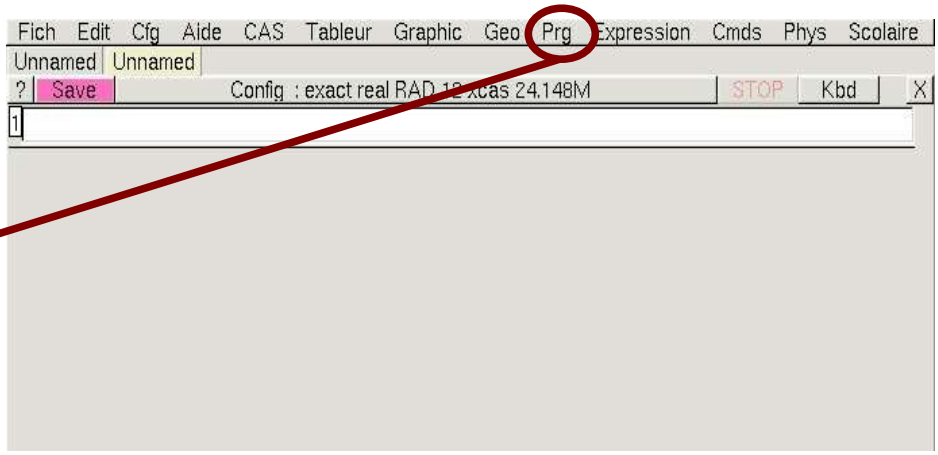
Programmes sous XCAS

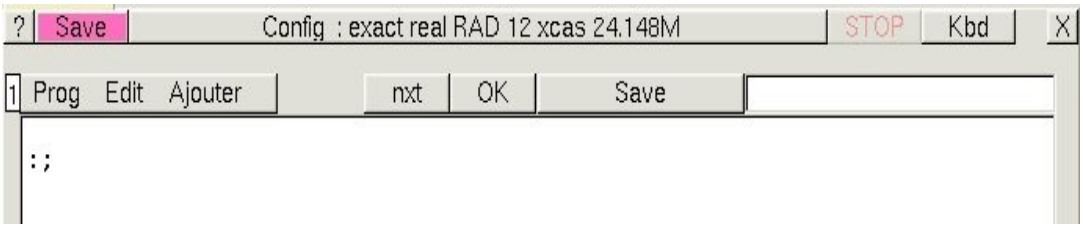
1. Interface de XCAS

XCAS présente des lignes de commandes dans lesquelles on effectue les calculs. L'interface est très proche du logiciel Derive.



Pour écrire un programme, ouvrir le menu « Prg » et choisir « nouveau programme ».

La barre de saisie est alors modifiée comme suit :



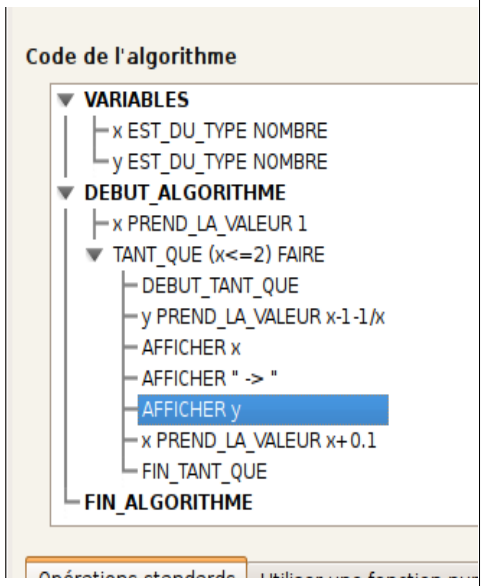
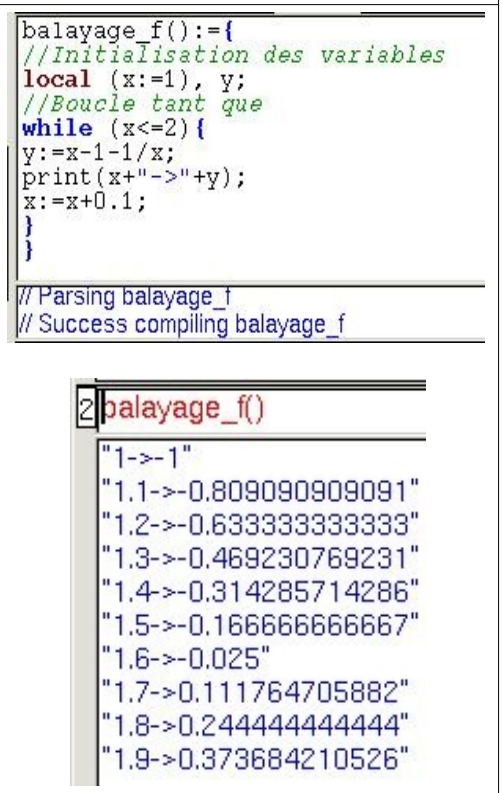
Le programme s'écrit dans la fenêtre blanche, en terminant chaque instruction par « ; ».
Une fois fini, le programme se teste en cliquant sur le bouton « OK ».

Remarque : si les sessions XCAS s'enregistrent avec l'extension .xws, les programmes peuvent s'enregistrer séparément en cliquant sur le bouton « Save » à côté de « OK ». L'extension du fichier est alors : .cxx. On recharge alors un programme en cliquant sur « Prog + Charger ».

Il y a deux manières d'utiliser un programme :
a) seul, en cliquant simplement sur « OK » pour l'exécuter.
b) en le définissant comme une fonction, ce qui nécessite un peu plus de rigueur d'écriture. On peut alors l'appeler dans une quelconque ligne de commande de XCAS. Nous privilégierons cette approche.

2. Ecrire et tester un programme simple

Reprenons le programme de calcul de valeurs d'une fonction par balayage :

Algobox	Comparatif	XCAS
	Pour être certain de ne pas utiliser de données issues d'un calcul précédent, les variables d'un programme sont *locales*. y est initialisée par défaut à 0; x reçoit la valeur 1. On affiche un résultat par la commande *print*. La concaténation des résultats de x, de la chaîne de caractères « - > », et de y, se fait avec des « + ». Le programme est compilé lorsqu'on clique sur « OK »; on l'appelle alors dans la ligne de commande : *balayage_f()*.	


```
balayage_f():={
//Initialisation des variables
local (x:=1), y;
//Boucle tant que
while (x<=2){
y:=x-1-1/x;
print(x+"->"+y);
x:=x+0.1;
}
}
// Parsing balayage_f
// Success compiling balayage_f
```

```
2 balayage_f()
"1->-1"
"1.1->-0.809090909091"
"1.2->-0.633333333333"
"1.3->-0.469230769231"
"1.4->-0.314285714286"
"1.5->-0.166666666667"
"1.6->-0.025"
"1.7->0.111764705882"
"1.8->0.244444444444"
"1.9->0.373684210526"
```

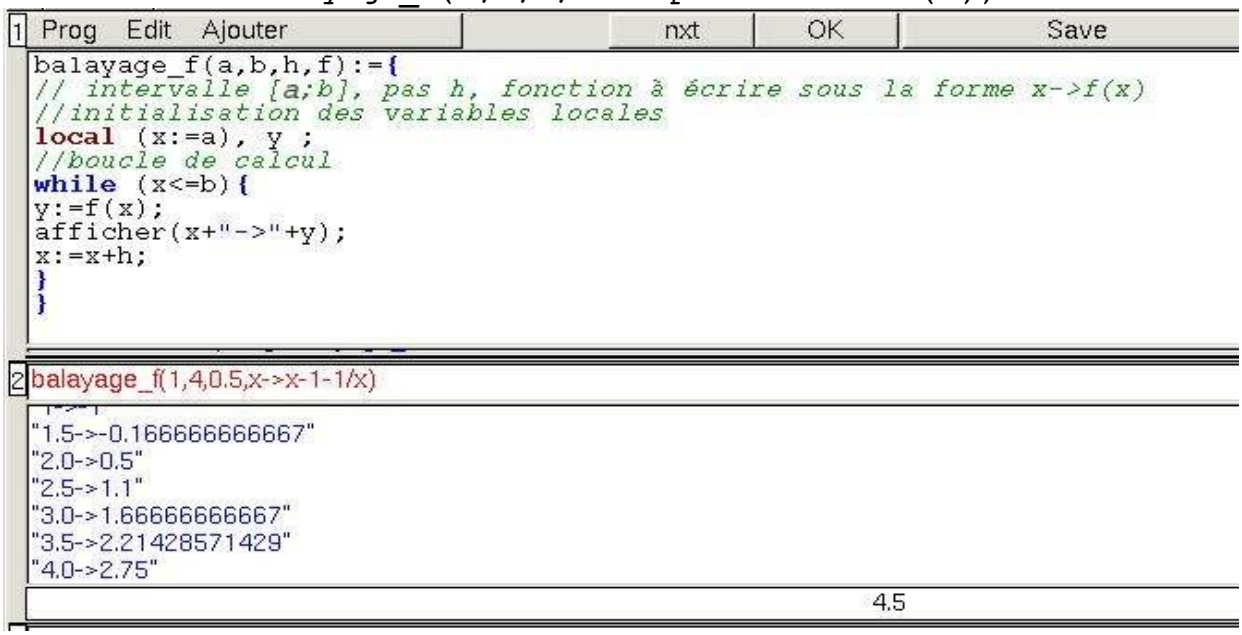
3. Programmes et paramètres

Le programme ci-dessus peut s'améliorer par l'utilisation de paramètres :

- les bornes *a* et *b* de l'intervalle de définition de la fonction;
- le pas *h* du balayage;
- l'expression de la fonction, enfin.

Ces paramètres peuvent être passés dans l'appel du programme :

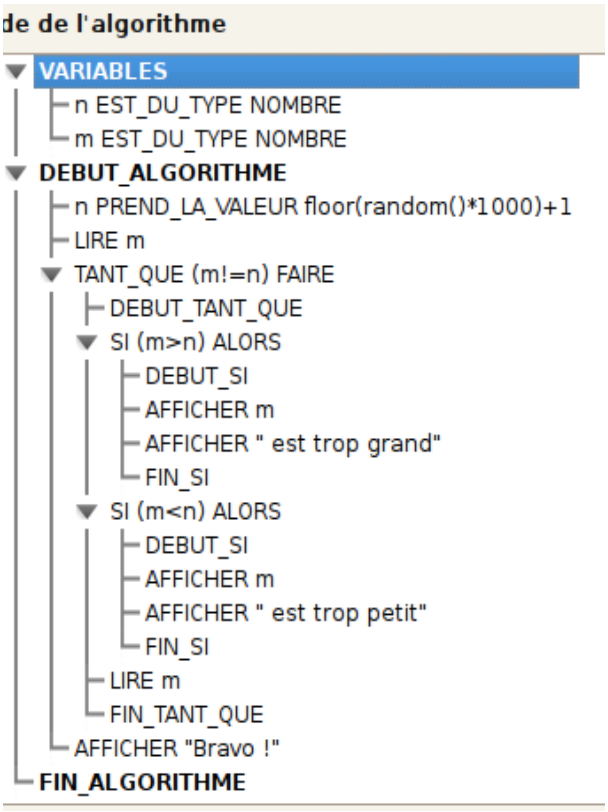
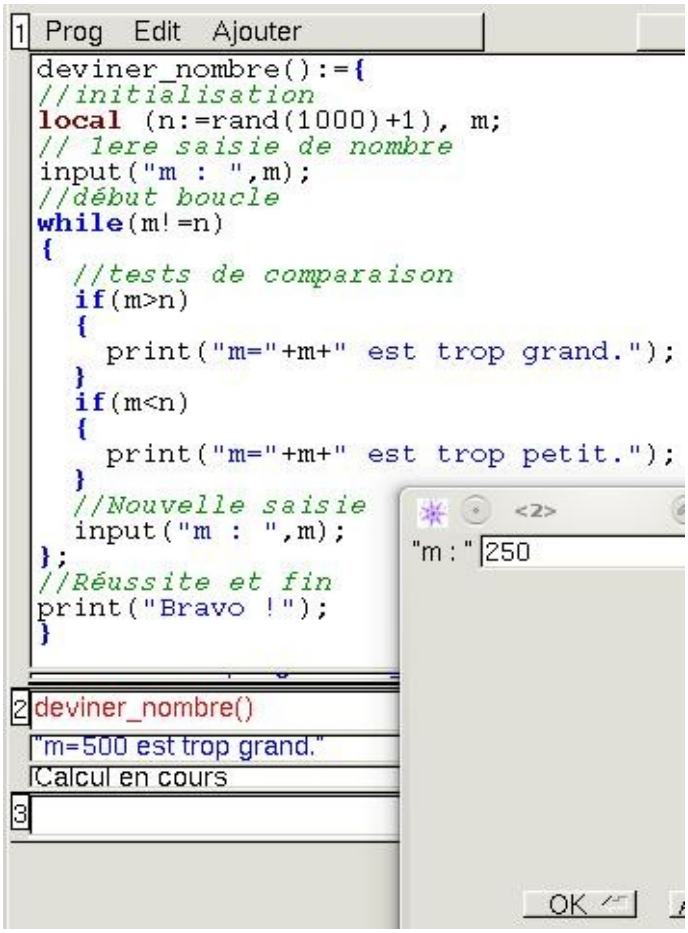
```
balayage_f(a,b,h,x->expression de f(x))
```



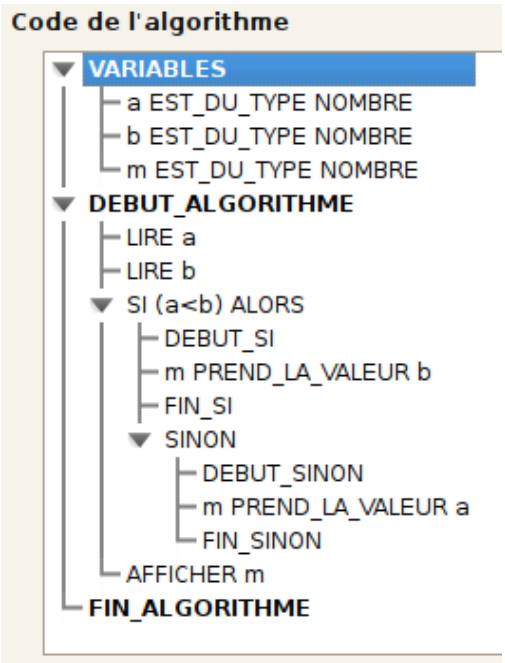
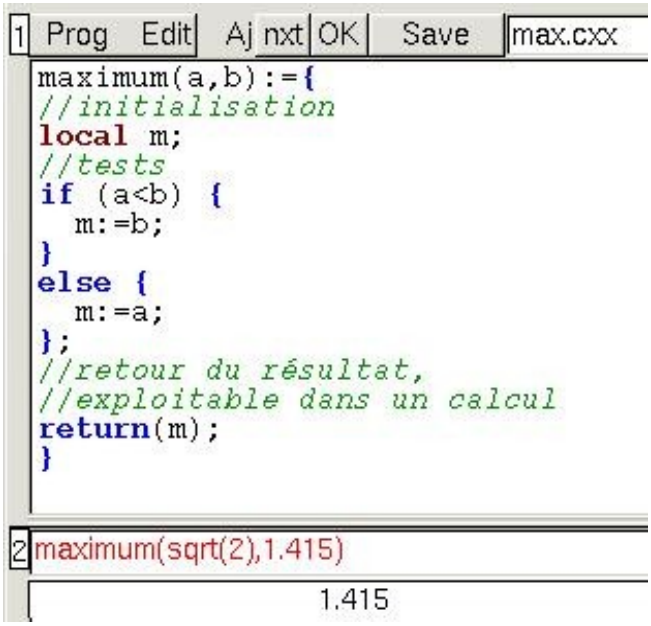
```
1 Prog  Edit  Ajouter                    nxt   OK   Save
balayage_f(a,b,h,f):={
// intervalle [a;b], pas h, fonction à écrire sous la forme x->f(x)
//initialisation des variables locales
local (x:=a), y ;
//boucle de calcul
while (x<=b){
y:=f(x);
afficher(x+"->"+y);
x:=x+h;
}
}

2 balayage_f(1,4,0.5,x->x-1-1/x)
"1.5->-0.166666666667"
"2.0->0.5"
"2.5->1.1"
"3.0->1.66666666667"
"3.5->2.21428571429"
"4.0->2.75"
                                          4.5
```

4. Autres algorithmes

a) Deviner un nombre

Algobox	XCAS
	

b) Maximum de deux nombres

Algobox	XCAS
	

c) Dessin d'une rosace avec création d'une variable en liste, sous XCAS