
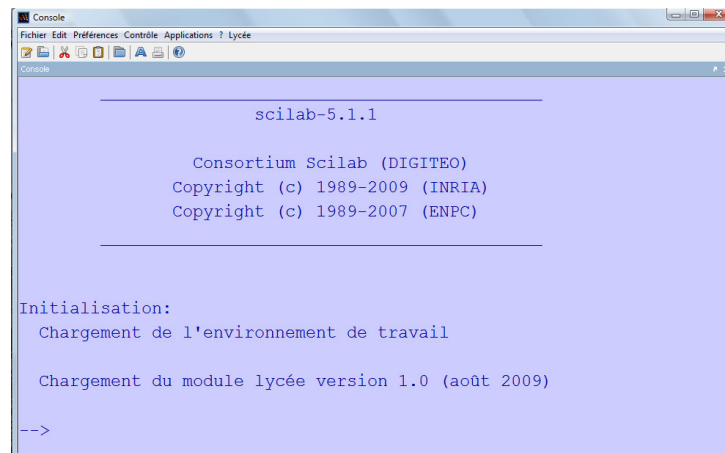


Scilab est un logiciel de calcul utilisé dans le monde de l'entreprise et les universités. Il a été complété par un module « lycée » pour être plus adapté à l'enseignement au lycée. On peut le télécharger gratuitement sur le site <http://www.scilab.org/lycee>.

Le travail se fait sur 3 écrans différents :

## La console

Quand on clique sur l'icône  , on voit apparaître la page suivante qu'on appelle feuille de calcul ou console.



On peut écrire des instructions directement dans la console après la flèche -->, puis taper la touche Entrée pour que le calcul s'effectue. **ans** signifie answer (réponse).

```
-->57/4
ans =
    14.25

-->(2+9)^5
ans =
    161051.
```

On peut revenir en arrière avec les flèches du clavier ←↵→ ou avec la souris, et modifier les instructions.

*S'entraîner à écrire dans la console, à reprendre ou à rectifier des calculs*


```
-->a=5;
-->2*a-7
ans =
    3.

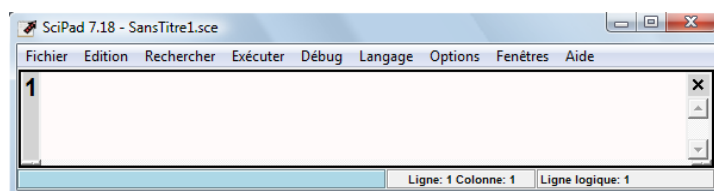
-->sqrt(a)
ans =
    2.2360679774998

-->2*a-10
ans =
    0.
```

Écrire directement dans la console a deux inconvénients : on ne peut pas enregistrer ce qu'on a fait, et, si on a écrit plusieurs lignes d'instructions, les modifications ne sont pas très faciles à réaliser.

Dès qu'on veut donner plusieurs instructions, on aura donc intérêt à les écrire dans l'éditeur.

Pour cela, dans la barre de menu en haut de la console, cliquer sur la première icône : . L'éditeur s'ouvre. Le fichier par défaut dans l'éditeur s'appelle **SansTitre1.sce**.



Ecrivons alors nos instructions. On peut les modifier comme dans n'importe quel traitement de texte.

Dans l'exemple suivant, à partir d'un nombre  $x$ , on fait calculer un nombre mystère, qu'on appellera  $y$  :

|  |   |           |
|--|---|-----------|
| Prendre un nombre $x$ (par exemple 35) | 1 | $x=35;$   |
| Lui ajouter 2                          | 2 | $y=x+2;$  |
| Multiplier le résultat par 3           | 3 | $y=y*3;$  |
| Ajouter $x$                            | 4 | $y=y+x;$  |
| Ajouter 22                             | 5 | $y=y+22;$ |
| Diviser le résultat par 4              | 6 | $y=y/4;$  |
| Enlever $x$ .                          | 7 | $y=y-x;$  |
| Afficher le nombre obtenu              | 8 | $y$       |

Il faut ensuite les copier dans la console. Pour cela plusieurs possibilités :

- Sélectionner la partie à copier, puis faire un Copier/Coller  
Dans ce cas le programme est écrit dans la console, il faut taper « entrée » pour l'exécuter.

Ou bien

- Dans la barre du haut, choisir le menu Exécuter-Charger dans Scilab (raccourci Ctrl+L). Dans ce cas le programme n'est pas écrit dans la console, il est exécuté en entier.

*Refaire l'exemple avec une autre valeur de  $x$ .*

On peut ensuite enregistrer ce fichier en cliquant dans Fichier – Enregistrer sous....

## Exercice

Taper dans la console **%pi** puis « entrée » . Que vaut **%pi** ?

Ecrire dans l'éditeur, puis faire exécuter le programme suivant :

```
R=4 ;
P=2*%pi*R ;
S=%pi*R^2 ;
afficher ([P,S])
```

Que calcule ce programme ?

## La fenêtre graphique

Elle s'ouvre dès qu'on demande un tracé.

## Scilab travaille sur des matrices (tableaux rectangulaires de nombres)

- Un nombre réel sera une matrice  $1 \times 1$ .
- Une suite  $u$  de  $n$  termes sera une matrice  $n \times 1$  ou un vecteur à  $n$  coordonnées, données par  $u(1), \dots, u(n)$ .
- Les nombres particuliers comme  $e$ ,  $\pi$ ,  $i$  se notent respectivement `%e`, `%pi` et `%i`.

## Le ;

- Il permet de ne pas afficher les résultats.
- Il permet aussi d'écrire plusieurs instructions à la suite sans aller à la ligne en les séparant par ;.

## Les opérations

- Ne pas les oublier, en particulier la multiplication : `+`, `-`, `*`, `/`, `^`.

## Les fonctions classiques

- `sqrt`, `exp` et `ln` sont les fonctions racine carrée, exponentielle et logarithme népérien.

## Pour effacer

- `clf` efface la fenêtre graphique.
- `clear u` efface le vecteur  $u$ .
- Ne pas utiliser `clear` tout seul, cela efface toutes les fonctions, y compris celles du module lycée.

## Pour faire une boucle

- `for n=1:50 ... end` permet de faire une boucle avec  $n$  variant de 1 à 50 avec un pas de 1.  
Si on veut un pas de 3, écrire `for n=1:3:50 ... end`
- `while ... end` permet de faire une boucle Tant Que.
- `while %T ... end` réalise une boucle sans fin.
- `break` permet de sortir d'une boucle.

## Pour faire un test

- `if ... then ... else ... end` réalise un test conditionnel.
- Dans un test, `==` signifie égal, `<>` signifie différent.
- `%T` signifie vrai, `%F` signifie faux.
- `&` (touche 1) signifie « et », `|` (touche AltGr+6) signifie « ou », `~` (touche AltGr+2) signifie « non ».
- `<`, `>`, `<=`, `>=` signifient plus petit, plus grand, plus petit ou égal, plus grand ou égal.

## Pour afficher

- `u` affiche les valeurs du vecteur  $u$  en colonne. `u(50)` affiche  $u(50)$ .
- `afficher([u,v])` affiche la matrice  $[u,v]$  de première colonne  $u$  et de deuxième colonne  $v$ .
- Pour afficher une phrase mettre une chaîne de caractères : `afficher("Bob a gagné.")`.
- Pour afficher une phrase avec des nombres dedans, il faut transformer les nombres en chaînes de caractères avec la fonction `string` et utiliser `+` pour concaténer : par exemple `afficher("Bob a gagné "+string(n)+" fois")`

ATTENTION, si la phrase contient une apostrophe, il faut la doubler : `"C' 'est juste"`.

Quand il y a beaucoup de valeurs à afficher, Scilab en donne quelques unes et demande s'il doit continuer : taper la touche `n` pour non et n'importe quelle autre touche pour oui. Tant qu'on ne répond pas, il attend.

## Pour faire tracer un nuage de points

- `plot(u, "r+")` trace le nuage des points  $(n, u(n))$  sous forme de croix rouges.
- Les couleurs sont : le bleu par défaut, `k` = noir, `b` = bleu, `r` = rouge, `g` = vert, `c` = cyan, `m` = magenta, `y` = jaune, `w` = blanc.
- Les styles de points sont : relié par défaut, ou `.`, `+`, `o`, `x`, `*`.

### Pour définir une fonction de $\mathbb{R}$ dans $\mathbb{R}$ et tracer sa courbe

- `function y = f(x); y = expression en fonction de x; endfunction` définit une fonction à une variable.  
On peut aussi définir des fonctions à plusieurs variables.
- `x = linspace(a,b,1000)` définit un vecteur  $x$  de 1000 valeurs régulièrement espacées entre  $a$  et  $b$ .
- `plot(x,f)` ou `plot(x,f,"g")` permet de tracer la courbe de  $f$ , en bleu si on n'écrit rien, en vert si on le précise, dans l'intervalle défini par l'instruction `x=linspace...`
- Dans la fenêtre graphique, on peut faire des zooms, et, à partir du menu Edit de la fenêtre graphique, on peut changer les propriétés des axes et de la figure.

### Pour repérer des termes particuliers dans un vecteur

- `find(L==valeur)` retourne les rangs des termes du vecteur  $L$  qui sont égaux à la valeur.
- On peut aussi taper `find(abs(L-5)<0.01)` pour trouver pour quels rangs les termes d'une suite s'approchent de 5 à 0,01 près par exemple.

### Pour faire des opérations sur les termes d'un vecteur

- `sum(u)` fait la somme des termes du vecteur  $u$ .
- `taille(u)` donne la taille du vecteur  $u$ .
- `moyenne(u)` donne la moyenne des termes du vecteur  $u$ .
- `unique(u)` supprime les termes de  $u$  qui apparaissent plusieurs fois.
- `zeros(n,1)` définit un vecteur contenant  $n$  termes nuls.

### Pour faire des tirages aléatoires

- `tirage_entier(p,m,n)` retourne un vecteur de  $p$  tirages entiers pris entre  $m$  et  $n$  avec  $p$  entier positif,  $m$  et  $n$  entiers et  $m \leq n$ .
- `tirage_reel(p,a,b)` retourne un vecteur de  $p$  tirages réels pris entre  $a$  et  $b$  avec  $p$  entier positif,  $a$  et  $b$  réels et  $a \leq b$ .
- `frequence(n,s)` retourne la fréquence de  $n$  dans la suite de nombres  $s$  avec  $n$  entier.

### Pour définir un ensemble avec des valeurs associées

- `ens=ensemble("r(1)","r(2)","r(3)","v(1)","v(2)")` définit un ensemble, ici l'ensemble de trois boules rouges numérotées 1, 2, 3 (leurs valeurs) et deux vertes numérotées 1, 2 (leurs valeurs).
- `tirage_ensemble(n,ens)` retourne un ensemble de  $n$  éléments pris parmi ceux de `ens`.
- `valeur(ens)` retourne le vecteur des valeurs des éléments de `ens`.

### Arithmétique

- `quotient(m,n)` retourne le quotient de  $m$  par  $n$  avec  $m$  entier et  $n$  entier non nul.
- `reste(m,n)` retourne le reste de  $m$  par  $n$  avec  $m$  entier et  $n$  entier non nul.
- `pgcd(m,n)`, `ppcm(m,n)` retournent le pgcd et le ppcm de  $m$  et  $n$  avec  $m$  et  $n$  entiers.
- `premier(n)` retourne `%T` si  $n$  est premier et `%F` sinon avec  $n$  entier positif ou nul.
- `diviseurs(n)` retourne tous les diviseurs positifs de l'entier  $n$ .
- `factorise(n)` retourne la décomposition en facteurs premiers de l'entier  $n$ .

### Demander la valeur d'une variable

- Pour demander à l'utilisateur la valeur de la variable  $A$ , écrire `A=input("A = ")`.

### Vitesse de calcul

- `tic; ... ;toc`

`tic` déclenche un chronomètre, et `toc` l'arrête, on a ainsi le temps écoulé pour le calcul effectué.

### Commentaires

- Les commentaires, qui ne sont pas pris en compte dans les calculs, doivent être précédés de `//`.