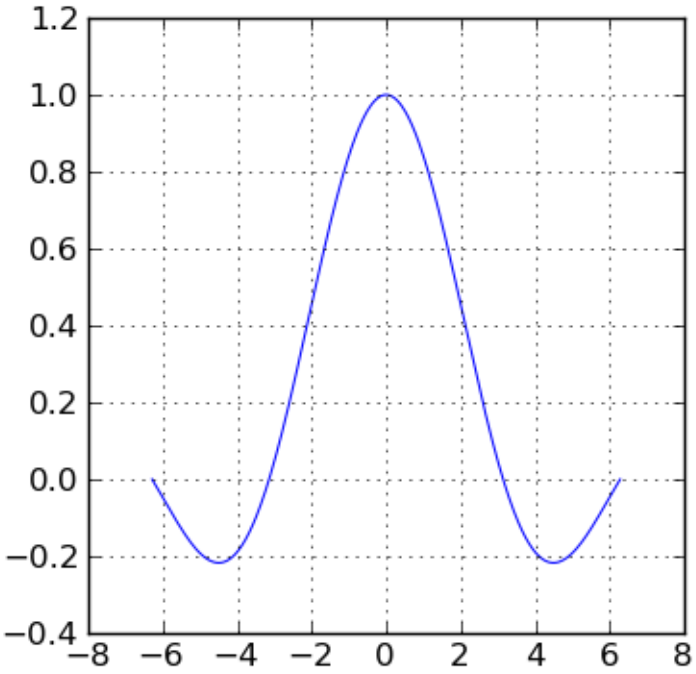
Un peu de graphisme

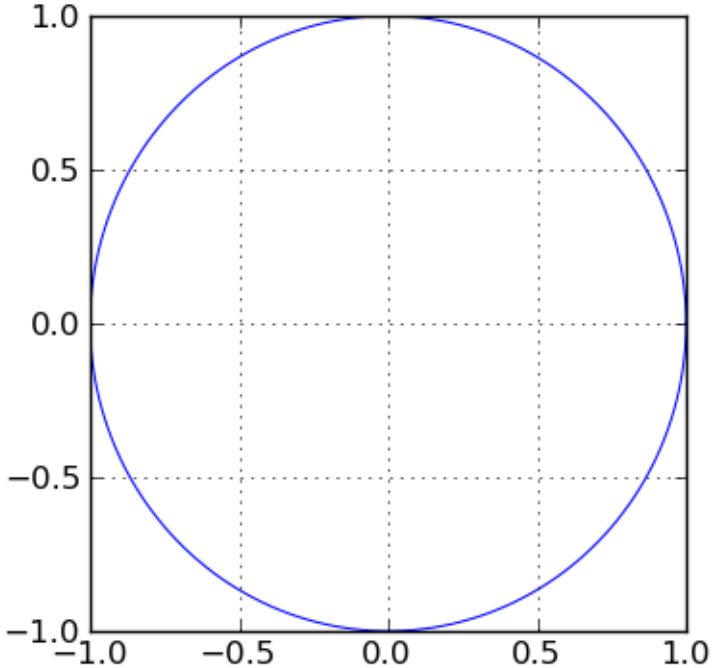
Pour les graphiques, on n'admet qu'une seule commande dans `giac`....

Il n'y a pas encore d'options (couleurs, epaisseur de ligne, taille, déscriptions, grille... mais ça va s'arranger...).

```
> plot(sin(x)/x, x=-2*pi..2*pi);
```



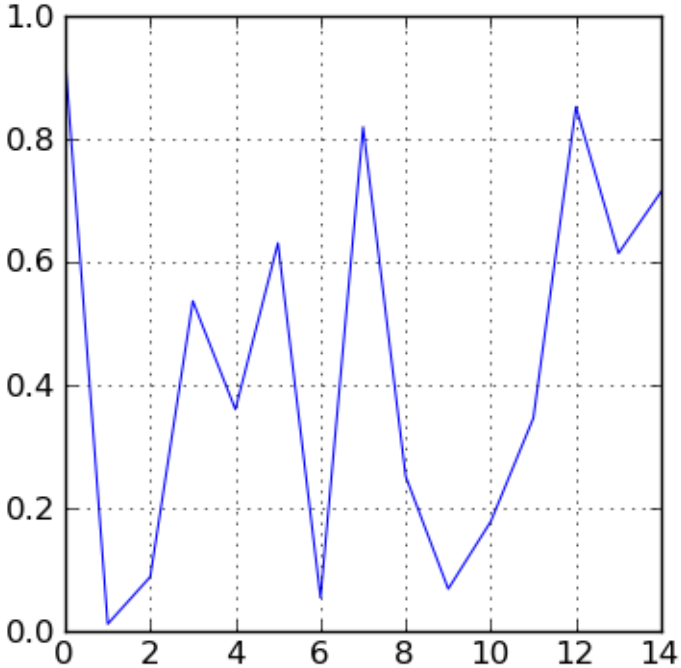

```
> paramplot([cos(t), sin(t)], t=0..2*pi)
```




```
> xx := [seq( rand(0,1), j=1..15)]:;
```

"Done"

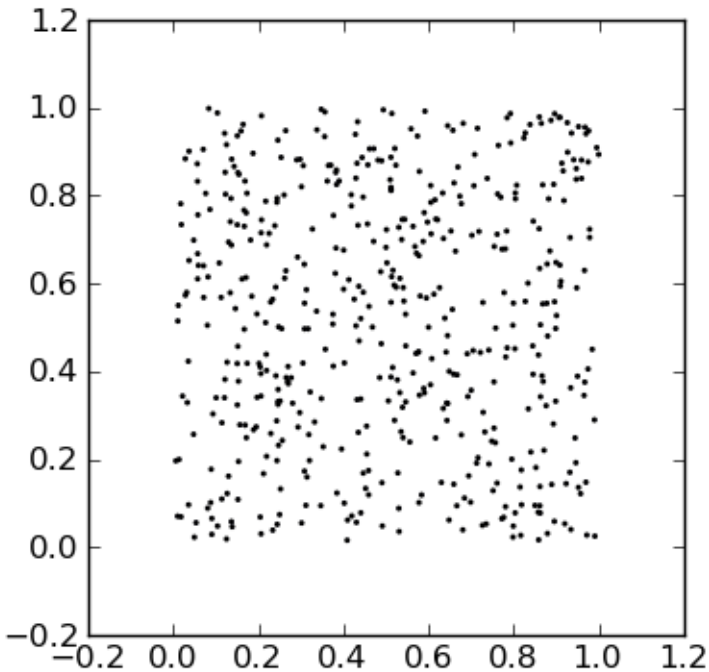
```
> listplot(xx)
```




```
> xx := [ seq( [rand(0,1), rand(0,1)], j=1..500)]:;
```

"Done"

```
> scatterplot(xx)
```

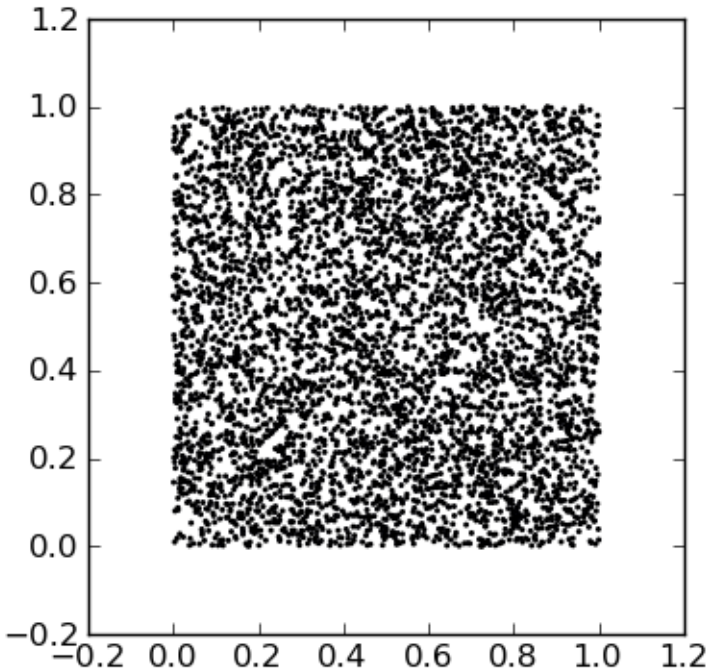


Pour des nuages plus lourds il vaut mieux ne pas appeler `scatterplot` mais traiter directement son argument. On peut le faire avec `py_scatterplot` :

```
> xx := [ seq( [rand(0,1), rand(0,1)], j=1..5000)]:;
```

"Done"

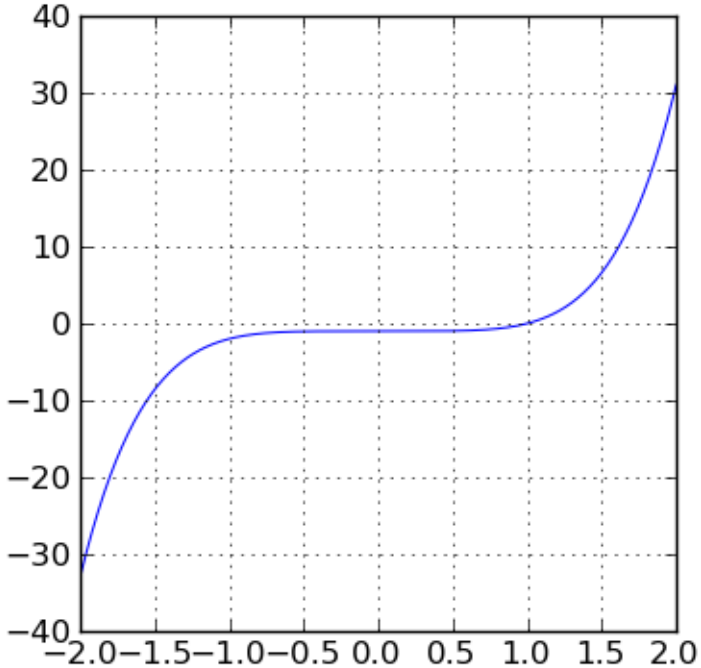
```
> py_scatterplot(xx)
```



Factorisation

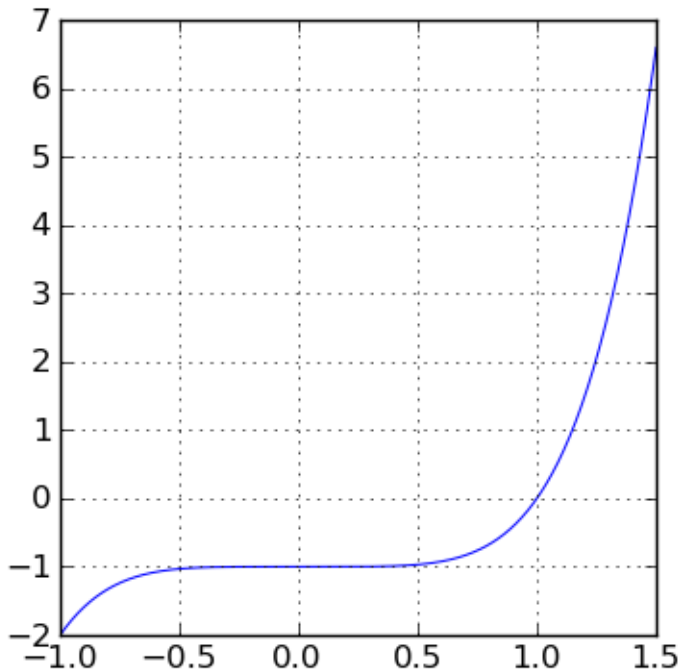
On se propose d'afficher la forme factorisée du polynôme $p := x^5 - 1$ dans $\mathbb{C}$. Il est clair que p possède une racine réelle $x = 1$.

```
> plot(p, x=-2..2);
```



En fait, l'échelle n'est pas très bien choisie. On refait donc

```
> plot(p, x=-1..1.5);
```



Factorisation de $x^5 - 1$ dans le corps des entiers :

```
> s:=factor(p)
```

$$(x-1)(x^4+x^3+x^2+x+1)$$

Et si on cherchait les racines ?

```
> csolve(p=0,x)
```

$$[1, \mathrm{rootof}\left([[1,0,0],[1,-1,1,-1,1]]\right), \mathrm{rootof}\left([[1,0,0,0,0,0,0,0,0],[1,-1,1,-1,1]]\right), \mathrm{rootof}\left(\left(\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 \end{array}\right.\right.$$

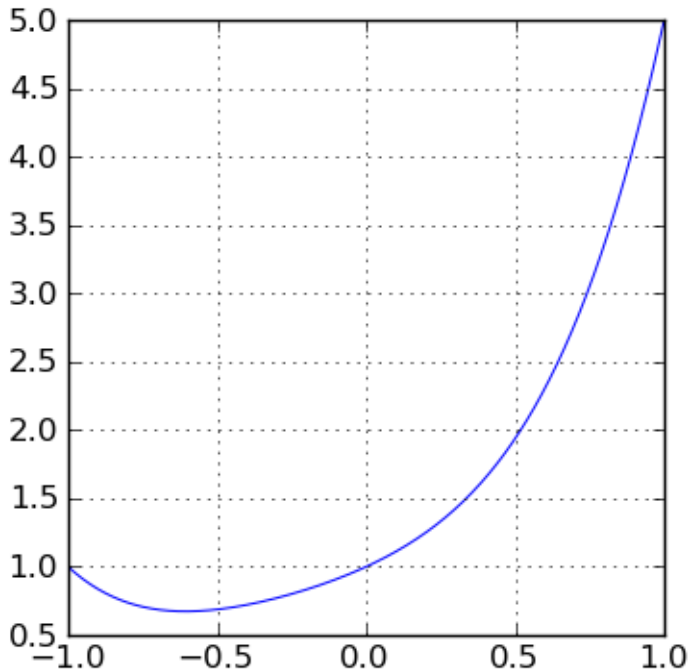
Ce n'est pas ce qu'on cherche... Ce n'est pas toujours évident de résoudre les équations de degré 5.

Voyons si le second terme a des racines réelles.

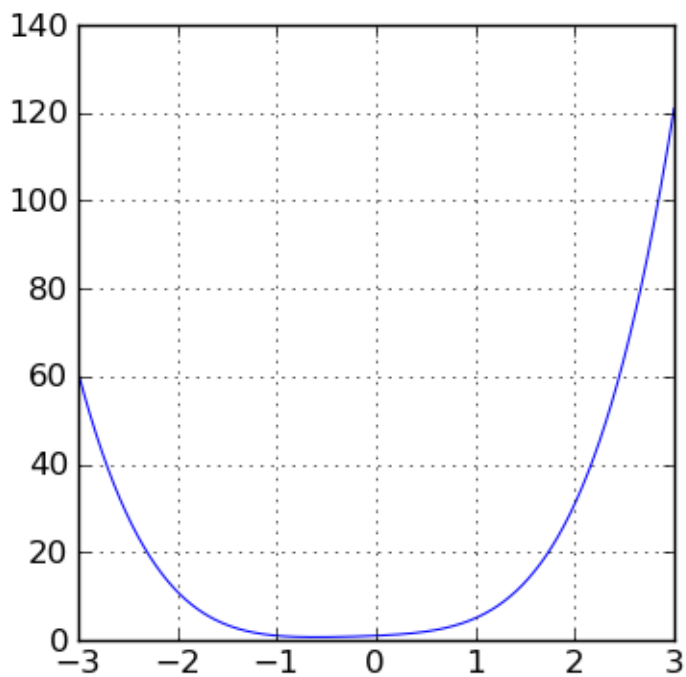
```
> s2 := s[2]
```

$$x^4+x^3+x^2+x+1$$

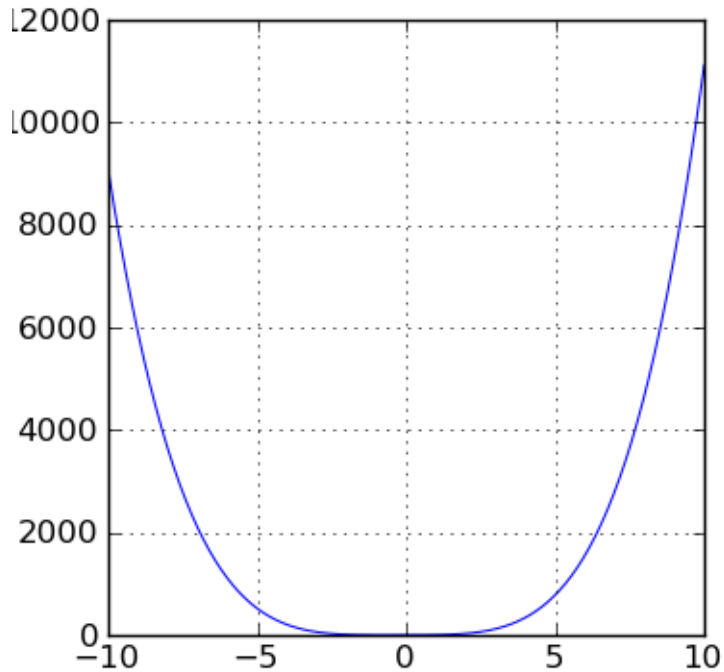
```
> plot(s2, x=-1..1);
```




```
> plot(s2, x=-3..3);
```




```
> plot(s2, x=-10..10);
```



Apparemment non. Bien. On sait pourtant que tout polynôme de degré 4 peut être factorisé dans $\mathbb{R}$. Ce n'est pas difficile, surtout si c'est le logiciel qui fait le calcul ! Définissons $q := (Ax^2+Bx+C)(Dx^2+Ex+F)$. En développant, on obtient $CEx+CDx^2+BFx+BEx^2+BDx^3+AFx^2+AEx^3+ADx^4+CF$. Voici les coefficients de deux polynômes à comparer:

```
> L1:= coeffs(s2,x)
```

$poly1[1,1,1,1,1]$

```
> L2:= coeffs(q,x)
```

$poly1[AD, AE+BD, AF+BE+CD, BF+CE, CF]$

On obtient donc les équations suivantes :

```
> eq0:= L1[0] = L2[0]
```

$1 = (AD)$

```
> eq1:= L1[1] = L2[1]
```

$1 = (AE+BD)$

```
> eq2:= L1[2] = L2[2]
```

$1 = (AF+BE+CD)$

```
> eq3:= L1[3] = L2[3]
```

$1 = (BF+CE)$

Est-ce que `solve` peut résoudre un tel système?

```
> solve([eq0,eq1,eq2,eq3],[A,B,C,D,E,F])
```

$[]$

Apparemment non. Enfin, ceci n'est pas un système linéaire. On va donc chercher une solution sous une forme plus simple. Supposons que $A := 1$ et $D := 1$

```
> solve([eq1,eq2,eq3],[B,C,E,F])
```

[]

Toujours rien. Bien, on a trois équations à quatre inconnues. On peut encore supposer que $F := 1$.

Il vient :

$1 = (E + B)$,

$1 = (1 + BE + C)$,

$1 = (B + CE)$.

```
> sol:=solve([eq1,eq2,eq3],[B,C,E]);
```

[[1,0,0],[-1/2*(sqrt(5)-1),1,1/2*(1+sqrt(5))],[-1/2*(-sqrt(5)-1),1,1/2*(1-sqrt(5))]]

Ca y est ! On remarque qu'un point-virgule à la fin de commande provoque l'affichage direct (tel que dans xcas, sans mise en forme). Sans le point-virgule on obtient la forme matricielle (mise en forme latex) :

$$\begin{pmatrix} 1 & 0 & 0 \\ \frac{-1}{2}(\sqrt{5}-1) & 1 & \frac{1}{2}\cdot(1+\sqrt{5}) \\ \frac{-1}{2}(-(\sqrt{5})-1) & 1 & \frac{1}{2}\cdot(1-(\sqrt{5})) \end{pmatrix}$$

Mais en fait, en supposant $F = 1$ on n'obtient pas un système équivalent. On veut donc maintenant tester si les coefficients trouvés donnent une factorisation de $x^4 + x^3 + x^2 + x + 1$:

```
> q2:=subst(q,[B=1,C=0,E=0])
```

$(x^2 + x)(x^2 + 1)$

```
> expand(q2)
```

$x^4 + x^3 + x^2 + x$

```
> coeffs(q2)
```

$poly1[1, 1, 1, 1, 0]$

Evidemment, ce n'est pas ça... Vérifions alors l'autre solution. En fait, on peut remarquer que par symétrie des coefficients, les deux dernieres lignes représentent les mêmes polynômes.

```
> q2:=subst(q,[B=sol[1][0],C=sol[1][1],E=sol[1][2]])
```

$$(x^2 + \frac{1}{-2}\cdot(\sqrt{5}-1)x + 1)(x^2 + \frac{1}{2}\cdot(1+\sqrt{5})x + 1)$$

Il faut encore développer : $\frac{4x^2}{2\cdot -2} + \frac{\sqrt{5}\,x^3}{-2} + \frac{\sqrt{5}x^3}{2} + \frac{\sqrt{5}x}{-2} + \frac{\sqrt{5}x}{2} + \frac{(-(x^3))}{-2} + \frac{x^3}{2} + \frac{(-x)}{-2} + \frac{x}{2} + x^4 + 2x^2 + 1$.

Si ce n'est pas encore évident, on peut simplifier cette expression à l'aide de `simplify` :

```
> simplify(q2)
```

$x^4 + x^3 + x^2 + x + 1$

ou encore déterminer ses coefficients avec `coeff` : $poly1[1, 1, 1, 1, 1]$.

Ca marche !

On veut maintenant retrouver cette factorisation en utilisant `factor` avec une extension $\sqrt{5}$.

```
> s:=factor(x^5-1,sqrt(5))
```

$$(x-1)(x+\frac{(-2\sqrt{\frac{(-(\sqrt{5})-5)}{2}} - (\sqrt{5})+1)}{4})(x+\frac{(-2\sqrt{\frac{(\sqrt{5}-5)}{2}} + \sqrt{5}+1)}{4})(x+\frac{(2\sqrt{\frac{(-(\sqrt{5})-5)}{2}} - (\sqrt{5})+1)}{4})(x+\frac{(2\sqrt{\frac{(\sqrt{5}-5)}{2}} + \sqrt{5}+1)}{4})$$

OK, mais il y a paraît-il des facteurs complexes. Retrouvons la factorisation dans $\mathbb{R}$. Dans ce but, on va extraire les facteurs :

```
> s1:=s[1]; s2:=s[2]; s3:=s[3]
```

x-1,x+(-2*sqrt((-sqrt(5)-5)/2)-sqrt(5)+1)/4,x+(-2*sqrt((sqrt(5)-5)/2)+sqrt(5)+1)/4

```
> s4:=s[4]; s5:=s[5]
```

x+(2*sqrt((-sqrt(5)-5)/2)-sqrt(5)+1)/4,x+(2*sqrt((sqrt(5)-5)/2)+sqrt(5)+1)/4

Une digression technique : en fait une ligne "multi-commande" n'est pas transformée en LaTeX. Cette syntaxe est donc à éviter sauf si on veut faire des calculs sans rien afficher. Reprenons :

```
> s1, s2, s3
```

$$x-1, x+\frac{(-2\sqrt{\frac{(-(\sqrt{5})-5)}{2}}-(\sqrt{5})+1)}{4}, x+\frac{(-2\sqrt{\frac{(\sqrt{5}-5)}{2}}+\sqrt{5}+1)}{4}$$

(pour une meilleure lisibilité, on affiche un facteur par ligne)

```
> s2
```

$$x+\frac{(-2\sqrt{\frac{(-(\sqrt{5})-5)}{2}}-(\sqrt{5})+1)}{4}$$

```
> s3
```

$$x+\frac{(-2\sqrt{\frac{(\sqrt{5}-5)}{2}}+\sqrt{5}+1)}{4}$$

```
> s4
```

$$x+\frac{(2\sqrt{\frac{(-(\sqrt{5})-5)}{2}}-(\sqrt{5})+1)}{4}$$

```
> s5
```

$$x+\frac{(2\sqrt{\frac{(\sqrt{5}-5)}{2}}+\sqrt{5}+1)}{4}$$

On retrouve plus facilement les facteurs conjugués sous forme "float"

```
> evalf(s)
```

$$(x+-1.000000)(x+-0.309016994375-0.951056516295i)(x+0.809016994375-0.587785252292i)(x+-0.309016994375+0.951056516295i)(x+0.809016994375+0.587785252292i)$$

Il vient

```
> simplify(expand(s2*s4))
```

$$\frac{(2x^2-x\sqrt{5}+x+2)}{2}$$

```
> simplify(expand(s3*s5))
```

$$\frac{(2x^2+x\sqrt{5}+x+2)}{2}$$

Dans ce cas, `normal` est plus efficace :

```
> s24:=normal(expand(s2*s4))
```

$$x^2+\frac{(-(\sqrt{5})+1)}{2}x+1$$

```
> s35:=normal(expand(s3*s5))
```

$$x^2+\frac{(\sqrt{5}+1)}{2}x+1$$

Finalement, on peut afficher la factorisation :

```
> s1*s24*s35
```

$$(x-1)(x^2+\frac{(-(\sqrt{5})+1)}{2}\cdot x+1)(x^2+\frac{(\sqrt{5}+1)}{2}\cdot x+1)$$

L'ultime vérification :

```
> normal(s1*s24*s35)
```

$$x^5-1$$

Quelques tests de plus

boucles, calculs multi-lignes, lignes multi-commandes... :

```
> restart
```

$$[A, D, F, \mathrm{L1}, \mathrm{L2}, \mathrm{eq0}, \mathrm{eq1}, \mathrm{eq2}, \mathrm{eq3}, p, \mathrm{p\_scatter\_data}, q, \mathrm{q2}, s, \mathrm{s1}, \mathrm{s2}, \mathrm{s24}, \mathrm{s3}, \mathrm{s35}, \mathrm{s4}, \mathrm{s5}, \mathrm{sol}, \mathrm{xx}]$$

```
> s:=factor(x^5-1) :; s1:=s[1] :; s2:=s[2] :;
```

"Done"

```
> s1*s2
```

$$(x-1)(x^4+x^3+x^2+x+1)$$

A little little program. Remark the curly brackets (not sure whether this could be handled with regular expressions).

```
> s:=0:; for(j :=1; j<=10; j++) {s += j;}
```

55

This poses a problem of latex formatting.

Multiline command. At this stage, the newline characters are deleted and multiple spaces squeezed to a single space. Hence it is mandatory to separate commands with semicolons.

```
> s:=0:; for(j :=1; j<=10; j++) {s++;}
```

10