TP 1 – Introduction à Scilab

1 Présentation

Lorsqu'on lance Scilab, on a plusieurs fenêtres rassemblées en une seule :

- Le navigateur de fichier
- La console
- Le navigateur de variable
- L'historique des commandes
- Le flux d'actualité.

Commencez par fermer le flux d'actualité que nous n'utiliserons pas.

Fenêtre perdue ? Si vous fermez par erreur une des fenêtres ci-dessus, vous pouvez la rouvrir en allant dans : « Application → nom de la fenêtre ». Vous pouvez alors la réintégrer avec les autres fenêtres en la prenant par son bandeau bleu et en la faisant glisser sur les autres fenêtres à l'emplacement qui vous convient.

Fermer la console ? Si vous tentez de fermer la console, un message de confirmation s'affichera vous demandant si vous souhaitez fermer Scilab : **fermer la console, c'est fermer Scilab**.

2 Utilisation de la console

L'invite de commande : La console s'utilise comme une calculatrice. Le symbole `-->` s'appelle « l'invite de commande ». Il indique que Scilab est prêt et vos instructions. Si vous voyez apparaître un autre symbole cela peut signifier plusieurs choses :

- Soit que Scilab est entrain de faire des calculs et il faut donc patienter.
- Soit que Scilab est « bloqué » dans une bouce infinie ou une autre subtilité de programmation. Dans ce cas, on force l'arrêt en appuyant sur les touches CTRL + C puis on force Scilab à quitter le programme en cliquant entrant l'instruction `abort.`

2.1 Opérations.

On peut réaliser les opérations de base :

- Addition : `a+b`
- Soustraction : `a-b`
- Multiplication : `a*b`
- Division : `a/b`
- **Puissance** : `a^b` ou `a**2`

Virgule : la virgule se note par un point : On écrira 2.1 et pas 2,1 !

2.2 Constantes

Les nombres e, π et i (nombre complexe) sont prédéfinis. On les utilise en tapant respectivement : `%e`, `%pi`, `%i`.

Exercice 1 :

Dans chaque cas, écrivez l'instruction à taper dans la console pour effectuer le calcul demandé **puis** (jouez le jeu) vérifiez avec la console. Vous pouvez noter vos éventuelles remarques dans la partie droite.

Opération	Instruction	Résultat attendu	Remarque / pense-bête
$2-8$		`-6.`	
$2+8$		`10.`	

Opération	Instruction	Résultat attendu	Remarque / pense-bête
$3{,}1 \times 5$		`15.5`	
2^3		`8.`	
$\frac{8}{3}$		`2.666667`	
$\frac{8}{\frac{2}{3}}$		`12.`	
$\frac{\frac{8}{3}}{2}$		`1.3333333`	
$3e + 2$		`10.154845`	
$\frac{\pi}{3}$		`1.0471976`	
$\frac{1}{2i}$		`0. - 0.5i`	
3(6-7)		`-3.`	
$(-1)^3(5 + 7i)$		`-5. - 7.i`	
$99\ 999\ 999$		`99999999.`	
$100\ 000\ 002$		`1.000D+08`	
2^{1000}		`1.07D+301`	
2^{10000}		`Inf`	

Affichage des valeurs : Comme les deux derniers calculs vous le montrent, Scilab affiche les résultats sous forme décimale ou sous forme scientifique. Cela dépend du nombre.

`1.000D+08` se lit 1×10^8 on peut donc penser que Scilab a arrondi la valeur $100\ 000\ 002$ à $10^8 = 100\ 000\ 000$. En fait, Scilab n'affiche pas toutes les décimales, mais il a bien gardé le 2 dans la valeur. Pour afficher le nombre que Scilab a en mémoire, tapez l'instruction `format(25)` (qui demande à afficher les nombres avec un maximum de chiffres possible), puis retapez l'instruction `100000002`. Vous constaterez alors que Scilab n'a pas arrondi ce nombre.

Valeurs maximales : Scilab ne peut manipuler des nombres que jusqu'à une certaine valeur ? Au-delà, il considère que c'est l'infini !! Pour le comprendre, entrer les deux valeurs ci-dessous :
`10^300` et `10^400`.
Vous voyez que le premier nombre est encore dans les limites de calcul de Scilab alors que le deuxième est interprété comme étant l'infini.

Valeurs exactes ou approchées ? Taper les deux calculs ci-dessous dans la console :
`1-5*0.2` et `1-0.2-0.2-0.2-0.2-0.2`
Les résultats sont-ils surprenants ? Dans le deuxième calcul, les erreurs d'arrondi s'accumulent au fur à mesure des soustractions ce qui fait que le résultat final est faux. Un logiciel de calcul tel que Scilab utilise toujours des valeurs approchées pour calculer. C'est grâce à cela qu'il peut calculer vite et bien. On peut du coup observer parfois des petites erreurs d'arrondis. Rien de grave, mais il faut en être conscient ! Il faut donc bien comprendre que `4.441D-16` représente en fait zéro (en effet, $4{,}41 \times 10^{-16}$ est très proche de 0).

2.3 Les fonctions usuelles.

Exponentielle : `exp(a)`

Logarithme népérien : `log(a)`

Cosinus, sinus, tangente : `cos(a)`, `sin(a)`, `tan(a)` (les angles sont à exprimer en radian).

Racine carrée : `sqrt(a)`

Valeur absolue : `abs(a)`

Partie entière : `floor(a)`

Partie réelle et imaginaire d'un complexe : `real(a)`, `imag(a)`

Exemples

```
-->cos(%pi)
 ans  =
    - 1.
-->exp(3)
 ans  =
    20.085537
-->floor(2.6)
 ans  =
    2.
```

Exercice 2 :

Opération	Instruction	Résultat attendu	Remarque / pense-bête
$\sqrt{4}$		`2.`	
$\sqrt{2}$		`1.4142136`	Scilab prend une valeur approchée de $\sqrt{2}$.
$\left(\sqrt{2}\right)^2 - 2$		`4.441D-16`	Scilab prend une valeur approchée de 0.
e^{π}		`23.140693`	
$e^{i\pi}$		`-1. + 1.225D-16i`	Scilab prend une valeur approchée de 0.
$\ln 1 - \ln e$		`-1.`	
$\sin\frac{3\pi}{4}$		`0.7071068`	Scilab prend une valeur approchée de $\frac{\sqrt{2}}{2}$.
$\lfloor e - 3{,}5 \rfloor$		`-1.`	
$re\left(e^{\frac{2i\pi}{3}}\right)$		`-0.5000000`	
$\lvert 2^5 - 5^2 \rvert$		`7.`	
$\lvert 1 + 2i \rvert$		`2.236068`	

2.4 Autres fonctionnalités

2.4.1 Reprendre une instruction précédente.

En utilisant la flèche ↑ de votre clavier, vous pouvez remonter dans l'historique de vos instructions. Vous pouvez aussi utiliser la fenêtre « historique des commandes » et double-cliquer sur la commande que vous voulez relancer.

Exercice 3 : Réaliser les 3 calculs suivants, **sans tout retaper à chaque fois !!!.**

Opération	Résultat attendu	Remarque / pense-bête
$\frac{3 \cdot 4^2 - 10e + 71(1 - 2\pi)}{6 + 2^e}$	`-28.160892`	
$\frac{3 \cdot 4^2 + 10e + 71(1 - 2\pi)}{6 + 2^e}$	`-23.839604`	
$\frac{3 \cdot 3^2 - 10e + 71(1 - 2\pi)}{6 + 2^e}$	`-29.830091`	

2.4.2 Messages d'erreur.

Lorsque Scilab rencontre une erreur, il affiche un message d'erreur avec la cause probable de l'erreur. **Il faut lire ces messages** car ils vous permettent en général de corriger rapidement votre erreur.

Par exemple, ci-dessous, on a la classique « erreur de syntaxe » ou « syntax error ». Cela signifie que vous avez tapé quelque chose qui ne veut rien dire pour Scilab. C'est l'erreur la plus fréquente, car il suffit d'oublier un * ou une parenthèse… Dans le premier cas ci-dessous, par exemple, on a oublié le *.

```
--> 2%e
2%e
^~~^
Erreur : syntax error, unexpected identifier, expecting end of file

--> (2+3
(2+3
   ^^
Erreur : syntax error, unexpected end of file

--> 3*a

Variable non définie : a
```

2.4.3 Masquer les résultats.

Tapez les instructions ci-dessous dans la console et observez le résultat.

```
--> a=2+3;

--> a=2+3
 a   =

    5.
```

Lorsque Scilab rencontre une erreur, il affiche un message d'erreur avec la cause probable de l'erreur. **Il faut lire ces messages** car ils vous permettent en général de corriger rapidement votre erreur.

3 Variables

Création et utilisation de variables numériques

En écrivant a=2, on crée une variable nommée a qui prend la valeur 2.

```
--> a=2
 a   =

   2.
```

Exercice 4

1. Créez les variables $a = \sqrt{5 + 8e}$ et $A = \frac{7^3-5}{2}$.
2. Calculez $\frac{7^3-5}{2\sqrt{5+8e}}$ à l'aide des variables a et A et mettez le résultat dans la variable bob

   Calcul effectué : …… Résultat attendu : `32.677981`

   La variable bob vaut maintenant $\frac{7^3-5}{2\sqrt{5+8e}}$ (ou du moins une valeur approchée).
3. Créez maintenant la variable Bob qui vaut a*A .
4. Effectuez ensuite les calculs suivants :

Opération	Instruction	Résultat attendu	Remarque / pense-bête
$a^2 + 2A + \frac{1}{\text{bob}}$		`364.77686`	
$\ln(\text{bob})$		`3.4867015`	
$\frac{1}{aA}$		`0.0011441`	

Comme on le voit, le nom d'une variable est sensible aux majuscules/minuscules. On est parcontre assez libre sur le choix des noms des variables.

Le navigateur de variables.

Dans le navigateur de variable, vous devez voir apparaître vos trois variables a, A et bob. Vous voyez aussi peut-être apparaître une variable ans qui contient la dernière réponse affichée par la console. Vous pouvez noter que toutes ces variables sont du type « double » ce qui, pour simplifier, signifie que ce sont des nombres.

On peut supprimer une variable dans le navigateur (mais ça n'aura que peu d'intérêt pour nous en pratique).

Supprimez toutes les variables en tapant `clear` dans la console.

4 Les chaines de caractères.

Dans la console, tapez l'instruction suivante :

```
a="coucou"
```

Vous avez ainsi créé une variable qui ne contient pas un nombre, mais qui contient un *mot* ou plus précisément une *chaîne de caractères*. Vous pouvez d'ailleurs voir dans le navigateur de variable que sont type est « chaîne de caractères. »

Attention, pour créer une chaine de caractère, il faut mettre cette chaîne entre les guillemets.
Tapez maintenant :

```
a=coucou
```

Vous obtenez alors une erreur car si vous écrivez coucou sans guillemets, vous demandez en fait à Scilab que a prennent la valeur de la variable coucou. Comme il n'y a pas de variable qui s'appelle coucou, Scilab renvoie une erreur.

Concaténation des chaînes de caractères.
Créez variable b qui contient votre prénom.
Tapez ensuite a+b.
On s'aperçoit que l'addition deux chaînes de caractères correspond à leur concaténation.
Modifiez la variable a comme suit, puis taper de nouveau a+b. Vérifiez que le résultat est plus satisfaisant.

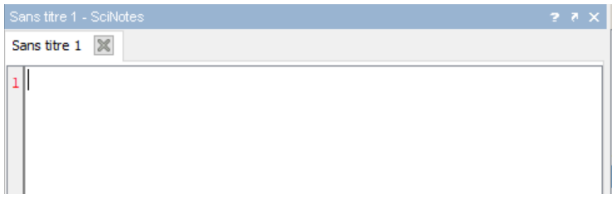
```
a="coucou "
```

Types de variables.
Nous avons donc vu deux types de variables : les nombres (que Scilab appelle des « double ») et les chaînes de caractères. On ne peut évidemment pas ajouter un nombre et une chaine de caractère :

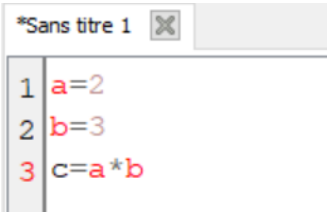
```
Variable non définie : bonjour

--> a=2
 a  =

   2.

--> b="Bonjour "
 b  =

  "Bonjour "

--> b+a

Opération indéfinie pour les opérandes données.
vérifier ou définir la fonction %c_a_s pour la surcharge.
```

5 L'éditeur SciNotes et la fonction `disp()`.

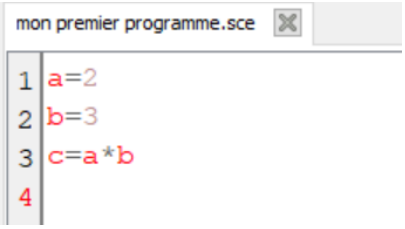
Ouvrez, si ce n'est pas déjà fait, la fenêtre Scinote. Cette fenêtre se présente sous la forme suivante :



Nous avons un onglet appelé « sans titre 1 » dans lequel nous pouvons entrer du texte.
Complétez le comme suit :



Enregistrez votre fichier dans le répertoire TP01 que vous avez créé en début de séance. Appellez votre fichier « mon premier programme».



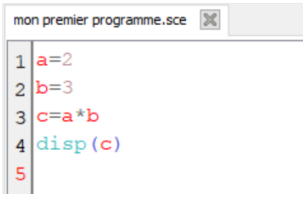
Une fois votre fichier enregistré, vous pouvez demander à Scilab de l'exécuter. Scilab va alors lire les lignes et les exécuter l'une après l'autre.

Pour cela, appuyez sur la touche F5 de votre clavier ou sur l'icone suivante :  .

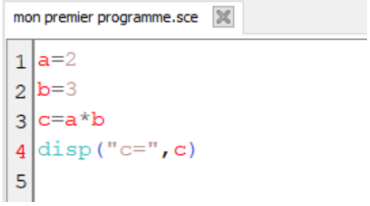
Observez alors ce qui s'est passé dans la console : rien ! Mais si vous observez le navigateur de variables, vous devez y trouver les trois variables a, b et c avec les valeurs, 2, 3 et 6.

Lorsqu'on execute un programme écrit dans Scinotes, Scilab n'affiche pas les résultats des calculs qu'on effectue. Pour lui demander d'afficher la valeur d'une variable, on utilise l'instruction `disp`.

Pour afficher la valeur de la variable c, ajoutez l'instruction suivante :
Puis exectutez votre programme.



Vous devez vous apercevoir que Scilab affiche la valeur 6 dans la console, sans expliquer que c'est la valeur de c. On peut alors afficher un petit message avant d'afficher la valeur de c :



NB : dans les anciennes versions de Scilab, il y avait un petit « bug » sur la fonction disp qui affichait les valeurs à l'envers.
On écrivait alors `disp(c,"c = ")` **au lieu de** `disp("c = ",c)`. **Vous trouverez donc cette façon d'écrire dans les annales des sujets de concours.**

6 La fonction `input()`

Afin de faire des programmes intéressants, il faut pouvoir demander à l'utilisateur de choisir la valeur qu'il veut donner à une variable donnée. Cela se fait à l'aide de la fonction `input()`.

Pour comprendre cette fonction, modifier votre programme de la façon suivante, puis exécutez-le.

mon premier programme.sce

```
1 a=input("Entrez la valeur de a ")
2 b=input("Entrez la valeur de b ")
3 c=a*b
4 disp("c=",c)
```

Mais l'utilisateur n'a pas besoin de savoir comment on a appelé les variables. On peut donc adapter le message pour rendre l'utilisation des variables « transparentes » pour l'utilisateur. Modifiez à nouveau votre programme comme suit, puis exécutez-le.

mon premier programme.sce

```
1 a=input("Entrez un nombre : ")
2 b=input("Entrez un autre nombre : ")
3 c=a*b
4 disp("Le produit de vos nombres vaut : ",c)
5
```

7 Exercices

Exercice 1 :

Ecrivez un programme qui demande trois nombres a, b et c à l'utilisateur, et affiche le discriminant du trinôme $ax^2 + bx + c$.

Exercice 2 (plus difficile) :

Ecrivez un programme qui demande un nombre de secondes à l'utilisateur et le converti en jours, heures, minutes, secondes. Vous aurez besoin pour ce programme de la fonction « partie entière », qui s'écrit `floor()`.