# Glossaire Xcas

Renée De Graeve Bernard Parisse

Octobre 2009

## 1 Pour écrire une fonction ou un programme

Vérifier au préalable soit avec le menu `Cfg->Configuration du CAS`, soit en cliquant sur la bouton `Config` en haut de la session, que :

- vous avez choisi la syntaxe en mode `Xcas`,
- l'unité d'angle est la bonne pour des programmes de géométrie, en cochant ou décochant `radian` dans la fenêtre de configuration qui s'ouvre en appuyant sur la barre `Config`,

Puis :

1. ouvrir un niveau éditeur de programme soit en tapant `Alt-p`, soit avec le menu `Prg->Nouveau programme`. Il contient déjà le `:;` qui doit terminer le programme.
2. taper la fonction en terminant chaque instruction par `;`. Les noms de cette fonction, de ses arguments, de ses variables locales ne doivent pas déjà être utilisés par `Xcas`. On peut commencer le nom des fonctions par une Majuscule pour diminuer le risques de conflits avec une fonction qui existe déjà dans `Xcas`. Notez que dans un niveau éditeur de programmes, les mots clés apparaissent en bleu et les nomms de commandes `Xcas` apparaissent en brun.
3. appuyer sur `OK(F9)` (touche F9), pour compiler le programme.
4. pour exécuter le programme, on se place dans une ligne de commande vide, on tape le nom de la fonction suivi entre parenthèses par les valeurs des paramètres séparées par des virgules. Pour l'exécuter en mode pas à pas, on précède le nom de la fonction de `debug(` et on clot la parenthèse à la fin.

## 2 Le menu `Add` d'un niveau éditeur de programme

Ce menu vous permet d'avoir facilement la syntaxe d'une fonction, d'un test et des différentes boucles. Par exemple une fonction s'écrit avec la syntaxe suivante :

```
f(x,y):={
  local z,a,b,...,val;
  instruction1;
  .....
  instructionk;
};
```

On termine par `:;` pour que la réponse à une compilation reussie soit `Done` ou par `;` pour avoir en réponse, la traduction du programme après la compilation.

# 3 Les instructions en français

| Instructions en fançais | |
|---|---|
| affectation | `a:=2;` |
| entrée expression | `saisir("a=",a);` |
| entrée chaine | `saisir_chaine("a=",a);` |
| sortie | `afficher("a=",a);` |
| valeur retournée | `retourne a;` |
| arrêt dans boucle | `break;` |
| alternative | `si <condition> alors <inst> fsi;` |
| | `si <condition> alors <inst1> sinon <inst2> fsi;` |
| boucle pour | `pour j de a jusque b faire <inst> fpour;` |
| | `pour j de a jusque b pas p faire <inst> fpour;` |
| boucle répéter | `repeter <inst> jusqua <condition>;` |
| boucle tantque | `tantque <condition> faire <inst> ftantque;` |
| boucle faire | `faire <inst1> si <condition> break;<inst2> ffaire;` |

# 4 Les instructions comme en C

| Instructions comme en C++ | |
|---|---|
| affectation | `a:=2;` |
| entrée expression | `input("a=",a);` |
| entrée chaine | `textinput("a=",a);` |
| sortie | `print("a=",a);` |
| valeur retournée | `return a;` |
| arrêt dans boucle | `break;` |
| alternative | `if (<condition>) {<inst>};` |
| | `if (<condition>) {<inst1>} else {<inst2>};` |
| boucle pour | `for (j:= a;j<=b;j++) {<inst>};` |
| | `for (j:= a;j<=b;j:=j+p) {<inst>};` |
| boucle répéter | `repeat <inst> until <condition>;` |
| boucle tantque | `while (<condition>) {<inst>};` |
| boucle faire | `do <inst1> if (<condition>) break;<inst2> od;` |

# 5 Les instructions en mode Maple

| Instructions en mode Maple | |
|---|---|
| alternative | `if <condition> then <inst> fi;` |
| | `if <condition> then <inst1> else <inst2> fi;` |
| boucle pour | `for j from a to b do <inst> od;` |
| | `for j from a to b by p do <inst> od;` |
| boucle tantque | `while <condition> do <inst> od;` |

(Attention, l'instruction `while ... do ... od;` nécessite d'avoir choisi la syntaxe en mode compatible Maple).

# 6 Les opérateurs

| Opérateurs | |
|---|---|
| `+` | addition |
| `-` | soustraction |
| `*` | mutiplication |
| `/` | division |
| `^` | puissance |
| `==` | teste l'égalité |
| `!=` | teste la différence |
| `<` | teste la stricte infériorité |
| `<=` | teste l'infériorité ou l'égalité |
| `>` | teste la stricte supériorité |
| `>=` | teste la supériorité ou l'égalité |
| `ou` | opérateur booléen infixé |
| `et` | opérateur booléen infixé |
| `non` | renvoie l'inverse logique de l'argument |
| `vrai` | est le booléen vrai ou true ou 1 |
| `faux` | est le booléen faux ou false ou 0 |

## 6.1 Séquences, listes et chaines de caractères

| Séquences et listes | |
|---|---|
| `S:=a,b,c` | $S$ est une séquence de 3 élements |
| `L:=[a,b,c]` | $L$ est une liste de 3 élements |
| `S:=NULL` | $S$ est une séquence de 0 élement |
| `L:=[]` | $L$ est une liste de 0 élement |
| `dim(S)` | renvoie le nombre d'éléments de $S$ |
| `S[0]` | renvoie le premier élément de $S$ |
| `S[n]` | renvoie le $n+1$ unième élément de $S$ |
| `S[dim(S)-1]` | renvoie le dernier élément de $S$ |
| `S:=S,d` | ajoute l'élément $d$ à la fin de la séquence $S$ |
| `L:=append(L,d)` | ajoute l'élément $d$ à la fin de la séquence $L$ |

| Chaines de caractères | |
|---|---|
| `S:="abc"` | $S$ est une chaine de 3 caractères |
| `S:=""` | $S$ est une chaine de 0 caractère |
| `dim(S)` | renvoie le nombre de caractères de $S$ |
| `S[0]` | renvoie le premier caractère de $S$ |
| `S[n]` | renvoie le $n+1$ unième caractère de $S$ |
| `S[dim(S)-1]` | renvoie le dernier caractère de $S$ |
| `S+d` | ajoute le caractère $d$ à la fin de la chaine $S$ |

# 7 Les fonctions utilisées

| Fonctions mathématiques | |
|---|---|
| `floor(t)` | renvoie la partie entière $t$ |
| `round(t)` | renvoie l'entier le plus proche de $t$ |
| `irem(a,b)` | renvoie le reste de la division de $a$ par $b$ |
| `iquo(a,b)` | renvoie le quotient de la division de $a$ par $b$ |
| `abs(t)` | renvoie la valeur absolue de $t$ |
| `sqr(t)` | renvoie la racine carrée de $t$ |

| Fonctions de géométrie | |
|---|---|
| `A:=point(a,b)` | point $A$ de coordonnées $(a,b)$ |
| `affichage=` `epaisseur_point_5` | 3ième argument de `point` pour le dessiner avec une croix d'épaisseur 5 |
| `triangle(A,B,C)` | triangle $ABC$ |
| `bissectrice(A,B,C)` | bissectrice de l'angle $A$ du triangle $ABC$ |
| `angle(A,B,C)` | valeur de la mesure (radians ou degrés) de l'angle $A$ du triangle $ABC$ |
| `mediane(A,B,C)` | médiane issue de $A$ du triangle $ABC$ |
| `mediatrice(A,B)` | médiatrice de $AB$ |
| `cercle(A,r)` | cercle de centre $A$ et de rayon $r$ |
| `cercle(A,B)` | cercle de diamètre $AB$ |
| `rayon(c)` | rayon du cercle $c$ |
| `centre(c)` | centre du cercle $c$ |
| `distance(A,B)` | longueur de $AB$ |
| `distance(A,d)` | distance de $A$ à la droite $d$ |
| `inter(G1,G2)` | liste des points de $G1 \cap G2$ |
| `inter_unique(G1,G2)` | un des points de $G1 \cap G2$ |
| `rotation(A,t,B)` | point transformé de $B$ par la rotation de centre $A$ et d'angle $t$ |