Calcul formel c’est quoi :

Logiciel de calcul mathématique créé en 2005 par William Stein
Alternative libre à Maple, Mathematica,
Basé sur des bibliothèques de calcul rapides (GMP, Pari, GAP, NTL, etc.)
Interface commune aux bibliothèques, basée sur python
Langage principal : Python
Listes : [1, 2, 3], mutable I
Tuples : (1, 2, 3), immuable I
Dictionnaires : {1:'a', 2:'b', 3:'c'}
range(12) renvoie des int
Flottants par défaut : RR

Une représentation fini et exacte
Arithmetique: nombre entiers, rationnel
Calcul algébrique: matrices, polynôme, series, groupes
Calcul symbolique: int´egration
⇒ Application: cryptographie, codage, robotique
calcul mathématique complexe par des approximations numeriques
calcul num´erique ⇒ r´esultat rapidement mais approch´e (voire faux)
calcul formel ⇒ r´esultat exact mais plus lent (voire inatteignable)
sage utilise un pr´ecision de 53 bits (16 chiffres decimaux)

exp.expand() ⇒ développe le produit
exp.collect() ⇒ factorise

corps finis `a q = p^n elements : GF(q)

Groupe multiplicatif : multiplication interne, inverse pour tous les éléments

Anneau : addition et multiplication internes, opposé pour tous les éléments
Corps : addition et multiplication internes, opposé pour tous les éléments, inverse
pour tous les éléments non nul

v.n(b) pour connaître v avec b bits de précision.
var('k')
limit(f(x),x=-infinity)
find_root
derivative(y(t),t)
solve(xp(t)==0,t)
eq.rhs() (rhs = right-hand side)
d = {R1:R1a, R2:R2a}
var('z')
fct(z)=5*z-10
solve([eq0,eq1],c1,c2)
nb.n(digits=10) nombre a 10 chiffres
Définir l'anneau des polynômes en une variable à coefficients rationnels, avec la commande R.<X> = QQ[]

matrix(n,n) construit une matrice de dimensions n×n remplie de 0
QQ(3) construit le rationnel 3 alors que ZZ(3) construit l'entier 3
pow(3,n,5) 3^n mod 5
A=Integer(50) stucture math des entier mod 50
A(65) contrsuit entier 65 dans A cad 65 mod 50 -> 15
find_root(f(x),-4,4) la racine dans l’intervalle [-4,4]
méthode roots() cherche à calculer toutes les racines de manières exacte dans SR
matrix(m,n[[ligne1],[ligne2],...,[lignem]]) m ligne, n colone
vector([e1,e2,....,em])

A1.rank() donne le rang
A.swap_rows(i,j) echange i et j