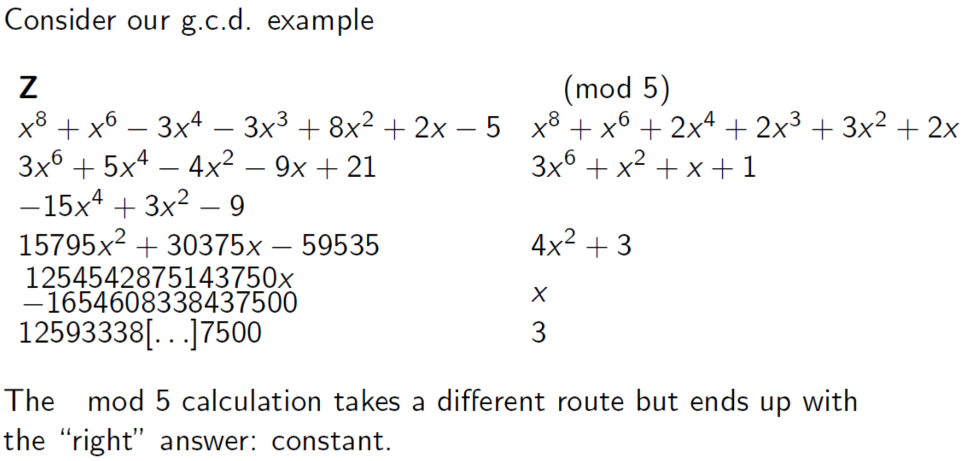
```
GI5=I5.groebner();singular.lead(GI5)
```

```
a,
b^2,
c^3,
b*c^2,
d^4,
c*d^3,
b*d^3,
c^2*d^2,
b*c*d^2,
b*c*d*e^2,
c*e^5,
b*e^5,
b*d^2*e^3,
d^2*e^5,
e^8,
d*e^7
```

For combining, we should first use the CRT to get the combining solutions. According to the lecture 7, we should use CRT with Farey Algorithm.

And we can see, any prime in the range of [2,13] is good polynomial for not dividing the any z_i

As an example, there is a typical one that we could show that the work.



Consider our g.c.d. example

$\mathbf{Z}$	(mod 5)
$x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5$	$x^8 + x^6 + 2x^4 + 2x^3 + 3x^2 + 2x$
$3x^6 + 5x^4 - 4x^2 - 9x + 21$	$3x^6 + x^2 + x + 1$
$-15x^4 + 3x^2 - 9$	
$15795x^2 + 30375x - 59535$	$4x^2 + 3$
$1254542875143750x$ -1654608338437500	x
$12593338[\ldots]7500$	3

The mod 5 calculation takes a different route but ends up with the “right” answer: constant.

The right is the result after doing the coefficient of elimination. And the example can be a strong evidence of not eliminating any prime of (2,13) because any prime which can even divide the number can even be good prime.

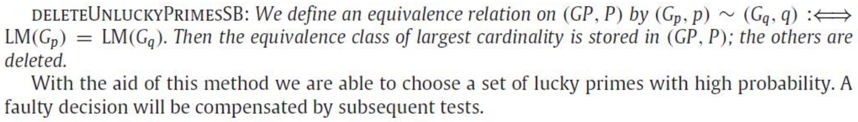
So here we would use all the prime between (2,13).

For how to choose the prime, I have something else to say. That is the one in the paper, (Arn2003 and the Id2011) it has something special to state.

4.2.1 PrimChoosing

I will quote something important here to say about the prime choosing.

First,



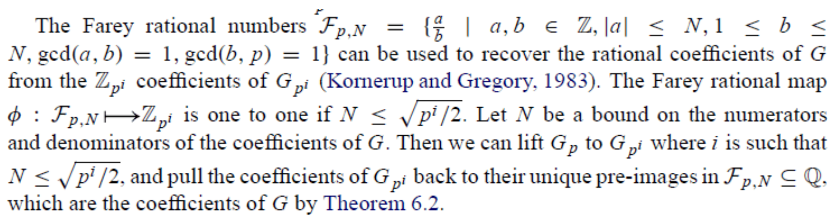
deleteUnluckyPrimesSB: *We define an equivalence relation on (GP, P) by $(G_p, p) \sim (G_q, q) :\Longleftrightarrow \mathrm{LM}(G_p) = \mathrm{LM}(G_q)$. Then the equivalence class of largest cardinality is stored in (GP, P); the others are deleted.*

With the aid of this method we are able to choose a set of lucky primes with high probability. A faulty decision will be compensated by subsequent tests.

So due to the theorm, we can see that we can ignore some primes due to this theorm. That is, once two primes can reduce the polynomials in the same result, then we can ignore the second prime, just using the first prime.

Hence deduce, we get the result of with p$_i$.

And once we want to get the lifting result(actually, it is the Farey Algorithm), we would use another algorithm.

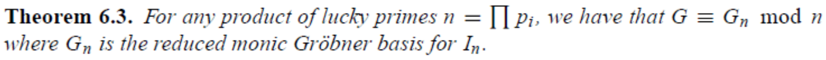


The Farey rational numbers $\mathcal{F}_{p,N} = \{\frac{a}{b} \mid a, b \in \mathbb{Z}, |a| \leq N, 1 \leq b \leq N, \gcd(a, b) = 1, \gcd(b, p) = 1\}$ can be used to recover the rational coefficients of G from the $\mathbb{Z}_{p^i}$ coefficients of G_{p^i} (Kornerup and Gregory, 1983). The Farey rational map $\phi : \mathcal{F}_{p,N} \longmapsto \mathbb{Z}_{p^i}$ is one to one if $N \leq \sqrt{p^i/2}$. Let N be a bound on the numerators and denominators of the coefficients of G. Then we can lift G_p to G_{p^i} where i is such that $N \leq \sqrt{p^i/2}$, and pull the coefficients of G_{p^i} back to their unique pre-images in $\mathcal{F}_{p,N} \subseteq \mathbb{Q}$, which are the coefficients of G by Theorem 6.2.

Here we lift the result of the prime p, to get the module of the p^i.

Then the following result can be got after the CRT.

That is for $n = \prod_{i=1}^{k} p_i^a i$ we can get the modular result, so that we can improve our efficiency greatly.

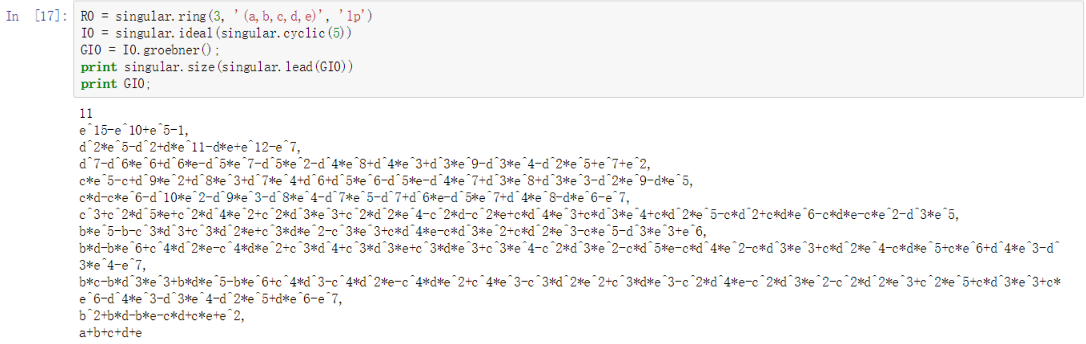


Theorem 6.3. *For any product of lucky primes $n = \prod p_i$, we have that $G \equiv G_n \mod n$ where G_n is the reduced monic Gröbner basis for I_n.*

Once the n is bigger than all the coefficient, we can say that we have achieved our goal with recognizing the R[x].

That is the general way for all the polynomials, I bet that is enough for this section.

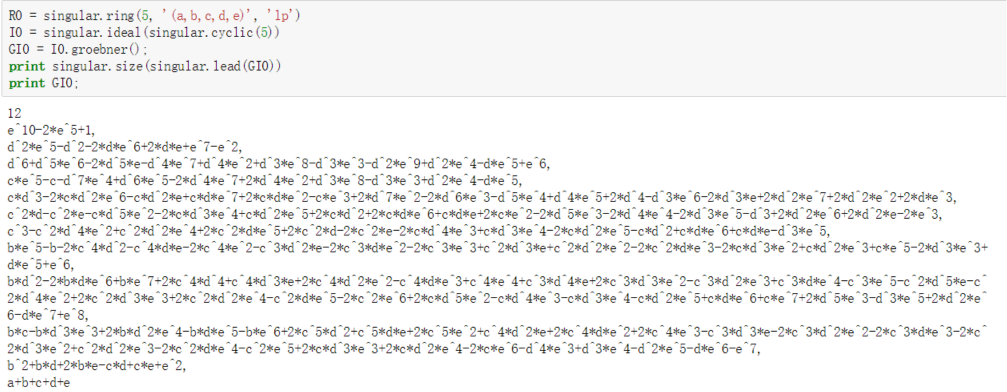
Next, I will show some results here. I will show some of the results of the calculation.
Module of 3:



```
In [17]: R0 = singular.ring(3, '(a,b,c,d,e)', 'lp')
         I0 = singular.ideal(singular.cyclic(5))
         GI0 = I0.groebner();
         print singular.size(singular.lead(GI0))
         print GI0;
```

```
11
e^15-e^10+e^5-1,
d^2*e^5-d^2+d*e^11-d*e+e^12-e^7,
d^7-d^6*e^6+d^6*e-d^5*e^7-d^5*e^2-d^4*e^8+d^4*e^3+d^3*e^9-d^3*e^4-d^2*e^5+e^7+e^2,
c*e^5-c+d^9*e^2+d^8*e^3+d^7*e^4+d^6+d^5*e^6-d^5*e-d^4*e^7+d^3*e^8+d^3*e^3-d^2*e^9-d*e^5,
c*d-c*e^6-d^10*e^2-d^9*e^3-d^8*e^4-d^7*e^5-d^7+d^6*e-d^5*e^7+d^4*e^8-d*e^6-e^7,
c^3+c^2*d^5*e+c^2*d^4*e^2+c^2*d^3*e^3+c^2*d^2*e^4-c^2*d-c^2*e+c*d^4*e^3+c*d^3*e^4+c*d^2*e^5-c*d^2+c*d*e^6-c*d*e-c*e^2-d^3*e^5,
b*e^5-b-c^3*d^3+c^3*d^2*e+c^3*d*e^2-c^3*e^3+c*d^4*e-c*d^3*e^2+c*d^2*e^3-c*e^5-d^3*e^3+e^6,
b*d-b*e^6+c^4*d^2*e-c^4*d*e^2+c^3*d^4+c^3*d^3*e+c^3*d*e^3+c^3*e^4-c^2*d^3*e^2-c*d^5*e-c*d^4*e^2-c*d^3*e^3+c*d^2*e^4-c*d*e^5+c*e^6+d^4*e^3-d^
3*e^4-e^7,
b*c-b*d^3*e^3+b*d*e^5-b*e^6+c^4*d^3-c^4*d^2*e-c^4*d*e^2+c^4*e^3-c^3*d^2*e^2+c^3*d*e^3-c^2*d^4*e-c^2*d^3*e^2-c^2*d^2*e^3+c^2*e^5+c*d^3*e^3+c*
e^6-d^4*e^3-d^3*e^4-d^2*e^5+d*e^6-e^7,
b^2+b*d-b*e-c*d+c*e+e^2,
a+b+c+d+e
```

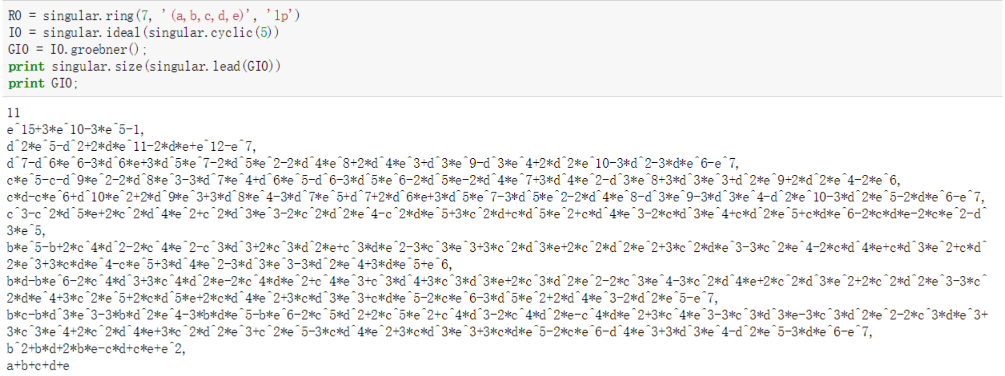
Module of 5:



```
R0 = singular.ring(5, '(a,b,c,d,e)', 'lp')
I0 = singular.ideal(singular.cyclic(5))
GI0 = I0.groebner();
print singular.size(singular.lead(GI0))
print GI0;
```

```
12
e^10-2*e^5+1,
d^2*e^5-d^2-2*d*e^6+2*d*e+e^7-e^2,
d^6+d^5*e^6-2*d^5*e-d^4*e^7+d^4*e^2+d^3*e^8-d^3*e^3-d^2*e^9+d^2*e^4-d*e^5+e^6,
c*e^5-c-d^7*e^4+d^6*e^5-2*d^4*e^7+2*d^4*e^2+d^3*e^8-d^3*e^3+d^2*e^4-d*e^5,
c*d^3-2*c*d^2*e^6-c*d^2*e+c*d*e^7+2*c*d*e^2-c*e^3+2*d^7*e^2-2*d^6*e^3-d^5*e^4+d^4*e^5+2*d^4-d^3*e^6-2*d^3*e+2*d^2*e^7+2*d^2*e^2+2*d*e^3,
c^2*d-c^2*e-c*d^5*e^2-2*c*d^3*e^4+c*d^2*e^5+2*c*d^2+2*c*d*e^6+c*d*e+2*c*e^2-2*d^5*e^3-2*d^4*e^4-2*d^3*e^5-d^3+2*d^2*e^6+2*d^2*e-2*e^3,
c^3-c^2*d^4*e^2+c^2*d^2*e^4+2*c^2*d*e^5+2*c^2*d-2*c^2*e-2*c*d^4*e^3+c*d^3*e^4-2*c*d^2*e^5-c*d^2+c*d*e^6+c*d*e-d^3*e^5,
b*e^5-b-2*c^4*d^2-c^4*d*e-2*c^4*e^2-c^3*d^2*e-2*c^3*d*e^2-2*c^3*e^3+c^2*d^3*e+c^2*d^2*e^2-2*c^2*d*e^3-2*c*d^3*e^2+c*d^2*e^3+c*e^5-2*d^3*e^3+
d*e^5+e^6,
b*d^2-2*b*d*e^6+b*e^7+2*c^4*d^4+c^4*d^3*e+2*c^4*d^2*e^2-c^4*d*e^3+c^4*e^4+c^3*d^4*e+2*c^3*d^3*e^2-c^3*d^2*e^3+c^3*d*e^4-c^3*e^5-c^2*d^5*e-c^
2*d^4*e^2+2*c^2*d^3*e^3+2*c^2*d^2*e^4-c^2*d*e^5-2*c^2*e^6+2*c*d^5*e^2-c*d^4*e^3-c*d^3*e^4-c*d^2*e^5+c*d*e^6+c*e^7+2*d^5*e^3-d^3*e^5+2*d^2*e^
6-d*e^7+e^8,
b*c-b*d^3*e^3+2*b*d^2*e^4-b*d*e^5-b*e^6+2*c^5*d^2+c^5*d*e+2*c^5*e^2+c^4*d^2*e+2*c^4*d*e^2+2*c^4*e^3-c^3*d^3*e-2*c^3*d^2*e^2-2*c^3*d*e^3-2*c^
2*d^3*e^2+c^2*d^2*e^3-2*c^2*d*e^4-c^2*e^5+2*c*d^3*e^3+2*c*d^2*e^4-2*c*e^6-d^4*e^3+d^3*e^4-d^2*e^5-d*e^6-e^7,
b^2+b*d+2*b*e-c*d+c*e+e^2,
a+b+c+d+e
```

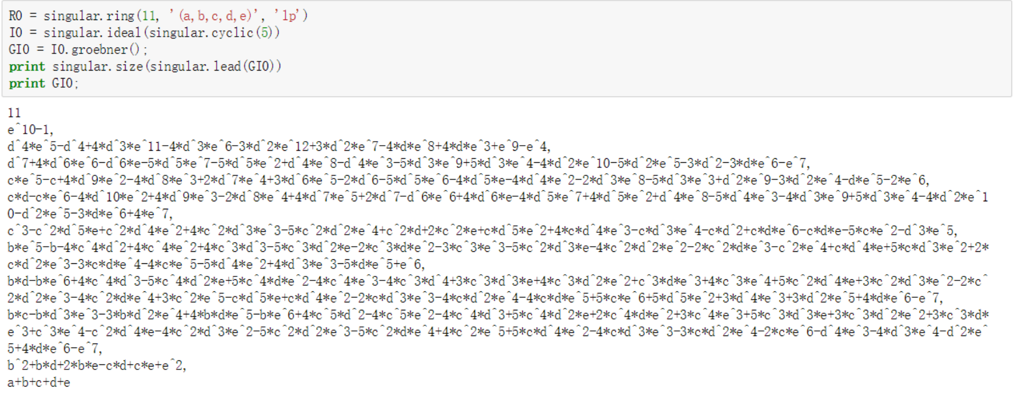
Module of 7:



```
R0 = singular.ring(7, '(a,b,c,d,e)', 'lp')
I0 = singular.ideal(singular.cyclic(5))
GI0 = I0.groebner();
print singular.size(singular.lead(GI0))
print GI0;
```

```
11
e^15+3*e^10-3*e^5-1,
d^2*e^5-d^2+2*d*e^11-2*d*e+e^12-e^7,
d^7-d^6*e^6-3*d^6*e+3*d^5*e^7-2*d^5*e^2-2*d^4*e^8+2*d^4*e^3+d^3*e^9-d^3*e^4+2*d^2*e^10-3*d^2-3*d*e^6-e^7,
c*e^5-c-d^9*e^2-2*d^8*e^3-3*d^7*e^4+d^6*e^5-d^6-3*d^5*e^6-2*d^5*e-2*d^4*e^7+3*d^4*e^2-d^3*e^8+3*d^3*e^3+d^2*e^9+2*d^2*e^4-2*e^6,
c*d-c*e^6+d^10*e^2+2*d^9*e^3+3*d^8*e^4-3*d^7*e^5+d^7+2*d^6*e+3*d^5*e^7-3*d^5*e^2-2*d^4*e^8-d^3*e^9-3*d^3*e^4-d^2*e^10-3*d^2*e^5-2*d*e^6-e^7,
c^3-c^2*d^5*e+2*c^2*d^4*e^2+c^2*d^3*e^3-2*c^2*d^2*e^4-c^2*d*e^5+3*c^2*d+c*d^5*e^2+c*d^4*e^3-2*c*d^3*e^4+c*d^2*e^5+c*d*e^6-2*c*d*e-2*c*e^2-d^
3*e^5,
b*e^5-b+2*c^4*d^2-2*c^4*e^2-c^3*d^3+2*c^3*d^2*e+c^3*d*e^2-3*c^3*e^3+3*c^2*d^3*e+2*c^2*d^2*e^2+3*c^2*d*e^3-3*c^2*e^4-2*c*d^4*e+c*d^3*e^2+c*d^
2*e^3+3*c*d*e^4-c*e^5+3*d^4*e^2-3*d^3*e^3-3*d^2*e^4+3*d*e^5+e^6,
b*d-b*e^6-2*c^4*d^3+3*c^4*d^2*e-2*c^4*d*e^2+c^4*e^3+c^3*d^4+3*c^3*d^3*e+2*c^3*d^2*e^2-2*c^3*e^4-3*c^2*d^4*e+2*c^2*d^3*e^2+2*c^2*d^2*e^3-3*c^
2*d*e^4+3*c^2*e^5+2*c*d^5*e+2*c*d^4*e^2+3*c*d^3*e^3+c*d*e^5-2*c*e^6-3*d^5*e^2+2*d^4*e^3-2*d^2*e^5-e^7,
b*c-b*d^3*e^3-3*b*d^2*e^4-3*b*d*e^5-b*e^6-2*c^5*d^2+2*c^5*e^2+c^4*d^3-2*c^4*d^2*e-c^4*d*e^2+3*c^4*e^3-3*c^3*d^3*e-3*c^3*d^2*e^2-2*c^3*d*e^3+
3*c^3*e^4+2*c^2*d^4*e+3*c^2*d^2*e^3+c^2*e^5-3*c*d^4*e^2+3*c*d^3*e^3+3*c*d*e^5-2*c*e^6-d^4*e^3+3*d^3*e^4-d^2*e^5-3*d*e^6-e^7,
b^2+b*d+2*b*e-c*d+c*e+e^2,
a+b+c+d+e
```

Module of 11:



```
R0 = singular.ring(11, '(a,b,c,d,e)', 'lp')
I0 = singular.ideal(singular.cyclic(5))
GI0 = I0.groebner();
print singular.size(singular.lead(GI0))
print GI0;
```

```
11
e^10-1,
d^4*e^5-d^4+4*d^3*e^11-4*d^3*e^6-3*d^2*e^12+3*d^2*e^7-4*d*e^8+4*d*e^3+e^9-e^4,
d^7+4*d^6*e^6-d^6*e-5*d^5*e^7-5*d^5*e^2+d^4*e^8-d^4*e^3-5*d^3*e^9+5*d^3*e^4-4*d^2*e^10-5*d^2*e^5-3*d^2-3*d*e^6-e^7,
c*e^5-c+4*d^9*e^2-4*d^8*e^3+2*d^7*e^4+3*d^6*e^5-2*d^6-5*d^5*e^6-4*d^5*e-4*d^4*e^2-2*d^3*e^8-5*d^3*e^3+d^2*e^9-3*d^2*e^4-d*e^5-2*e^6,
c*d-c*e^6-4*d^10*e^2+4*d^9*e^3-2*d^8*e^4+4*d^7*e^5+2*d^7-d^6*e^6+4*d^6*e-4*d^5*e^7+4*d^5*e^2+d^4*e^8-5*d^4*e^3-4*d^3*e^9+5*d^3*e^4-4*d^2*e^1
0-d^2*e^5-3*d*e^6+4*e^7,
c^3-c^2*d^5*e+c^2*d^4*e^2+4*c^2*d^3*e^3-5*c^2*d^2*e^4+c^2*d+2*c^2*e+c*d^5*e^2+4*c*d^4*e^3-c*d^3*e^4-c*d^2+c*d*e^6-c*d*e-5*c*e^2-d^3*e^5,
b*e^5-b-4*c^4*d^2+4*c^4*e^2+4*c^3*d^3-5*c^3*d^2*e-2*c^3*d*e^2-3*c^3*e^3-5*c^2*d^3*e-4*c^2*d^2*e^2-2*c^2*d*e^3-c^2*e^4+c*d^4*e+5*c*d^3*e^2+2*
c*d^2*e^3-3*c*d*e^4-4*c*e^5-5*d^4*e^2+4*d^3*e^3-5*d*e^5+e^6,
b*d-b*e^6+4*c^4*d^3-5*c^4*d^2*e+5*c^4*d*e^2-4*c^4*e^3-4*c^3*d^4+3*c^3*d^3*e+4*c^3*d^2*e^2+c^3*d*e^3+4*c^3*e^4+5*c^2*d^4*e+3*c^2*d^3*e^2-2*c^
2*d^2*e^3-4*c^2*d*e^4+3*c^2*e^5-c*d^5*e+c*d^4*e^2-2*c*d^3*e^3-4*c*d^2*e^4-4*c*d*e^5+5*c*e^6+5*d^5*e^2+3*d^4*e^3+3*d^2*e^5+4*d*e^6-e^7,
b*c-b*d^3*e^3-3*b*d^2*e^4+4*b*d*e^5-b*e^6+4*c^5*d^2-4*c^5*e^2-4*c^4*d^3+5*c^4*d^2*e+2*c^4*d*e^2+3*c^4*e^3+5*c^3*d^3*e+3*c^3*d^2*e^2+3*c^3*d*
e^3+c^3*e^4-c^2*d^4*e-4*c^2*d^3*e^2-5*c^2*d^2*e^3-5*c^2*d*e^4+4*c^2*e^5+5*c*d^4*e^2-4*c*d^3*e^3-3*c*d^2*e^4-2*c*e^6-d^4*e^3-4*d^3*e^4-d^2*e^
5+4*d*e^6-e^7,
b^2+b*d+2*b*e-c*d+c*e+e^2,
a+b+c+d+e
```

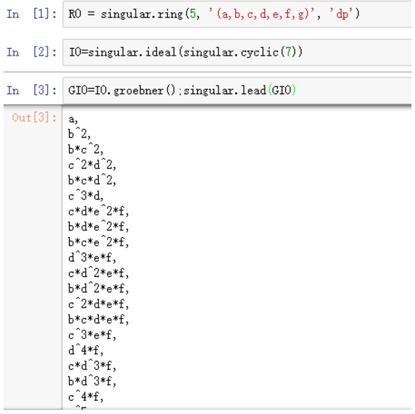
And we can see that all these result are good, the resting parts can be end now.

4.3 Cyclic-7

I understand this section as a part to implement the modular algorithm to get the simplification of the coefficient like the one described Arn2003. That is to say the process is almost the same as the one in the Cyclic-5 and what we should do is to repeat the process.

So here I will start to list some of these simplification.

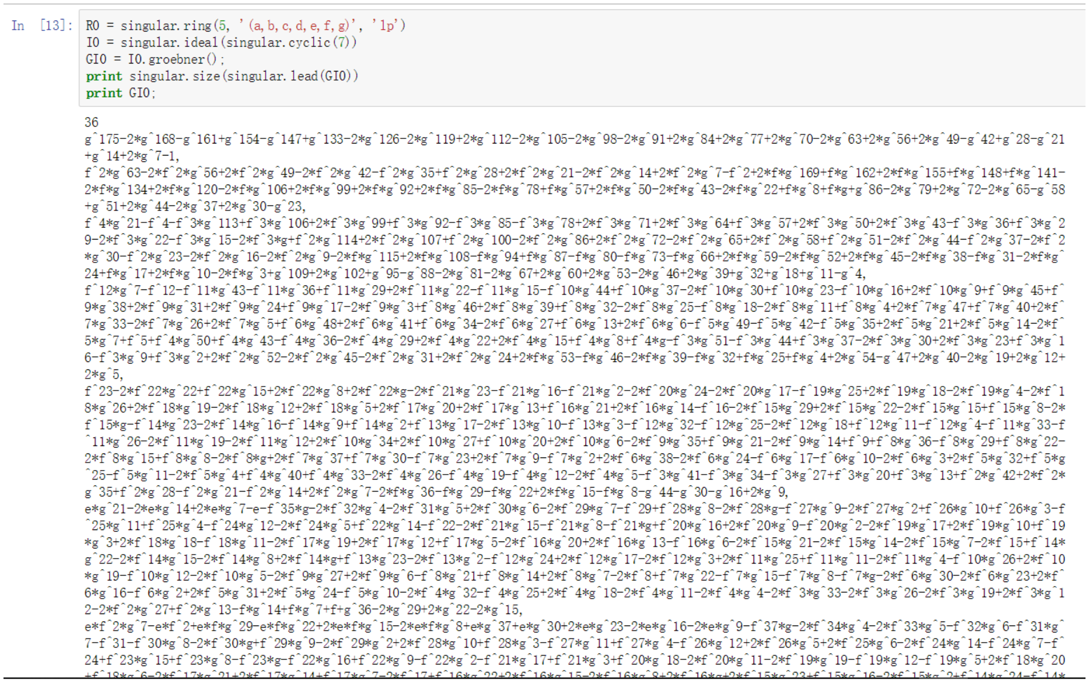
But I am not sure whether I should give the process about the CRT, it is quite hard to test. So that I can not show the result of testing whether it is a good polynomial. I will only give out some pictures of using some prime to reduce.



```
In  [1]: R0 = singular.ring(5, '(a,b,c,d,e,f,g)', 'dp')

In  [2]: I0=singular.ideal(singular.cyclic(7))

In  [3]: GI0=I0.groebner();singular.lead(GI0)

Out[3]: a,
        b^2,
        b*c^2,
        c^2*d^2,
        b*c*d^2,
        c^3*d,
        c*d*e^2*f,
        b*d*e^2*f,
        b*c*e^2*f,
        d^3*e*f,
        c*d^2*e*f,
        b*d^2*e*f,
        c^2*d*e*f,
        b*c*d*e*f,
        c^3*e*f,
        d^4*f,
        c*d^3*f,
        b*d^3*f,
        c^4*f,
```

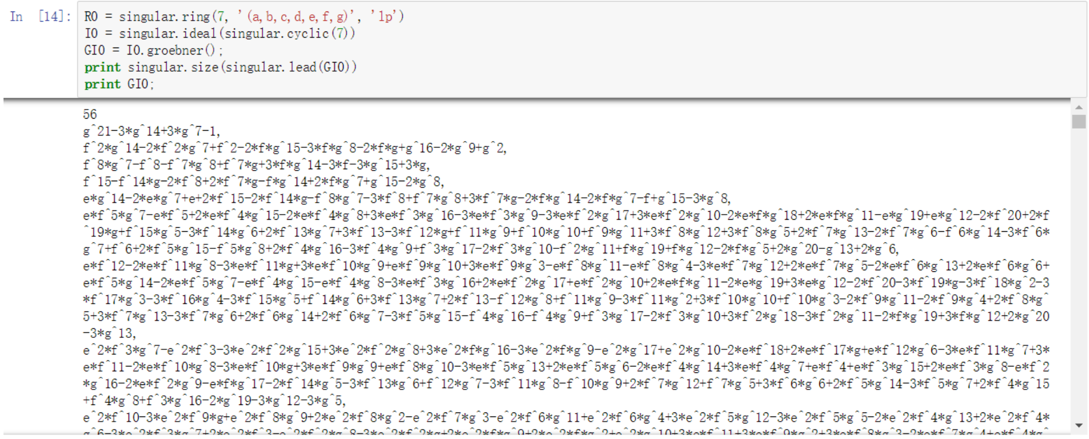
I will post some of the results here.

Using the module of 5, the result should be:



```
In  [13]: R0 = singular.ring(5, '(a,b,c,d,e,f,g)', '1p')
          I0 = singular.ideal(singular.cyclic(7))
          GI0 = I0.groebner();
          print singular.size(singular.lead(GI0))
          print GI0;

36
g^175-2*g^168-g^161+g^154-g^147+g^133-2*g^126-2*g^119+2*g^112-2*g^105-2*g^98-2*g^91+2*g^84+2*g^77+2*g^70-2*g^63+2*g^56+2*g^49-g^42+g^28-g^21
+g^14+2*g^7-1,
f^2*g^63-2*f^2*g^56+2*f^2*g^49-2*f^2*g^42-f^2*g^35+f^2*g^28+2*f^2*g^21-2*f^2*g^14+2*f^2*g^7-f^2+2*f*g^169+f*g^162+2*f*g^155+f*g^148+f*g^141-
2*f*g^134+2*f*g^120-2*f*g^106+2*f*g^99+2*f*g^92+2*f*g^85-2*f*g^78+f*g^57+2*f*g^50-2*f*g^43-2*f*g^22+f*g^8+f*g+g^86-2*g^79+2*g^72-2*g^65-g^58
+g^51+2*g^44-2*g^37+2*g^30-g^23,
f^4*g^21-f^4-f^3*g^113+f^3*g^106+2*f^3*g^99+f^3*g^92-f^3*g^85-f^3*g^78+2*f^3*g^71+2*f^3*g^64+f^3*g^57+2*f^3*g^50+2*f^3*g^43-f^3*g^36+f^3*g^2
9-2*f^3*g^22-f^3*g^15-2*f^3*g+f^2*g^114+2*f^2*g^107+f^2*g^100-2*f^2*g^86+2*f^2*g^72-2*f^2*g^65+2*f^2*g^58+f^2*g^51-2*f^2*g^44-f^2*g^37-2*f^2
*g^30-f^2*g^23-2*f^2*g^16-2*f^2*g^9-2*f*g^115+2*f*g^108-f*g^94+f*g^87-f*g^80-f*g^73-f*g^66+2*f*g^59-2*f*g^52+2*f*g^45-2*f*g^38-f*g^31-2*f*g^
24+f*g^17+2*f*g^10-2*f*g^3+g^109+2*g^102+g^95-g^88-2*g^81-2*g^67+2*g^60+2*g^53-2*g^46+2*g^39+g^32+g^18+g^11-g^4,
f^12*g^7-f^12-f^11*g^43-f^11*g^36+f^11*g^29+2*f^11*g^22-f^11*g^15-f^10*g^44+f^10*g^37-2*f^10*g^30+f^10*g^23-f^10*g^16+2*f^10*g^9+f^9*g^45+f^
9*g^38+2*f^9*g^31+2*f^9*g^24+f^9*g^17-2*f^9*g^3+f^8*g^46+2*f^8*g^39+f^8*g^32-2*f^8*g^25-f^8*g^18-2*f^8*g^11+f^8*g^4+2*f^7*g^47+f^7*g^40+2*f^
7*g^33-2*f^7*g^26+2*f^7*g^5+f^6*g^48+2*f^6*g^41+f^6*g^34-2*f^6*g^27+f^6*g^13+2*f^6*g^6-f^5*g^49-f^5*g^42-f^5*g^35+2*f^5*g^21+2*f^5*g^14-2*f^
5*g^7+f^5+f^4*g^50+f^4*g^43-f^4*g^36-2*f^4*g^29+2*f^4*g^22+2*f^4*g^15+f^4*g^8+f^4*g-f^3*g^51-f^3*g^44+f^3*g^37-2*f^3*g^30+2*f^3*g^23+f^3*g^1
6-f^3*g^9+f^3*g^2+2*f^2*g^52-2*f^2*g^45-2*f^2*g^31+2*f^2*g^24+2*f*g^53-f*g^46-2*f*g^39-f*g^32+f*g^25+f*g^4+2*g^54-g^47+2*g^40-2*g^19+2*g^12+
2*g^5,
f^23-2*f^22*g^22+f^22*g^15+2*f^22*g^8+2*f^22*g-2*f^21*g^23-f^21*g^16-f^21*g^2-2*f^20*g^24-2*f^20*g^17-f^19*g^25+2*f^19*g^18-2*f^19*g^4-2*f^1
8*g^26+2*f^18*g^19-2*f^18*g^12+2*f^18*g^5+2*f^17*g^20+2*f^17*g^13+f^16*g^21+2*f^16*g^14-f^16-2*f^15*g^29+2*f^15*g^22-2*f^15*g^15+f^15*g^8-2*
f^15*g-f^14*g^23-2*f^14*g^16-f^14*g^9+f^14*g^2+f^13*g^17-2*f^13*g^10-f^13*g^3-f^12*g^32-f^12*g^25-2*f^12*g^18+f^12*g^11-f^12*g^4-f^11*g^33-f
^11*g^26-2*f^11*g^19-2*f^11*g^12+2*f^10*g^34+2*f^10*g^27+f^10*g^20+2*f^10*g^6-2*f^9*g^35+f^9*g^21-2*f^9*g^14+f^9+f^8*g^36-f^8*g^29+f^8*g^22-
2*f^8*g^15+f^8*g^8-2*f^8*g+2*f^7*g^37+f^7*g^30-f^7*g^23+2*f^7*g^9-f^7*g^2+2*f^6*g^38-2*f^6*g^24-f^6*g^17-f^6*g^10-2*f^6*g^3+2*f^5*g^32+f^5*g
^25-f^5*g^11-2*f^5*g^4+f^4*g^40+f^4*g^33-2*f^4*g^26-f^4*g^19-f^4*g^12-2*f^4*g^5-f^3*g^41-f^3*g^34-f^3*g^27+f^3*g^20+f^3*g^13+f^2*g^42+2*f^2*
g^35+f^2*g^28-f^2*g^21-f^2*g^14+2*f^2*g^7-2*f*g^36-f*g^29-f*g^22+2*f*g^15-f*g^8-g^44-g^30-g^16+2*g^9,
e*g^21-2*e*g^14+2*e*g^7-e-f^35*g-2*f^32*g^4-2*f^31*g^5+2*f^30*g^6-2*f^29*g^7-f^29+f^28*g^8-2*f^28*g-f^27*g^9-2*f^27*g^2+f^26*g^10+f^26*g^3-f
^25*g^11+f^25*g^4-f^24*g^12-2*f^24*g^5+f^22*g^14-f^22-2*f^21*g^15-f^21*g^8-f^21*g+f^20*g^16+2*f^20*g^9-f^20*g^2-2*f^19*g^17+2*f^19*g^10+f^19
*g^3+2*f^18*g^18-f^18*g^11-2*f^17*g^19+2*f^17*g^12+f^17*g^5-2*f^16*g^20+2*f^16*g^13-f^16*g^6-2*f^15*g^21-2*f^15*g^14-2*f^15*g^7-2*f^15+f^14*
g^22-2*f^14*g^15-2*f^14*g^8+2*f^14*g+f^13*g^23-2*f^13*g^2-f^12*g^24+2*f^12*g^17-2*f^12*g^3+2*f^11*g^25+f^11*g^11-2*f^11*g^4-f^10*g^26+2*f^10
*g^19-f^10*g^12-2*f^10*g^5-2*f^9*g^27+2*f^9*g^6-f^8*g^21+f^8*g^14+2*f^8*g^7-2*f^8+f^7*g^22-f^7*g^15-f^7*g^8-f^7*g-2*f^6*g^30-2*f^6*g^23+2*f^
6*g^16-f^6*g^2+2*f^5*g^31+2*f^5*g^24-f^5*g^10-2*f^4*g^32-f^4*g^25+2*f^4*g^18-2*f^4*g^11-2*f^4*g^4-2*f^3*g^33-2*f^3*g^26-2*f^3*g^19+2*f^3*g^1
2-2*f^2*g^27+f^2*g^13-f*g^14+f*g^7+f+g^36-2*g^29+2*g^22-2*g^15,
e*f^2*g^7-e*f^2+e*f*g^29-e*f*g^22+2*e*f*g^15-2*e*f*g^8+e*g^37+e*g^30+2*e*g^23-2*e*g^16-2*e*g^9-f^37*g-2*f^34*g^4-2*f^33*g^5-f^32*g^6-f^31*g^
7-f^31-f^30*g^8-2*f^30*g+f^29*g^9-2*f^29*g^2+2*f^28*g^10+f^28*g^3-f^27*g^11+f^27*g^4-f^26*g^12+2*f^26*g^5+2*f^25*g^6-2*f^24*g^14-f^24*g^7-f^
24+f^23*g^15+f^23*g^8-f^23*g-f^22*g^16+f^22*g^9-f^22*g^2-f^21*g^17+f^21*g^3+f^20*g^18-2*f^20*g^11-2*f^19*g^19-f^19*g^12-f^19*g^5+2*f^18*g^20
+f^18*g^6-2*f^17*g^21+2*f^17*g^14+f^17*g^7-2*f^17+f^16*g^22+2*f^16*g^15-2*f^16*g^8+2*f^16*g+2*f^15*g^23+f^15*g^16-2*f^15*g^2+f^14*g^24-f^14*
```

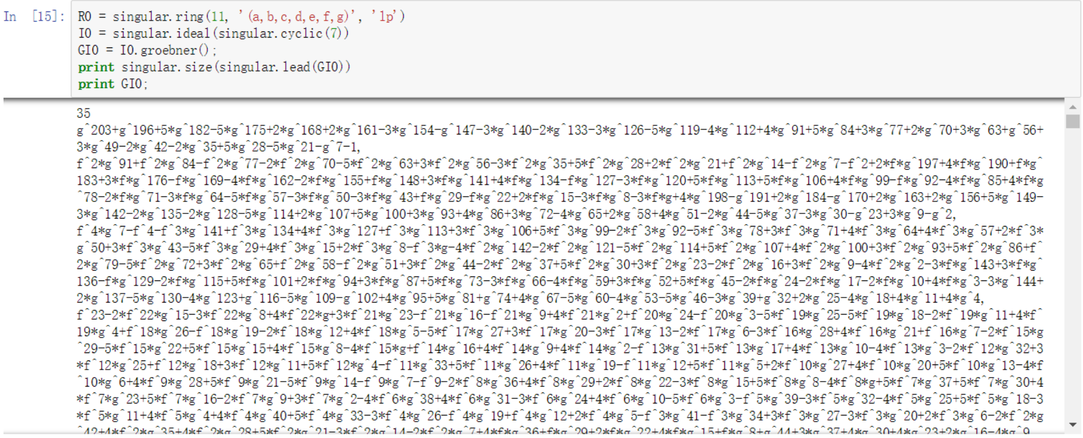
Using the module of 7, the result should be:



```
In  [14]: R0 = singular.ring(7, '(a,b,c,d,e,f,g)', 'lp')
          I0 = singular.ideal(singular.cyclic(7))
          GI0 = I0.groebner();
          print singular.size(singular.lead(GI0))
          print GI0;
```

```
56
g^21-3*g^14+3*g^7-1,
f^2*g^14-2*f^2*g^7+f^2-2*f*g^15-3*f*g^8-2*f*g+g^16-2*g^9+g^2,
f^8*g^7-f^8-f^7*g^8+f^7*g+3*f*g^14-3*f-3*g^15+3*g,
f^15-f^14*g-2*f^8+2*f^7*g-f*g^14+2*f*g^7+g^15-2*g^8,
e*g^14-2*e*g^7+e+2*f^15-2*f^14*g-f^8*g^7-3*f^8+f^7*g^8+3*f^7*g-2*f*g^14-2*f*g^7-f+g^15-3*g^8,
e*f^5*g^7-e*f^5+2*e*f^4*g^15-2*e*f^4*g^8+3*e*f^3*g^16-3*e*f^3*g^9-3*e*f^2*g^17+3*e*f^2*g^10-2*e*f*g^18+2*e*f*g^11-e*g^19+e*g^12-2*f^20+2*f
^19*g+f^15*g^5-3*f^14*g^6+2*f^13*g^7+3*f^13-3*f^12*g+f^11*g^9+f^10*g^10+f^9*g^11+3*f^8*g^12+3*f^8*g^5+2*f^7*g^13-2*f^7*g^6-f^6*g^14-3*f^6*
g^7+f^6+2*f^5*g^15-f^5*g^8+2*f^4*g^16-3*f^4*g^9+f^3*g^17-2*f^3*g^10-f^2*g^11+f*g^19+f*g^12-2*f*g^5+2*g^20-g^13+2*g^6,
e*f^12-2*e*f^11*g^8-3*e*f^11*g+3*e*f^10*g^9+e*f^9*g^10+3*e*f^9*g^3-e*f^8*g^11-e*f^8*g^4-3*e*f^7*g^12+2*e*f^7*g^5-2*e*f^6*g^13+2*e*f^6*g^6+
e*f^5*g^14-2*e*f^5*g^7-e*f^4*g^15-e*f^4*g^8-3*e*f^3*g^16+2*e*f^2*g^17+e*f^2*g^10+2*e*f*g^11-2*e*g^19+3*e*g^12-2*f^20-3*f^19*g-3*f^18*g^2-3
*f^17*g^3-3*f^16*g^4-3*f^15*g^5+f^14*g^6+3*f^13*g^7+2*f^13-f^12*g^8+f^11*g^9-3*f^11*g^2+3*f^10*g^10+f^10*g^3-2*f^9*g^11-2*f^9*g^4+2*f^8*g^
5+3*f^7*g^13-3*f^7*g^6+2*f^6*g^14+2*f^6*g^7-3*f^5*g^15-f^4*g^16-f^4*g^9+f^3*g^17-2*f^3*g^10+3*f^2*g^18-3*f^2*g^11-2*f*g^19+3*f*g^12+2*g^20
-3*g^13,
e^2*f^3*g^7-e^2*f^3-3*e^2*f^2*g^15+3*e^2*f^2*g^8+3*e^2*f*g^16-3*e^2*f*g^9-e^2*g^17+e^2*g^10-2*e*f^18+2*e*f^17*g+e*f^12*g^6-3*e*f^11*g^7+3*
e*f^11-2*e*f^10*g^8-3*e*f^10*g+3*e*f^9*g^9+e*f^8*g^10-3*e*f^5*g^13+2*e*f^5*g^6-2*e*f^4*g^14+3*e*f^4*g^7+e*f^4+e*f^3*g^15+2*e*f^3*g^8-e*f^2
*g^16-2*e*f^2*g^9-e*f*g^17-2*f^14*g^5-3*f^13*g^6+f^12*g^7-3*f^11*g^8-f^10*g^9+2*f^7*g^12+f^7*g^5+3*f^6*g^6+2*f^5*g^14-3*f^5*g^7+2*f^4*g^15
+f^4*g^8+f^3*g^16-2*g^19-3*g^12-3*g^5,
e^2*f^10-3*e^2*f^9*g+e^2*f^8*g^9+2*e^2*f^8*g^2-e^2*f^7*g^3-e^2*f^6*g^11+e^2*f^6*g^4+3*e^2*f^5*g^12-3*e^2*f^5*g^5-2*e^2*f^4*g^13+2*e^2*f^4*
```

Using the module of 11, the result should be:



```
In  [15]: R0 = singular.ring(11, '(a,b,c,d,e,f,g)', 'lp')
          I0 = singular.ideal(singular.cyclic(7))
          GI0 = I0.groebner();
          print singular.size(singular.lead(GI0))
          print GI0;
```

```
35
g^203+g^196+5*g^182-5*g^175+2*g^168+2*g^161-3*g^154-g^147-3*g^140-2*g^133-3*g^126-5*g^119-4*g^112+4*g^91+5*g^84+3*g^77+2*g^70+3*g^63+g^56+
3*g^49-2*g^42-2*g^35+5*g^28-5*g^21-g^7-1,
f^2*g^91+f^2*g^84-f^2*g^77-2*f^2*g^70-5*f^2*g^63+3*f^2*g^56-3*f^2*g^35+5*f^2*g^28+2*f^2*g^21+f^2*g^14-f^2*g^7-f^2+2*f*g^197+4*f*g^190+f*g^
183+3*f*g^176-f*g^169-4*f*g^162-2*f*g^155+f*g^148+3*f*g^141+4*f*g^134-f*g^127-3*f*g^120+5*f*g^113+5*f*g^106+4*f*g^99-f*g^92-4*f*g^85+4*f*g
^78-2*f*g^71-3*f*g^64-5*f*g^57-3*f*g^50-3*f*g^43+f*g^29-f*g^22+2*f*g^15-3*f*g^8-3*f*g+4*g^198-g^191+2*g^184-g^170+2*g^163+2*g^156+5*g^149-
3*g^142-2*g^135-2*g^128-5*g^114+2*g^107+5*g^100+3*g^93+4*g^86+3*g^72-4*g^65+2*g^58+4*g^51-2*g^44-5*g^37-3*g^30-g^23+3*g^9-g^2,
f^4*g^7-f^4-f^3*g^141+f^3*g^134+4*f^3*g^127+f^3*g^113+3*f^3*g^106+5*f^3*g^99-2*f^3*g^92-5*f^3*g^78+3*f^3*g^71+4*f^3*g^64+4*f^3*g^57+2*f^3*
g^50+3*f^3*g^43-5*f^3*g^29+4*f^3*g^15+2*f^3*g^8-f^3*g-4*f^2*g^142-2*f^2*g^121-5*f^2*g^114+5*f^2*g^107+4*f^2*g^100+3*f^2*g^93+5*f^2*g^86+f^
2*g^79-5*f^2*g^72+3*f^2*g^65+f^2*g^58-f^2*g^51+3*f^2*g^44-2*f^2*g^37+5*f^2*g^30+3*f^2*g^23-2*f^2*g^16+3*f^2*g^9-4*f^2*g^2-3*f*g^143+3*f*g^
136-f*g^129-2*f*g^115+5*f*g^101+2*f*g^94+3*f*g^87+5*f*g^73-3*f*g^66-4*f*g^59+3*f*g^52+5*f*g^45-2*f*g^24-2*f*g^17-2*f*g^10+4*f*g^3-3*g^144+
2*g^137-5*g^130-4*g^123+g^116-5*g^109-g^102+4*g^95+5*g^81+g^74+4*g^67-5*g^60-4*g^53-5*g^46-3*g^39+g^32+2*g^25-4*g^18+4*g^11+4*g^4,
f^23-2*f^22*g^15-3*f^22*g^8+4*f^22*g+3*f^21*g^23-f^21*g^16-f^21*g^9+4*f^21*g^2+f^20*g^24-f^20*g^3-5*f^19*g^25-5*f^19*g^18-2*f^19*g^11+4*f^
19*g^4+f^18*g^26-f^18*g^19-2*f^18*g^12+4*f^18*g^5-5*f^17*g^27+3*f^17*g^20-3*f^17*g^13-2*f^17*g^6-3*f^16*g^28+4*f^16*g^21+f^16*g^7-2*f^15*g
^29-5*f^15*g^22+5*f^15*g^15+4*f^15*g^8-4*f^15*g+f^14*g^16+4*f^14*g^9+4*f^14*g^2-f^13*g^31+5*f^13*g^17+4*f^13*g^10-4*f^13*g^3-2*f^12*g^32+3
*f^12*g^25+f^12*g^18+3*f^12*g^11+5*f^12*g^4-f^11*g^33+5*f^11*g^26+4*f^11*g^19-f^11*g^12+5*f^11*g^5+2*f^10*g^27+4*f^10*g^20+5*f^10*g^13-4*f
^10*g^6+4*f^9*g^28+5*f^9*g^21-5*f^9*g^14-f^9*g^7-f^9-2*f^8*g^36+4*f^8*g^29+2*f^8*g^22-3*f^8*g^15+5*f^8*g^8-4*f^8*g+5*f^7*g^37+5*f^7*g^30+4
*f^7*g^23+5*f^7*g^16-2*f^7*g^9+3*f^7*g^2-4*f^6*g^38+4*f^6*g^31-3*f^6*g^24+4*f^6*g^10-5*f^6*g^3-f^5*g^39-3*f^5*g^32-4*f^5*g^25+5*f^5*g^18-3
*f^5*g^11+4*f^5*g^4+4*f^4*g^40+5*f^4*g^33-3*f^4*g^26-f^4*g^19+f^4*g^12+2*f^4*g^5-f^3*g^41-f^3*g^34+3*f^3*g^27-3*f^3*g^20+2*f^3*g^6-2*f^2*g
```

5 Conclusion

This partion is quite difficult for me, I think. The most important thing is that I have to master some other basic skills about the abstract algebra. I have tried to explain everything clearly now. I hope I really do well.