TP Maple 1 | Le système Maple et son bon usage

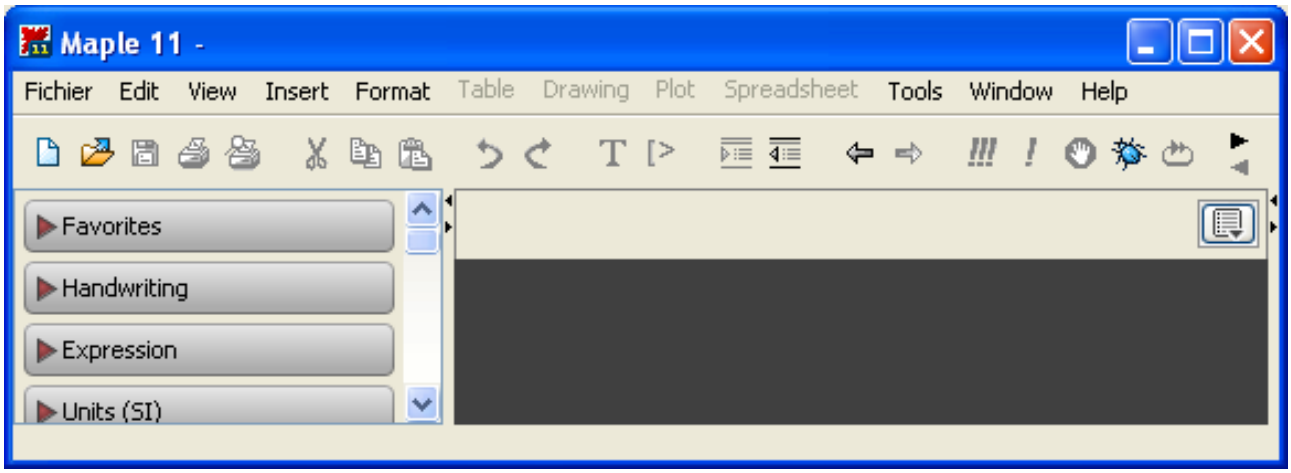
Maple est né en 1981 au sein du Symbolic Computation Group de l'université de Waterloo dans l'Ontario (Canada). Le logiciel fut commercialisé dès 1985 par la firme Maplesoft (voir le site www.maplesoft.com pour de plus amples informations). La version la plus récente du logiciel est Maple 12 et date de 2009.

1. Découverte du système Maple

On commencera par remarquer (et même déplorer !) que l'interface Maple-utilsateur est entièrement écrite en Anglais. Ce qui ne posera aucun problème pour de nombreuses routines (sauvegardes, accès aux barres de symboles, zoom, etc.) s'avérera plus ennuyeux lors de la consultation de la rubrique d'aide.

1.1. Les menus déroulants et la barre d'outils

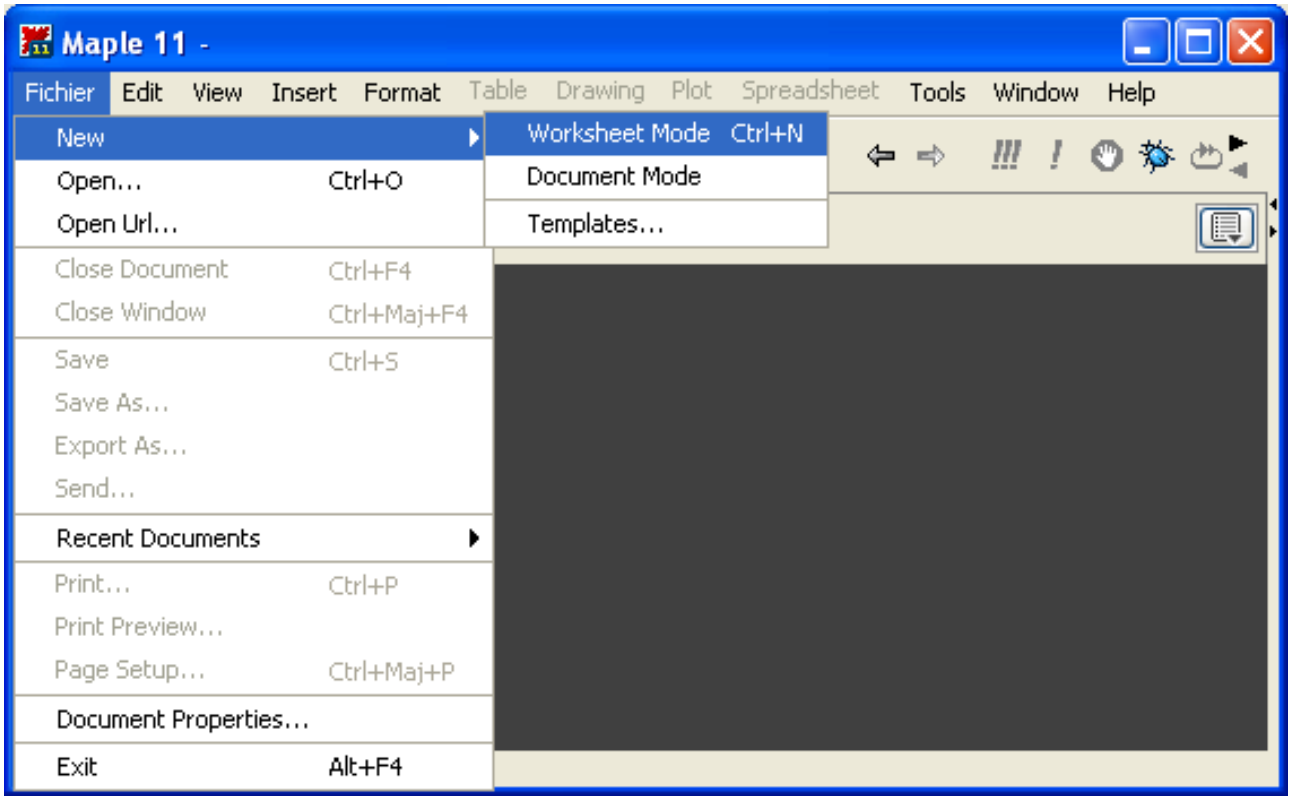
A l'ouverture de Maple 11, l'utilisateur verra apparaître la fenêtre suivante à l'écran :



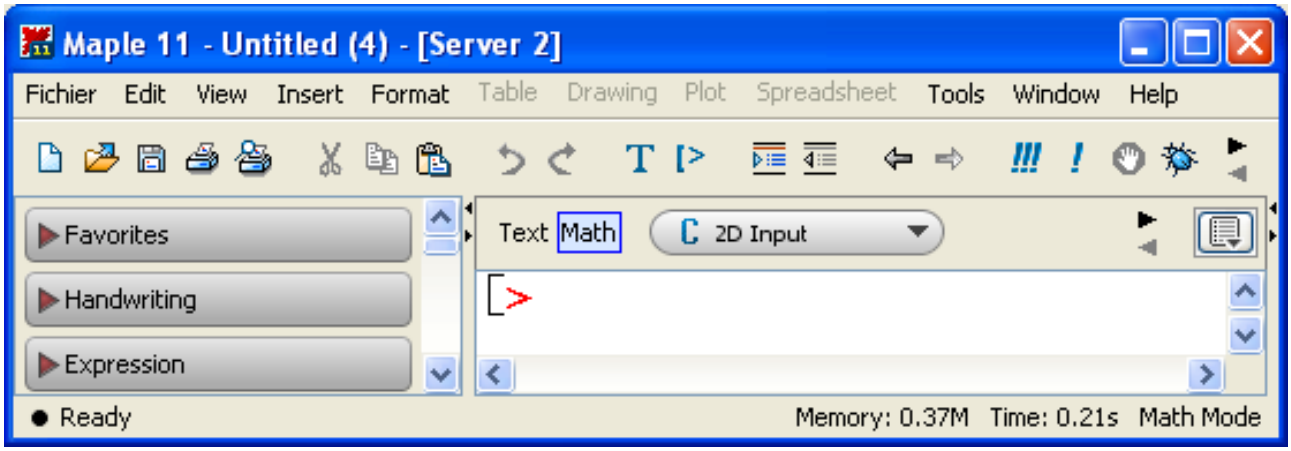
Le lecteur trouvera de quoi ouvrir, sauvegarder, etc. un fichier dans le menu déroulant **FILE**. Le menu **VIEW** comporte une option de zoom parfois utile. On trouvera dans le menu déroulant **HELP**, et en particulier dans la rubrique **TOPIC SEARCH**, de précieux renseignements (in english of course) sur toutes les commandes : fonction, syntaxe et variantes, le tout agrémenté de nombreux exemples.

1.2. Ouverture d'une session

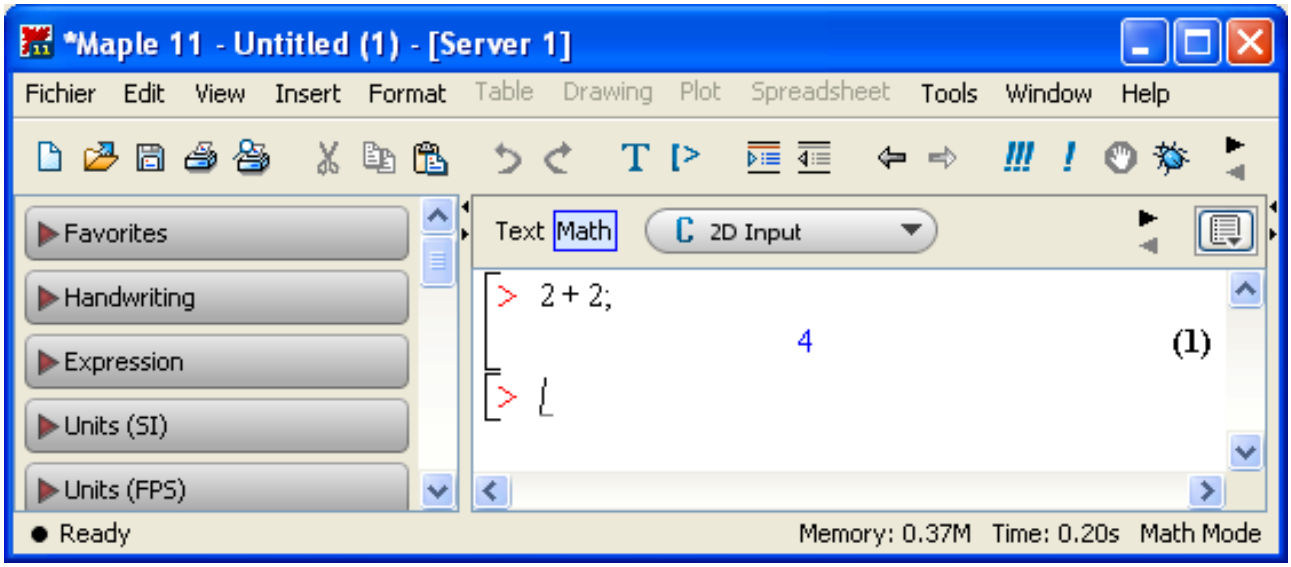
On débute une *session* de Maple en ouvrant une nouvelle *feuille de calcul* (a Maple worksheet) dans le menu déroulant **FILE** au moyen de l'onglet **NEW** en selectionnant WORKSHETT MODE.



La session peut alors commencer, le *prompt* > apparaît et le curseur | clignote en attendant les commandes.



Ces dernières sont exécutées après validation par la touche **ENTRÉE**.



L'interface de Maple comporte de nombreux aménagements permettant à l'utilisateur d'arriver plus vite aux résultats souhaités ; on retiendra en premier lieu dans le menu déroulant **VIEW**, l'option **PALETTES** qui comporte trois barres de raccourcis de commandes usuelles : **SYMBOL PALETTE**, **EXPRESSION PALETTE** et **MATRIX PALETTE**. On retiendra également la présence d'onglets disposés verticalement à droite de la feuille de calcul : voir dans la figure ci-dessus les onglets ▶ HANDWRITING, ▶ EXPRESSION, etc.

Les feuilles de calcul sont sauvegardées sous l'extension **. mws** (mws pour *maple worksheet*) à l'aide des options **SAVE** ou **SAVE AS...** du menu déroulant **FILE**.

1.3. Les modes « calcul » et « texte »

Le mode calcul est celui où l'utilisateur dialogue avec le logiciel par l'intermédiaire de lignes de commandes. Ce mode est facilement repérable par la présence du *prompt* >.

Il souvent utile, pour des raisons de lisibilité, d'insérer des commentaires dans une feuille de calcul Maple. On a alors le choix entre deux options.

► le passage en *mode texte* (le prompt > disparaît alors) en cliquant sur l'îcone **T** de la barre d'outil qui se situe sous les menus défilants en haut de l'écran. On peut aussi aller dans le menu défilant **INSERT** et choisir l'option **TEXT INPUT**. L'utilisateur pourra mettre en page ses commentaires en utilisant les diverses options de centrage du menu **OPTIONS** ainsi que les barres d'outil qui apparaissent en mode texte.

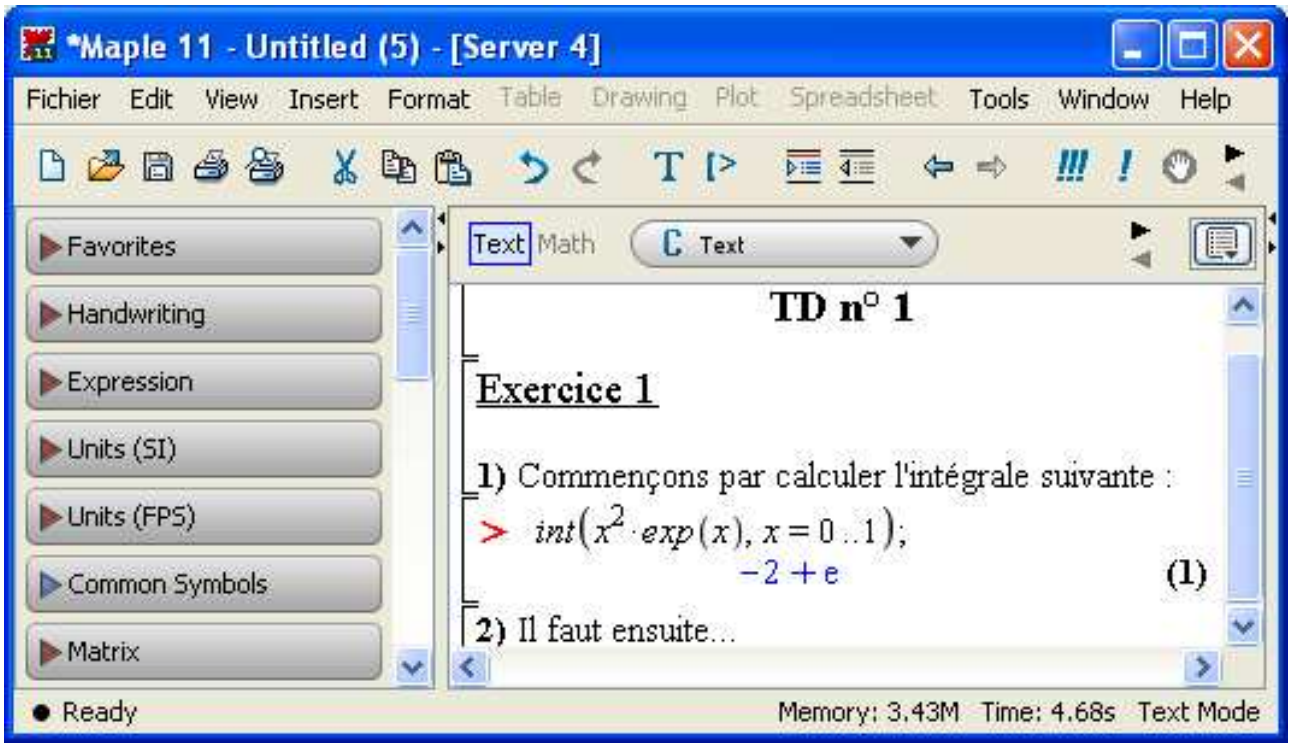
► l'utilisation du symbole **#**. Tout ce qui suit # sur une ligne donnée est ignoré par Maple.

```
> (2+I)^5  # Snif, snif... Maple va m'ignorer !
```

$$38+41I$$

Le retour au *mode calcul* s'effectue en cliquant sur l'îcone > de la barre d'outil qui se situe sous les menus défilants en haut de l'écran. On peut aussi aller dans le menu défilant **INSERT** et choisir l'option **MAPLE INPUT**. Le prompt > réapparaît à l'écran.

Le mélange de ces deux modes permet la création de documents scientifiques clairs (pour des rapports, des transparents, des cours, des devoirs libres, des travaux dirigés, etc.).

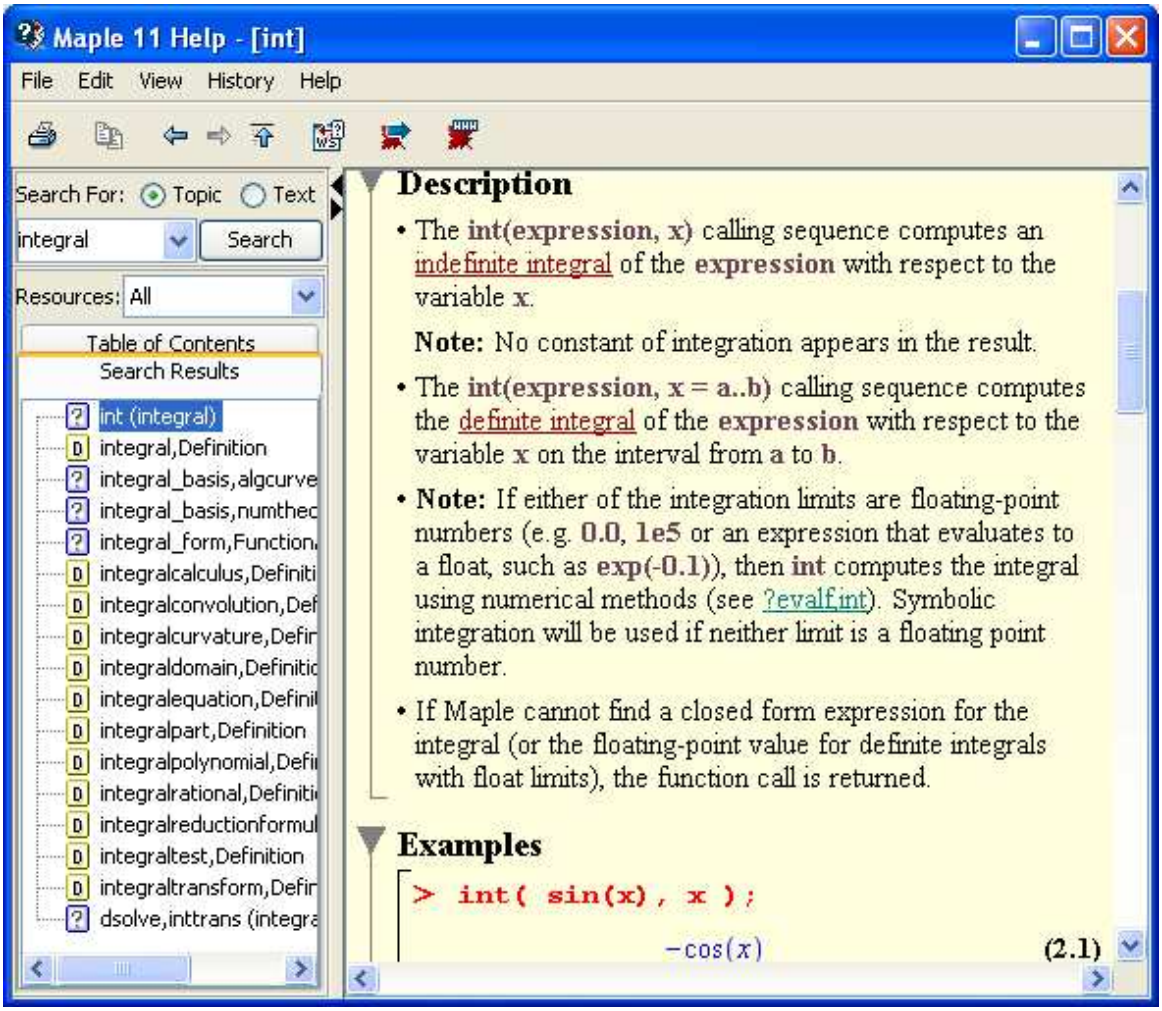


1.4. L'aide de Maple

Le lecteur ne doit aucunement connaître par cœur toutes les fonctionnalités de Maple. A l'oral des concours, il aura le droit de consulter l'aide du logiciel. Il devra se familiariser avec la recherche d'outils en Anglais. Dans le menu déroulant **HELP**, il trouvera la rubrique **MAPLE HELP** qui lui permettra de rechercher les entrées de l'aide correspondant à sa demande. Voici un petit exemple : recherchons des informations sur le calcul intégral en entrant *integral* (in english if you please). Le résultat de la recherche s'affiche dans la fenêtre de gauche. On peut alors accéder aux différentes entrées en cliquant sur les noms correspondants. Cliquons par exemple sur *int*. Apparaît alors dans la fenêtre de droite l'article correspondant. On peut y lire un descriptif de la commande **int** permettant de calculer des intégrales et des primitives de fonctions usuelles. On y trouvera tous les détails : les syntaxes, les variables d'entrée, le résultat fourni, la description [1] de l'algorithme de calcul utilisé (dans le cas d'une méthode numérique), les options, les variantes, le tout étant illustré par de nombreux exemples élémentaires. Recherchons dans l'article comment calculer sous Maple les primitives suivantes :

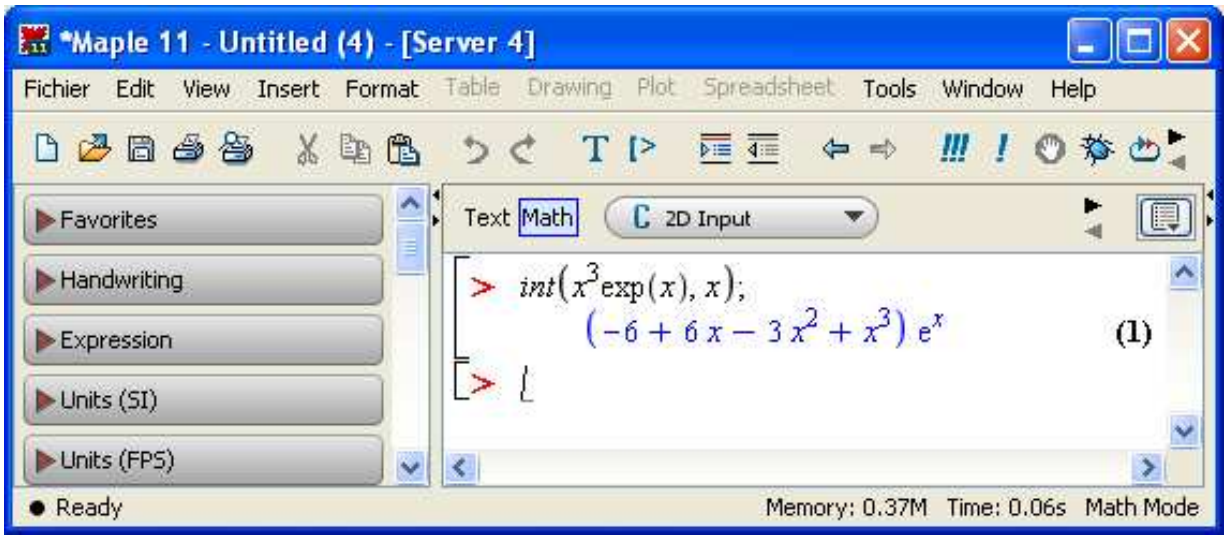
$$\int x^3 e^x dx.$$

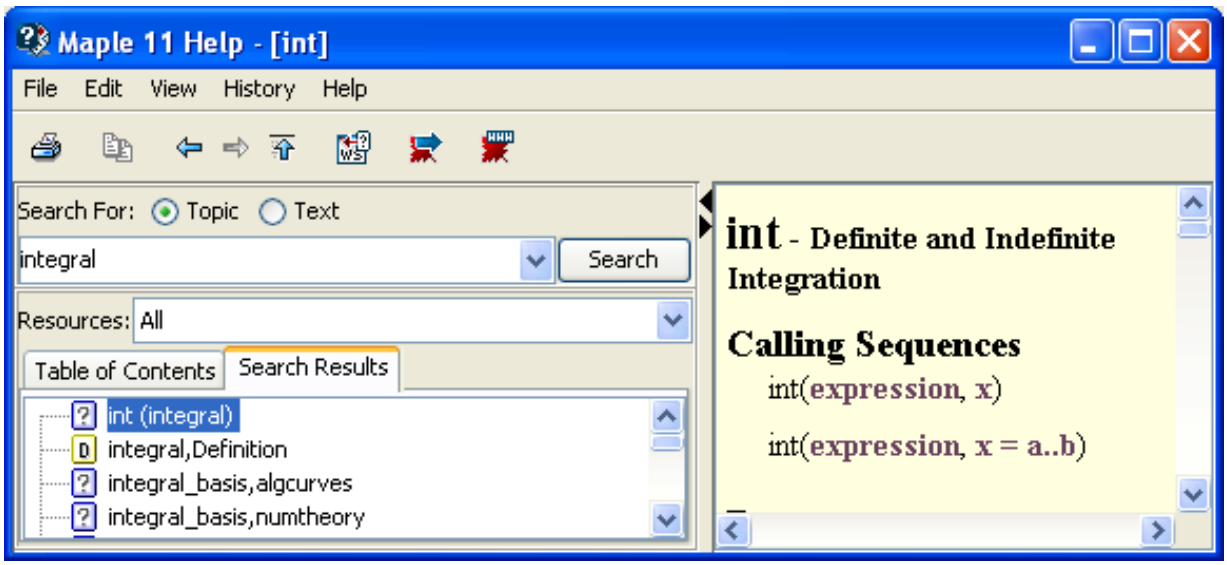
On déroule pour cela l'article verticalement. On trouve par exemple :



1. parfois très succinte !...

La description de la commande **int** est claire (but in English...) et étayée par l'exemple du sinus. Allons-y : retournons à notre feuille de calcul.





Exercice 1.

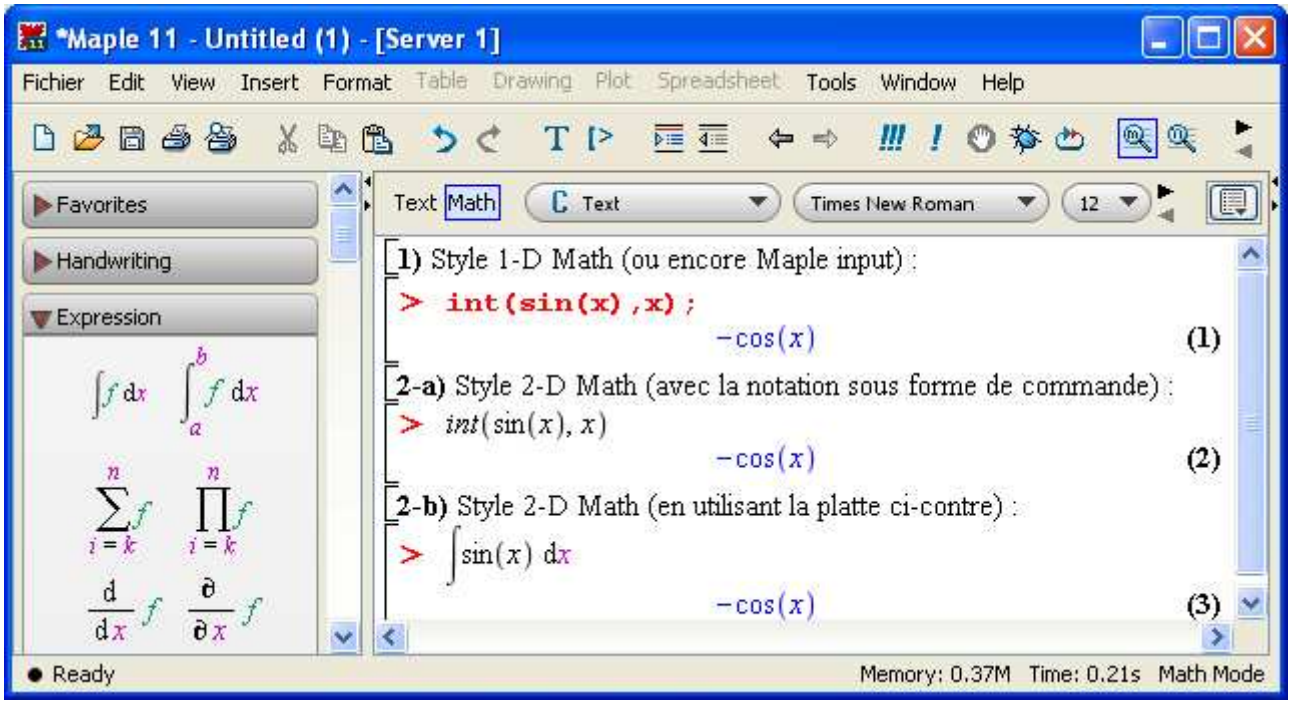
En utilisant l'aide en ligne, trouver comment calculer le module d'un nombre complexe.

Exercice 2.

En utilisant l'aide en ligne, trouver comment dériver vingt fois $\ell n\left(1+x^2\right)$ par rapport à x.

1.5. Les styles 1-D Math et 2-D Math

Depuis la version 10 de Maple, deux styles différents de notations sont disponibles pour les *entrées*. L'ancien style, appelé le 1-D MATH, est propre au logiciel contrairement au style 2-D MATH qui donne *en plus* accès à la notation standard mathématique. En guise d'exemple, voici le calcul des primitives du sinus programmé dans les deux styles.



Bien qu'utilisables sous ces deux styles, les instructions standards de Maple (telles que **int** ci-dessus) sont soumises à des règles syntaxiques plus fortes en 1-D MATH qu'en 2-D MATH : on pourra par exemple omettre le point-virgule (*semi-colon* in english) en 2-D MATH alors que cela conduirait à une erreur en 1-D MATH. Afin d'entraîner le lecteur à la rigueur syntaxique (indispensable lorsque l'on aborde des langages plus puissants comme **C**, **Fortran**, etc.) et pour des raisons de compatibilité avec les versions antérieures de Maple, nous choisirons de programmer dans le style 1-D MATH dans la suite de ce cours. L'utilsateur pourra fixer cette configuration en changeant les paramètres du logiciel dans la suite d'onglets TOOLS - OPTIONS... - DISPLAY : *Maple Notation* pour input display et *2-D math Notation* pour output dispaly.

2. Le bon usage

Voici quelques règles élémentaires d'utilisation de Maple.

2.1. Lignes d'instructions

Une ligne de commandes est une succession de caractères écrite en mode mathématique, donc commençant par le symbole « > » .

```
> 2+3: 5*9: # Ceci est une ligne de commandes !
```

Une session de Maple consiste en une succession de lignes de commandes écrites dans une feuille de calcul en mode mathématique, éventuellement entrecoupées de commentaires écrits en mode texte.

Il est recommandé de passer à la ligne manuellement lorsque la largeur de la ligne de commandes dépasse celle de la fenêtre du logiciel. Cela rend les lignes de commandes plus lisibles. On passera à la ligne (sans changer de ligne de commandes !) en activant simultanément les touches *Shift* et *Return* du clavier.

```
> 5+5; 6+4; 7+8;
  3+6; 4+8; 7-5;
  # Ceci est une seule ligne de commande !
```

Une ligne de commande sera éxécutée après activation de la touche *Return* du clavier[2]. Toute ligne de commande devra se finir par les symboles « **;** » ou « **:** » . Le point virgule « **;** » est utilisé en fin de ligne lorsque l'on souhaite l'affichage du résultat à l'écran. Dans le cas contraire, on emploiera le double point « **:** » . On retiendra la règle fondamentale suivante :

Terminaison d'une ligne de commande

Une ligne de commande doit *toujours* s'achever par « **;** » (*les commandes sont effectuées et leur résultat est affichée*) ou « **:** » (*les commandes sont effectuées mais leur résultat ne s'affiche pas*) sous peine d'un message d'erreur.

Le lecteur méditera les exemples suivants...

```
> 1/3-1/4: 1/3-1/4;
```

$$\frac{1}{12}$$

S'il souhaite afficher des résultats consécutifs sur une même ligne, l'utilisateur utilisera une seule ligne de commandes, séparant les calculs les uns des autres par une virgule.

```
> 2+3, 3-9, 2-8;
```

$$5, -6, -6$$

2. Il n'est pas indispensable que le curseur « I » soit situé en fin de ligne pour que l'éxécution ait lieu. Cela est particulièrement utile en cas de correction d'une ligne.

Dans le cas où la ligne de commandes n'est pas correcte, par exemple en cas d'erreur de parenthésage, l'analyseur syntaxique de Maple adresse un message d'erreur à l'utilisateur. Le type d'erreur y est indiqué.

```
> 1/3-1/4
Warning, inserted missing semicolon at end of statement
```

Comme l'indique le logiciel, la ligne de commande ci-dessus n'est pas correcte car sa terminaison (par **;** ou **:**) est manquante.

Il existe de nombreux cas où une erreur ne sera pas détéctée par le logiciel, car il s'agira d'une erreur pour l'utilisateur mais pas pour le logiciel. C'est en particulier le cas des erreurs d'orthographe :

```
> cosinus(Pi/2);
```

$$\text{cosinus}(\pi/2)$$

Dans cet exemple, le logiciel reconnaît la chaîne de caractères « *cosinus(Pi/2)* » , il n'y a donc pas d'erreur de syntaxe. Cependant, le résultat escompté par l'utilisateur était bien-sûr la valeur de $\cos(\pi/2)$, c'est-à-dire 0. Voici une ligne de commandes correcte :

```
> cos(Pi/2);
```

$$0$$

2.2. Les opérations élémentaires

Les opérations usuelles (liées à l'addition et la multiplication) sont notées usuellement. On retiendra cependant que la notation multiplicative abrégée (comme dans $7x$ au lieu de $7 * x$) n'est pas syntaxiquement correcte.

Syntaxe des opérations élémentaires

- L'addition est notée par « + » ;
- La soustraction est notée par « − » ;
- La multiplication est notée **obligatoirement** par « ⋆ » ;
- La division est notée par « / » .

Les priorités relatives des opérations sont également usuelles, de même que l'utilisation des parenthèses au sein d'un calcul.

```
> 2+6*8;

                                  50

> (2+6)*8;

                                  64
```

On notera que les *entrées* sont affichées à l'écran en rouge et les sorties (c'est-à-dire les résultats) sont affichés en bleu et centrés sur la ligne [3].

2.3. Majuscules et minuscules

On retiendra la règle fondamentale suivante :

Majuscules et minuscules

Maple distingue les lettres majuscules des lettres minuscules.

2.4. Feuille de calcul Vs. noyau d'une session

Après lancement de l'application Maple, l'utilisateur ouvrira une session en cliquant sur la suite d'onglets NEW - WORKSHEET MODE du menu déroulant FICHIER. Sera alors automatiquement créée dans l'ordinateur une zone de mémoire, extensible au fur et à mesure des besoins, pour stocker les données propres au créateur de la session (noms des variables, expressions, etc.) pendant sa période d'utilisation du logiciel : c'est ce qu'on appelle le noyau (*the kernel*). Il est important de distinguer le contenu du noyau de la feuille de calcul : le noyau est toujours cohérent alors que la feuille de calcul peut ne pas l'être... Examinons le cas où l'utilisateur revient sur ses pas et modifie le début de sa feuille de calcul sans en modifier la fin : dans cette situation, la feuille de calcul, lue dans son intégralité, sera *mathématiquement* incohérente [4]. Considérons la séquence suivante :

```
> a:=4:  # ligne validée par un return
> a^2;   # ligne validée par un return

                                  16
```

3. Ce sont les couleurs par défaut, l'utilisateur peut bien-sûr effectuer d'autres réglages.

4. Cette particularité est dûe à la structure de *pile* du noyau : on peut considérer que les différentes actions de l'utilisateur sont « *empilées* » les unes sur les autres dans le noyau. La trace de toutes ces actions subsistant dans le noyau, toute lecture de ce dernier ne présenterait aucune incohérence mathématique. C'est dans ce sens qu'il faut entendre la différence entre le noyau et la feuille de calcul (qui ne conserve aucune trace de son histoire).

Il faut la comprendre de la manière suivante [5] : la case-mémoire de l'ordinateur désignée par la lettre a contient la valeur numérique 2. Lorsuqe l'on demande le calcul de a^2, le logiciel retourne la valeur $2^2 = 4$. Si l'on modifie la première ligne de cette séquence par $a := 3$ on obtient après validation par la touche *Return* :

```
> a:=3:   # ligne re-validée par un return
> a^2;    # ligne non re-validée

                                  16
```

Si l'on enrigistrait la feuille de calcul à cet instant, il pourrait sembler que le logiciel ne sait pas calculer... Pour obtenir le résultat escompté, il faut valider à nouveau la ligne concernant a^2 au moyen de la touche *Return* du clavier :

```
> a:=3:
> a^2;  # ligne re-validée par un return

                                  9
```

Ouf ! On en déduit la règle fondamentale suivante :

Modification d'une ligne de commandes

Toute modification d'une ligne de commandes doit s'accompagner d'une validation (par la touche *Return*) des lignes suivantes sous peine d'ambiguité.

Lorsque l'on ouvre une ancienne session, il ne reste que la trace de la feuille de calcul : le noyau initial (conçu à la création de la feuille) a été effacé. Si l'on veut le reconstituer, il faudra resaisir chacune des lignes de commandes de la feuille en question. C'est ce qu'on le fera – en cas de doute – lors de l'ouverture d'une session antérieure.

2.5. Utilisation du symbole % lors d'un calcul

Le symbole pour-cent %, employé comme argument, désigne le dernier résultat (*chronologiquement !*) calculé par Maple [6]. De même %% désigne l'avant-dernier résultat calculé et %%% l'antépénultième. Ce procédé de rappel s'arrête là. Cette technique évite l'emploi du copier-coller, ce qui est parfois apréciable :

5. Le lecteur nous permettra une petite anticipation (pédagogique !) sur la notion de variable...
6. Cela n'est vrai qu'à partir de Maple V, release 5. Avant cette version, c'est le caractère « " » qui jouait ce rôle.

```
> 8;
                                    8
> %*6;
                                   48
> (2+%%)*%;
                                  480
> (20+%)*%%+%%%;
                                 24008
```

3. Les variables

Une variable est une chaîne de caractères associée à un emplacement de la mémoire utilisée par Maple[7]. Cette « *case de mémoire* » peut contenir une donnée choisie par l'utilisateur. Toutes les chaînes de caractéres ne sont pas permises, certaines sont réservées (comme **Pi** qui contient une valeur approchée du nombre réel π ou encore **I** qui désigne le nombre complexe i). Mis à part quelques mots réservés[8] On retiendra la règle suivante :

Noms de variables sous Maple

Les noms de variable doivent commencer par une lettre mais peuvent ensuite contenir des chiffres ainsi que le caractère « _ » . Le logiciel distingue les minuscules des majuscules.

3.1. Assignations et évaluations

Par exemple, on peut *ranger* la valeur 8 dans la case de mémoire se trouvant à l'*adresse a* ; on utilise pour cela l'opération d'*affectation* suivante : **a :=8**. La variable a est dite *affectée* ou encore *que l'on a affecté la valeur* 8 *à la variable a*. Dans la suite des calculs, Maple remplacera systématiquement la variable a par son contenu 8. On peut alors *représenter* la variable a comme un « objet » *pointant* vers la valeur 8 :

$$a \rightarrow 8$$

On peut accéder (on dit parfois *évaluer*) au *contenu*[9] d'une variable par simple appel de son nom.

7. Comme une adresse est associée à un bâtiment . . .
8. Par exemple : **Pi**, **I**, **O**, **D** et **gamma**.
9. On dit aussi *accéder en lecture* à la variable. On emploie aussi le terme « *assignation* » .

```
> a:=8;
```

$$a := 8$$

```
> a;
```

$$8$$

On peut également *changer le contenu d'une variable* par une nouvelle affectation [10] (qui « écrase » l'ancien contenu !).

```
> a:=12:a;
```

$$12$$

Des affectations multiples sont possibles.

```
> b,c,d:=1,2,5:d,c,b;
```

$$5, 2, 1$$

L'utilisateur peut *empêcher l'accès en enregistrement* à une variable au moyen de la commande [11] **protect('*nom de la variable*')**.

```
> a:=5:protect('a'):a:=11;
Error, attempting to assign to `a` which is protected
> a;
```

$$5$$

L'opération inverse se fait avec la commande **unprotect('*nom de la variable*')**.

```
> unprotect('a'):a:=11;
```

$$11$$

10. On dit aussi *accéder en enregistrement* à la variable.
11. Surtout ne pas oublier les quotes « ' ' » sous peine d'erreur !

3.2. Utilisation des variables

On peut conserver un résultat utile (par exemple pour la suite d'un très long calcul) au moyen d'une variable. Examinons un exemple tiré du cours sur les nombres complexes (Attention, **I** désigne le nombre complexe i sous Maple !)

```
> a:=(I+8)*(-I+5);
```

$$a := 41 - 3I$$

```
> a^2;
```

$$1672 - 246I$$

```
> a*(7*I-5);
```

$$-184 + 302I$$

3.3. Variables libres et variables affectées

Une variable est dite libre lorsqu'elle n'a subie aucune affectation au cours de la séquence de calcul. On parle de variable *affectée* dans le cas contraire. On peut *libérer* une variable *affectée* en utilisant la commande **unassign**.

```
> unassign('a'):a;
```

$$a$$

On peut également employer la syntaxe suivante :

```
> a:='a':a;
```

$$a$$

Une manière brutale de libérer est une variable est d'utiliser la commande **restart** qui a en fait pour effet de libérer *toutes les variables* affectées par l'utilisateur au cours de la session de calcul.

Utilisation de **restart**

On commencera chaque nouvel exercice ou thème de calcul par une commande **restart**.

Exercice 3.

Effectuer, afficher et commenter la séquence suivante :

```
> b:=cos(a): a:=Pi/2: b;
```

$$0$$

```
> a:=2*Pi/3: b;
```

$$-\frac{1}{2}$$

```
> c:=cos(a): a:=3*Pi/4: b, c;
```

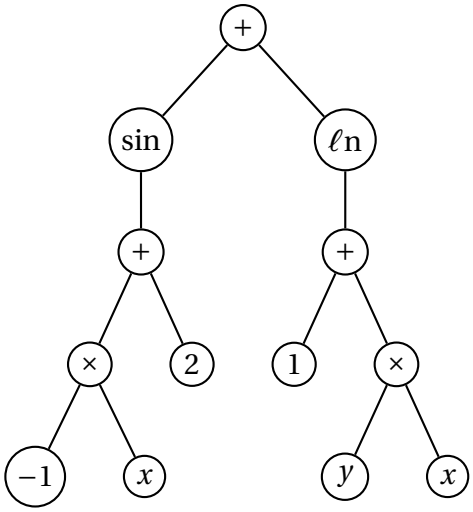
$$-\frac{1}{2}\sqrt{2},\ -\frac{1}{2}$$

4. Les expressions sous Maple

Maple manipule essentiellement des « *expressions* ». Tout est expression (ou presque sous Maple) : les nombres, les ensembles, les listes, le résultat de certaines commandes, etc. Une expression n'est ni plus ni moins qu'un arbre au sens des Mathématiques.

4.1. La structure d'arbre d'une expression

Examinons par exemple comment le logiciel « *compose* » l'expression $\sin(-x+2)+\ell\mathrm{n}(1+yx)$. La structure de l'expression est la suivante :

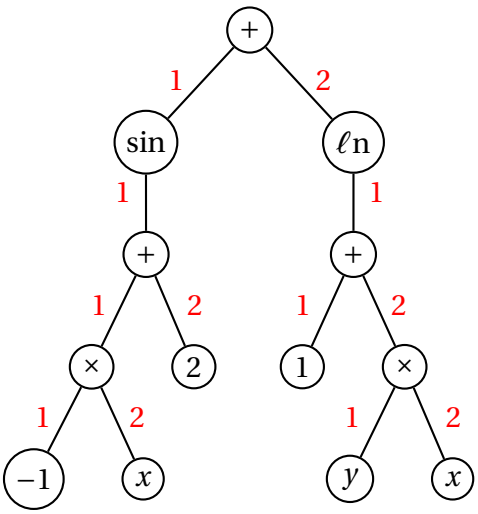


Une telle structure est appelée un arbre. Les cercles sont appelés les *noeuds* et les segments les reliant sont appelés les *branches* de l'arbre. Le premier noeud (celui qui contient +) est appelé la *racine* de l'arbre. Il faut la comprendre de la manière suivante : l'expression est le résultat d'une somme de termes, le premier étant le sinus d'une somme (de deux termes dont le premier est le produit de deux termes (l'entier -1 et l'expression x) et le second un nombre entier) et le second étant une somme de deux termes (dont le premier terme est un entier et le second le produit de deux termes (le premier étant l'entier 2 et le second l'expression x)).

Les résultats des commandes Maple étant des expressions, il est important de bien comprendre la structure de ces dernières afin de savoir exploiter efficacement les réponses du logiciel. Cela nous permettra de savoir où se trouve l'information qui nous intéresse et comment l'extraire en évitant le copier-coller.

4.2. La commande op

Le nombre d'opérandes d'une expression E est par définition le nombre de sous-arbre(s) issu(s) de la racine de l'arbre E. Par exemple, le nombre d'opérandes de l'expression illustrant le paragraphe précédant est 2. On accède à cette valeur au moyen de la commande **nops** et à la séquence des opérandes par la commande **op**. On peut également lire les opérandes des sous-arbres (et ainsi de suite) en précisant un *chemin* (dans l'arbre de E) en option dans la commande **op**. Pour cela, on utilise la notation suivante des branches de l'arbre :



Par exemple, l'expression y apparaissant dans le logarithme contenu dans E se trouve en choisissant le chemin suivant à partir de la racine :

branche 2 → branche 1 → branche 2 → branche 1

On notera ce chemin sous la forme d'une liste : [2,1,2,1]. On l'isolera en utilisant **op([2,1,2,1],E)**. Plutôt qu'un long discours, reprenons l'exemple précédent...

```
> E:=sin(-x+2)+ln(1+2*x):nops(E),op(E),op(2,E),
  op([2,1],E),op([2,1,2],E),op([2,1,2,2],E);
```

$$2, -\sin(x-2), \ell\mathrm{n}(1+2x), \ell\mathrm{n}(1+2x), 1+2x, 2x, x$$

La commande **op**

► **op(E)** : renvoie la liste des opérandes de E (ie les expressions correspondant aux branches issues de la racine).

► **op(m,E)** : renvoie la m-ième opérande de E.

► **nops(E)** : renvoie le nombre d'opérandes de E.

► **op([m,n,...,p],E)** : renvoie l'expression correspondant au chemin $[m,n,...,p]$ en partant de la racine dans l'arbre de E.

4.3. Type d'une expression

Maple attribue un type à toute expression.

```
> whattype(x^2+x), whattype(2*x), whattype(2);
```

$$`+`, `*`, \text{integer}$$

Parmi les types possibles, on trouvera : + (somme) , ∗ (produit) , (puissance) , = (égalité) , <> (non égalité) , < , <= (inégalité au sens strict ou large) , and , or , not (et, ou, non) , integer (nombre entier) , fraction (nombre rationnel) , float (nombre réel en virgule flottante) , complex (nombre complexe) , numeric (numérique) , symbol (symbole), string (chaîne de caractères) , list (liste) , set (ensemble) , table (table) , array(tableau) , function (fonction) , name (nom) , .. (intervalle).

Le type d'une expression E s'obtient également par la ligne de commande **op(0,E)** : il s'agit par définition de l'opérande de niveau 0 de E. Ainsi, pour l'expression E du paragraphe précédent :

```
> whattype(E), op(0,E);
```

$$+, +$$

Les variables non-affectées sont du type *symbol*.

Type d'une variable

L'appel **whattype(a)** retourne *symbol* si la variable a est libre. Sinon, elle retourne le type de l'expression contenue dans la variable a.

Exercice 4.

Effectuer, afficher et commenter la séquence suivante :

```
> b:=cos(a): a:=Pi/2: b;
```

$$0$$

```
> a:=2*Pi/3: b;
```

$$-\frac{1}{2}$$

```
> c:=cos(a): a:=3*Pi/4: b, c;
```

$$-\frac{1}{2}\sqrt{2},\ -\frac{1}{2}$$

4.4. Substitution : la commande subs

La commande subs

La ligne de commande **subs (x=a, E)** a pour résultat l'évaluation de l'expresion E pour la valeur a de l'expression x.

```
> F:=x^3+x^2+x+1: subs(x=0,F);
```

$$1$$

Dans l'expression F de l'exemple précédent, on peut aussi remplacer la « puissance 2 » par une « puissance 1 » :

```
> F:=x^3+x^2+x+1: subs(2=1,F);
```

$$x^3+2x+1$$

Il faut donc bien comprendre que la ligne de commande **subs(x=a,E)** a pour effet de remplacer l'expression x par l'expression a à chaque occurrence dans E.

4.5. Evaluation : la commande eval

Soit E une expression formée à partir d'autres expressions parmi lesquelles figurent le symbole [12] a. Evaluer E pour la valeur A (qui est aussi une expression) de a signifie que l'on souhaite substituer A à a dans E [13]. L'évaluation n'est donc qu'un cas particulier de substitution. En un mot, la commande **subs** permet plus que l'évalutation d'une expression.

On dispose de trois syntaxes pour l'évaluation d'une expression sous Maple.

```
> restart: F:=x^2=1: x:=I: F,x;
```

$$0, I$$

```
> restart: F:=x^2+1: subs(x=I,F),x;
```

$$0, x$$

```
> restart: F:=x^2+1: eval(F,x=-1),x;
```

$$2, x$$

L'utilisation des commandes **subs** et **eval** est clairement plus intéressante car elle n'affecte pas la variable x. Pour des raisons subtiles [14], il est recommandé d'utiliser de préférence **eval** pour les évaluations. Dans le cas où le résultat de l'évaluation est numérique (ie un nombre complexe), on pourra utiliser **evalf** pour demander une évaluation approchée plutôt que le résultat exact renvoyé par défaut par Maple.

12. C'est mal dit (à dessein) : plus précisément, a est une expression de type *symbol*. Rappelons que les expressions de type *symbol* sont des noms de variables utilisables.

13. Cela pourra occasionner un message d'erreur car toutes les substitutions ne sont pas valables !

14. Essentiellement parce que lors d'une commande **subs(x=a,E)** où E est une procédure, le logiciel commence par évaluer tous les paramètres d'entrée de l'expression E pour $x = a$ (ce qui s'avérera parfois parfois fâcheux) mais ce n'est pas le cas de la commande **eval**.

La commande d'évaluation **eval**

- **eval (E,x=a)** : évalue sous forme symbolique l'expression E pour $x = a$.
- **evalf (E)** : donne des valeurs approchées de l'expression numérique E .

```
> evalf(Pi);
```

3.141592654

Nous reviendrons sur ces évaluations approchées dans la séance consacrée aux nombres.

5. Manipulation des expressions usuelles

Voici en vrac quelques commandes permettant le calcul des expressions sous **MAPLE**.

```
> cos(2*x):expand(%);
```

$$2\cos(x)^2 - 1$$

```
> (a+b)^4:expand(%);
```

$$a^4 + 4a^3b + 6a^2b^2 + 4ab^3 + b^4$$

```
> expand(exp(a+ln(b)));
```

$$e^a b$$

```
> cos(x)^3:combine(%);
```

$$\frac{1}{4}\cos(3x) + \frac{3}{4}\cos(x)$$

```
> f := x*(x+1)+y*(x+1);
```

$$f := x(x+1) + y(x+1)$$

```
> collect(f,x);
```

$$x^2 + (1+y) * x + y$$

```
> P:=1+X^8+X^9+X+6*X^2+6*X^3+2*X^9+5*X^2-6*X-2;
```

$$P := 3X^9 + X^8 + 6X^3 + 11X^2 - 5X - 1$$

```
> sort(%);
```

$$3X^9 + X^8 + 6X^3 + 11X^2 - 5X - 1$$

Manipulation des expressions usuelles

▷ **expand** : développe par distributivité des expressions polynomiales, calcule les fonctions circulaires en nx en fonction de leur valeur en x, applique les règles de calcul sur les logarithmes et les exponentielles ;

▷ **combine** : linéarise s'utilise avec différentes options (cf. l'aide de **Maple**). Permet par exemple d'exprimer le sinus et le cosinus à l'aide de l'exponentielle complexe (ie applique les formules d'Euler !) ;

▷ **convert** : pour faire passer une expression d'une forme à une autre (par exemple de l'écriture décimale à l'écriture binaire pour les entiers). Voir l'aide en ligne pour les autres fonctionnalités ;

▷ **collect** : effectue des regroupements ;

▷ **sort** : écrit un polynôme selon les puissances décroissantes.

La commande **convert** permet de convertir une expression en une autre. Elle permet par exemple de *transformer* un développement de Taylor en une expression du type *polynom* en extrayant sa partie principale. Elle permet également la conversion des entiers d'une base de numération à une autre, la décomposition en éléments simples de sfractions rationnelles : on l'aura compris, la commande **convert** offre de très nombreuses fonctionnalités, on consultera l'aide en ligne afin d'explorer l'étendue de ses possibilités.

```
> convert(9, binary);
```

$$1001$$

```
> convert(1.892,fraction);
```

$$\frac{473}{250}$$

```
> f := sinh(x)+sin(x): convert(f, exp);
```

$$\frac{1}{2}e^{x} - \frac{1}{2}e^{-x} - \frac{1}{2I}(e^{Ix} - e^{-Ix})$$

Exercice 5.

Vérifier avec Maple que le produit de 4 nombres entiers en progression arithmétique auquel on ajoute la puisance quatrième de la raison est toujours un carré.

Exercice 6.

Déterminer une équation à coefficients entiers du septième degré dont $2\cos(\pi/7)$ est solution. On pourra partir de **expand(cos(7*t))**.

Exercice 7. *La commande de conversion*

Lire la rubrique consacrée à la commande **convert** de l'aide en ligne.

1. Trouver l'expression de

$$\cos^3(2x)\sin^2(x)$$

en fonction de $t = \tan(x/2)$ en utilisant **convert** avec l'option **tan**. On obtiendra une fraction rationnelle en la variable t que l'on factorisera au maximum.

2. Ecrire 0.64547887876865865 sous la forme d'un quotient de deux entiers naturels au moyen de la commande **convert** avec l'option **rational**.

6. De la simplification sous Maple

En Mathématiques, on cherche souvent à présenter le résultat d'un calcul sous une forme optimale, c'est-à-dire simplifiée au maximum. Remarquons que Maple n'effectue pas toutes les simplifications :

```
> cos(x+Pi), (x^3+1)/(x+1) ;
```

$$-\cos(x),\ \frac{x^3+1}{x+1}$$

En l'abscence de simplification, l'utilisateur devra forcer le logiciel à transformer l'expression une (voire plusieurs) fois avant d'aboutir à un résultat optimal.

```
> simplify((x^3+1)/(x+1)) ;
```

$$x^2 - x + 1$$

Quelques opérateurs de simplification

▷ **normal** : simplifie les fractions rationnelles ;

▷ **simplify** : permet de simplifier des expressions algébriques. Cf. l'aide en ligne ;

▷ **rationalize** : réalise la méthode classique de multiplication par l'expression conjuguée pour simplifier des fractions rationnelles dont le dt'enominateur comprend des racines carrées.

```
> (X+3)/(X^2+4*X+3):normal(%);
```

$$\frac{1}{X+1}$$

```
> simplify(exp(a+ln(b*exp(c))));
```

$$be^{a+c}$$

```
> rationalize(1/(sqrt(2)+sqrt(3)));
```

$$\sqrt{3}-\sqrt{2}$$

Exercice 8.

Simplifier

$$\frac{\dfrac{\dfrac{1}{a}+\dfrac{1}{b}}{\dfrac{1}{a}-\dfrac{1}{b}}+\dfrac{\dfrac{1}{a}-\dfrac{1}{b}}{\dfrac{1}{a}+\dfrac{1}{b}}}{\dfrac{\dfrac{1}{a}+\dfrac{1}{b}}{\dfrac{1}{a}-\dfrac{1}{b}}-\dfrac{\dfrac{1}{a}-\dfrac{1}{b}}{\dfrac{1}{a}+\dfrac{1}{b}}}$$

Exercice 9. *Simplifications pas si simples...*

Définir et simplifier les expressions suivantes sous Maple :

1. $\exp(1)*\exp(-1)-1$;

2. $\dfrac{1}{1+\sqrt{2}+\sqrt{3}}-\dfrac{1}{4}\sqrt{2}+\dfrac{1}{2}-\dfrac{1}{4}\sqrt{2}\sqrt{3}$;

7. Les fonctions mathématiques

On peut manipuler les fonctions sous Maple de deux manières différentes : en utilisant des expressions ou en utilisant une définition plus proche des notations usuelles des Mathématiques.

7.1. Utilisation d'une expression

Définissons le cosinus comme une expresion.

```
> restart: f:=cos(x):
```

Nous avons vu que pour évaluer l'expression f en une valeur donnée, il faut *affecter à x cette valeur puis évaluer f ou utiliser la commande* **subs** :

```
> x:=Pi/5: f, subs(x=0,f);
```

$$\frac{1+\sqrt{5}}{4},\ 1$$

7.2. Modes de définition d'une fonction

Nous voudrions utiliser la notation mathématique des fonction : coder « **f(Pi) ;** » au lieu de « **x :=Pi : f ;** » pour calculer $\cos(\pi)$...

Maple permet à l'utilisateur de créer ses propres fonctions au moyen d'une syntaxe inspirée de celle des Mathématiques. Les fonctions seront également enregistrées dans des variables. Pour définir une fonction f, on utilisera la syntaxe suivante :

Définition d'une fonction f

f := x -> *expression dépendant de* x

Le symbole de la flèche s'obtient en juxtaposant les signes − et > du clavier. Contrairement au cas des expressions, le caractère libre ou non de x n'a ici aucune importance.

```
> f:=x->x+sin(x);
```

$$f := x \to x + \sin(x)$$

L'utilisateur accédera aux valeurs de la fonction f en utilisant la syntaxe usuelle des mathématiciens.

```
> f(Pi), evalf(f(Pi));
```

$$\pi, 3.141592654$$

Remarquons que la variable x est *muette*, on peut la remplacer par n'importe quelle autre lettre.

```
> f:=u->u+sin(u): evalf(f(Pi));
```

$$3.141592654$$

Si f est une fonction, « $f(x)$ » sera considéré comme une expression par **MAPLE**. Cette expression peur très bien dépendre d'autres paramètres que x.

```
> restart: f:=u->sin(t*u):t:=1:evalf(f(Pi));
```

$$0$$

```
> t:=0.5:f(Pi), t:=1.5:f(Pi);
```

$$1, -1$$

Dans cet exemple, f est en fait une expression de la variable t qui est une fonction. On peut également définir des fonctions de plusieurs variables. On retiendra alors la syntaxe suivante :

Définition d'une fonction f de plusieurs variables

f := ($x_1, \ldots, x_n$) -> *expression dépendant des* x_k

```
> g:=(x,y,z)->x*cos(y^2+z^2);
```

$$g := (x, y, z) \to x\cos(y^2 + z^2)$$

Une autre possibilité est l'utilisation de la commande **unapply**. Pour définir la fonction f qui à la variable *nom de la variable* associe *expression*, on appliquera la syntaxe suivante :

Définition d'une fonction par **unapply**

f := unapply(***expression, nom de la variable*** **)**

```
> f:=unapply(sin(t)*cos(t),t);
```

$$f := t \rightarrow \sin(t)\cos(t)$$

7.3. Fonctions définies par morceaux

On retiendra la syntaxe suivante pour définir une fonction f valant f_i si la condition cond_i est vérifiée (i variant de 1 à n) et valant f_{sinon} dans les autres cas [15].

Définition d'une fonction par morceaux

$$f := x \text{->} \textbf{piecewise}(\text{cond}_1,\ f_1,\ \text{cond}_2,\ f_2,\ \ldots,\ \text{cond}_n,\ f_n,\ f_{\text{sinon}})$$

Attention, la commande **piecewise** ne définit qu'une expression ! On retiendra également que les conditions $cond_i$ sont testées successivement dans l'ordre croissant des indices [16]. On tirera tous les enseignements de l'exemple qui suit.

```
> f:=x->piecewise(x<0,-1,x<1,0,1);
```

$$f := x \rightarrow \text{piecewise}(x < 0, -1, x < 1, 0, 1)$$

```
> f(-2);f(0.5);f(2);
```

$$-1, 0, 1$$

7.4. Fonctions usuelles

Maple dispose d'une librairie de fonctions usuelles prédéfinies. En voici un aperçu non exhaustif ...

15. La valeur par défaut de f_{sinon} est zéro.

16. Ce qui permet de définir une fonction par morceaux d'une autre manière qu'en mathématiques (où l'on utilise une partition de l'ensemble de définition pour définir une fonction par morceaux).

Fonctions usuelles sous Maple

- L'exponentielle : **exp** ;
- La racine carrée : **sqrt** ;
- Le logarithme népérien : **ln** ***ou*** **log** ;
- $\sqrt[n]{x}$ pour $x \in \mathbb{R}$ et n impair : **surd(x,n)** ;
- Cosinus, sinus, tangente, cotangente : **cos, sin, tan, cot**.
- Cosinus, sinus, tangente hyperboliques : **cosh, sinh, tanh** ;
- Arcosinus, arcsinus, arctangente : **arccos, arcsin, arctan** ;
- Argument cosinus, argument sinus, argument tangente hyperbolique : **arccosh, arcsinh, arctanh**.

On se souviendra que les nombres réels e et π sont respectivement codés **exp(1)** et **Pi** sous Maple.

7.5. Opérations sur les fonctions

Maple est capable d'effectuer les opérations algébriques élémentaires sur les fonctions telles que la somme, la produit, etc.

```
> restart:f:=x->x:g:=x->sin(x):h:=f+g, h(Pi), i:=f*g, i(Pi);
```

$$h := f+g, \pi, i := fg, 0$$

Opérations usr les fonctions

- **f + g** : somme de f et g ;
- **f * g** : produit de f et g ;
- **f / g** : quotient de f par g ;
- **f @ g** : composée $f \circ g$;
- **f @@ n** : composée $f \circ \cdots \circ f$ (n fois).

Exercice 10.

On défini la suite $(u_n)_{n\in\mathbb{N}}$ par $u_0 = 1.5$ et $\forall n \geqslant 0,\ u_{n+1} = \sin(u_n)$. Calculer u_{2009}.