

1) **Choisir sa configuration**

⚠ Le jour de l'épreuve les valeurs par défaut ne coïncideront pas forcément avec vos préférences. Au premier lancement on répond à quelques questions, on peut ensuite modifier/affiner ses choix dans le menu Cfg. Puis Sauvez les grâce au menu. Il faut donc choisir le mode xcas ou maple, et étudier la configuration du "cas" pour ne pas mal interpréter les réponses.

NB : Sur une version 32 bits, il y a 2 modes pour la précision où les algorithmes utilisés sont différents : moins de 14 chiffres, et plus de 15 chiffres. (Ex avec 10 chiffres, on travaille tout de même avec 14 même s'il n'y en a que 10 d'affichés).

2) **Les modes xcas et maple**

i et *I* désignent le nombre complexe de carré -1 selon le mode. Les tableaux commencent à 0 contre 1, et le symbole % est réservé aux objets modulaires pour xcas, alors qu'il représente le dernier résultat pour maple. Les symboles { } servent en mode xcas pour regrouper des intructions (par exemple boucle, fonction), alors qu'en mode maple ils sont réservés pour définir un ensemble. Les commentaires sont // resp #. L'instruction subs prend ses arguments dans des ordres inversés. La lettre e est réservée en mode xcas pour exp(1), en mode maple elle est libre.

Ensuite pour les boucles, procedures, ... les 2 syntaxes fonctionnent en mode xcas, et celles qui n'utilisent pas de {} fonctionnent aussi en mode maple.

Pour la configuration du "cas", je conseille de :
 COCHER radians, et de
 DÉCOCHER : approx, complex, Cmpx_var, Sqrt.
 On travaillera ainsi en mode exact dans le corps engendré par les coefficients de l'expression, alors que si l'on coche complex ou Sqrt, on rajoute *I* et des racines carrées ce qui ne permet pas par exemple une factorisation de polynôme sur un corps pres-crit.

3) **Racourcis claviers**

copier CTRL c
 coller: CTRL v et aussi CTRL y ou SHIFT ins
 couper la selection: CTRL x
 undo: CTRL z
 couper ce qui suit: CTRL k (kill)
 effacer toute la ligne: ECHAP
 remonter/descendre dans l'historique:
 ALT fleche haut/bas (parfois CTRL)

Et pour insérer une nouvelle entrée, une partie graphique2D, une partie graphique3D, un programme, un tableur :

ALT n, ALT g, ALT h, ALT p, ALT t.

4) **Supprimer/copier/deplacer des lignes**

Noircir les petis carrés portant le numéro de la ligne en les selectionnant. On peut ensuite deplacer/coller en positionnant la souris sur un autre carré de ce type.

5) **Suites** : Pour créer une suite d'objets séparés par des virgules, on peut utiliser :

seq(trucdej, j=liste)

Exemples :

```
seq(j^2, j=1..5);
L:=[1,4,5,5,6,2];seq(j^2, j=L);
A:=[seq(j^2, j=1..5)];A[3];
```

pour aller de 2 en 2 avec une version récente on peut faire : seq(j^2, j=1..5, 2);

⚠ seq a de nombreuses syntaxes qui sont toutes valables que l'on soit en mode xcas ou maple. Par exemple seq(j^2, j, 1, 4) est la syntaxe dite TI et retournera [1,2,3,4] et non (1,2,3,4) comme les précédentes instructions.

6) **Boucles** : Les deux styles (genre maple ou xcas) fonctionnent en mode xcas la syntaxe est donc indifférente tant que l'on respecte les caractères réservés de chaque mode.

a) (style xcas² genre C)

```
for(j:=1; j<=5; j++){toto;tutu}
```

b) (style xcas genre algo)

```
pour j de 1 jusque 15 pas 2 faire
  print(j);
fpour
```

c) (comme dans maple)

```
for j from 1 to 5 do tutu;toto; od
```

(On peut remplacer od par end do ou end ou end_for mais PAS end_do. On peut préciser le pas avec un by après le to 5)

7) **Tant que** :

a) (style xcas genre C)

```
while(j<5){print(j);j:=j+1}
```

b) (style xcas genre algo)

1. <http://www.math.jussieu.fr/~han/agreg>
 2. Ne fonctionne pas en mode maple à cause des {

```

tantque (j<5) faire
  print(j);j:=j+1;
ftantque

```

c) (style maple)

```

while j<5 do print(j);j:=j+3 end do

```

8) Conditions :

a) (style xcas genre C)

```

if( a=1 ){print("a vaut 1")}

```

pour le if then else, on peut faire :

```

if( ) { } else { }

```

b) (style xcas genre algo)

```

si a=1 alors
  print("1")
sinon print("0")
fsi;

```

c) (style maple)

```

if a=1 then print("a vaut 1\n"); fi

```

On peut aussi faire : `if...then...else...fi` et remplacer le `fi` par `end if`, ou bien `end_if` ou `end`

9) Tests :

a) (à la maple :

égal, différent, inférieur, inférieur ou égal :

```

=      <>      <      <=

```

b) (autre possibilité :)

le test d'égalité est aussi `==`, celui de différence est aussi `!=`

10) Opérateurs logiques :

ou, et, négation :

a) (à la maple :) `or`, `and`, `not`

b) Autre possibilité : `||` `&&` `!`

11) Définition d'une fonction :

```

f:=x->x^2+1;g:=(k,l)->k*l;

```

on peut ensuite faire `f(y)`, ou `f(3)`.

12) Programme/procédure :

a) version xcas :

```

P(a,b):={.....}

```

b) ou bien xcas :

```

fonction P(a,b)
  ...
ffonction;

```

c) version maple :

```

P:=proc(a,b)

```

```

  ...

```

```

end;

```

13) Tableaux, matrices, vecteur :

```

matrix([[1,2],[3,4]]);

```

```

A:=[[1,2],[3,4]];

```

```

matrix(2,2,(k,l)->k*l);

```

⚠ Les coefficients commencent à 1 en mode maple, et à 0 en mode xcas. (Matrix n'existe pas).

a) opérations élémentaires :

```

A*A;A^2; A^(-1);inverse(A); det(A);

```

```

transpose(A); ⚠ mais PAS : A.A.

```

La première ligne de A est obtenue avec : `A[1]` ; Les vecteurs sont en lignes : `v:=[5,6]` ; avec une certaine souplesse pour le produit. Par exemple *produit scalaire* est juste `v*v`, $A(v)$ est `A*v`; et ${}^t v.A$ est `v*A`

b) `A:=matrix(3,3)+1`; fait maintenant comme sous maple. (1 est l'identité, avant c'était la matrice de même taille avec des 1 partout. Soyez donc prudent avec cette syntaxe)

14) Astuces :

a) En mode maple, le dernier résultat évalué : `%` mais il doit souvent être isolé par des espaces. Ex `evalf(%)`;

b) ⚠ lorsque l'on ouvre une feuille contenant de la géométrie, il vaut mieux faire un `Edit>Executer>Session` ou bien `F9` pour que tout soit évalué, sinon il y a un risque de plantage.

15) Calcul modulaire.

a) EN MODE XCAS vous travaillez avec les fonctions usuelles car il y a des objets modulaires. Ex `P:=x^2+1%2;factor(P)`;

Astuce : Si l'on a besoin de remonter un modulaire en un entier, alors on le passe modulo zéro : Par exemple en mode xcas : `a:=3%7;a%0`; NB : xcas fait automatiquement les puissances rapides si elles sont assez grandes. (à partir de 2^{31} ? Attention autour de 2^{30} il ne le fait pas et ça peut être long ou poser problème, dans ce cas mieux vaut utiliser `powmod`)

```

7^(12^50) mod 101;

```

16) Créer une fonction à partir d'un symbole

```

P:=x^2+1; f:=unapply(P,x); f(3);

```

On peut aussi **substituer** une variable par quelque chose : utiliser `subs` (⚠ : l'ordre des arguments est différent entre les modes xcas et maple) ou `subst` qui lui se comporte toujours de la même manière en mode xcas et maple.

17) **Obtenir une écriture binaire**; Par exemple, pour obtenir l'écriture de 7 en base 2, on peut utiliser `convert` et renverser l'ordre.

```
l:=convert(7,base,2); v:=revlist(1);
```

On veut parfois **compléter par des zéros** pour avoir une taille constante, il suffit d'ajouter un vecteur nul de la taille totale voulue, par exemple :

```
v:=[1,2];v:=v+[0$5];
```

donnera [1,2,0,0,0]. (0\$5 voulant dire 0 répété 5 fois.)

18) **Suites, listes, et ensembles** Dans xcas les objets avec parenthèses sont à plat : essayer : `eval((1,(2,3),3)`; Ceux entre crochets n'y sont pas et donnent la possibilité de faire des listes de listes, c'est par exemple le cas d'une matrice. Ex : `eval([1,[2,3],3])`;

Pour les ensembles, en mode maple les `{}` suffisent : `{1,2,3,3}` . En mode xcas il faut un `%`. Exemple : `%{1,2,3,3%}` ou bien `set[1,2,3,3]`

Astuce : On peut trier, sélectionner des éléments avec une fonction de tri arbitraire que l'on donne en argument. (cf `sort` et `codeselect`). Par exemple pour trier la liste de vecteurs suivante selon leur première coordonnée : `l:=[[1,3],[2,5],[0,6]]` `sort(l,(x,y)->x[1]>=y[1])`; (En mode maple)

19) Créer une fonction à partir d'un symbole : la méthode la plus fiable est d'utiliser `unapply`. par exemple : `P:=x^2+a`; `f:=unapply(P,x)`; On peut alors évaluer (f)

20) Pour les matrices par blocs, juxtaposition (à coté ou en dessous) étudier : matrices par blocs, `blockmatrix`, `diag`, `augment`, `concat`. On peut aussi faire des matrices de permutation, (Cf `permutation`).

21) **Géométrie** On peut affecter à une variable un objet géométrique. Les points peuvent être donnés en coordonnées, ou bien par leur affixe complexe. (rappel : en mode maple I est le nombre complexe, en mode xcas c'est i) :

```
d1:=droite(0,1-i); d2:=droite(x+y=0);
```

```
P1:=point(1+i);
```

```
plotparam(cos(t)+i*sin(t),t=-2..2,tstep=0.01)
```

Pour afficher le tout simultanément : `d1,d2,P1`;

Pour afficher l'objet : `objetgeo` en changeant ses attributs :

```
couleur(objetgeo,red+point_width_2+hidden_name),
```

mais aussi de maniere globale on peut faire : `couleur(hidden_name)`;

Tout ceci peut maintenant être obtenu par des exemples dans le menu "Graphic".

ASTUCES : Le menu "Graphic"

-La fonction "Attribut" du menu "Graphic" insère dans votre ligne de commande les options d'affichages de votre objet géométrique que vous avez choisies à la souris. (ALT k)

-On peut cliquer sur la petite icône de couleur (souvent magenta) qui apparaît dans de nombreuses fonctions du menu "Graphic". (Ex : `Graphic>Courbes>plotparam`)

22) **Stats** Pour faire des stats sur la suite : `l:=seq(rand(1001),n=1..100)`; on doit d'abord regrouper ses éléments en classes. Pour des classes de largeur constante, on donne le debut de la première classe (ici 0) et la largeur des intervalles (ici 1001/5) :

```
classes(1,0,1001/5),
```

```
histogram(classes(1,0,1001/5));
```

pour avoir juste les cardinaux des classes il vaut mieux `diagramme_batons`

23) **Le calcul modulaire en mode maple**

INUTILE SI L'ON EST EN MODE XCAS!

C'est pour éviter cela qu'il vaut mieux être en mode xcas.

on reste toujours dans \mathbb{Z} , il faut donc utiliser des instructions particulieres que l'on devine ainsi : Le nom de l'instruction usuelle mais avec une majuscule. Ex :

```
factor(P);Factor(P) mod 2;
```

On peut cependant obtenir un objet modulaire bienque l'on soit en mode maple : Le plus naturel est de passer temporairement³ en mode xcas, de créer le 1 de $\mathbb{Z}/N\mathbb{Z}$ puis de revenir en mode maple.

```
maple_mode(0); a:=1%2; b:=x^2+3%7;
c:=b%0;maple_mode(1);
```

on peut utiliser la fonction⁴ `makemod` :

```
a:=makemod(1,2); factor(x^2+a);
```

où bien via `pari` : On charge `pari` une fois pour toute avec `pari()`; puis on peut faire `a:=Mod(1,2)`; et utiliser `a` dans xcas même si l'on est en mode maple.

Exemples d'instructions très utiles si l'on est en mode maple :

```
Gcd, Gcdex, Rem(P,Q,x) mod 2;, Powmod(P,Q,x)
mod 2; Nullspace(A) mod 2;
```

3. Soit par menu déroulant, soit avec l'instruction `maple_mode`

4. qui n'a pas l'air documentée