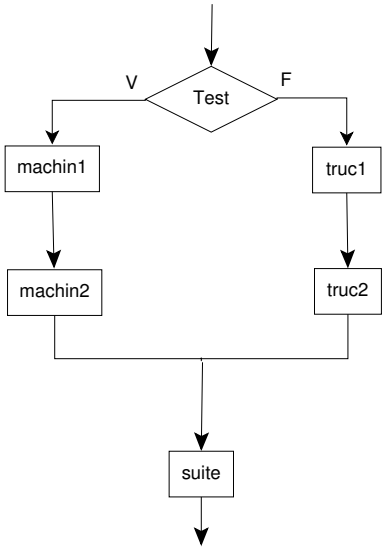
Eléments de syntaxe du langage de programmation Maple




```
if test then
  machin1;
  machin2;
else
  truc1;
  truc2;
fi;
suite;
```

FIG. 1 – structure de contrôle "if"

Vous pouvez utiliser les feuilles de calcul Psyntelem.mw et Psyntelem.mws (télécharger et enregistrer avant d'ouvrir).

Une session de travail maple est un document qui mélange du texte inerte et des enchaînements d'instructions qui peuvent être exécutées. En ce qui concerne le texte inerte, le logiciel dispose de fonctionnalité de type traitement de texte (paragraphes, styles, ...).

Il faut savoir passer du mode "texte" au mode "intructions".
Une intruction commence par un >

```
> bouh! ;
```

est une instruction.
Une instruction doit toujours se terminer par un " ;" (ou un " :").
Pour exécuter une instruction, il suffit de placer le curseur n'importe où dans la ligne la contenant et appuyer sur la touche "entrée".
En réponse à l'exécution, Maple modifie son état et affiche (en bleu ou en rouge violet) une réponse. Une réponse en violet traduit une erreur de syntaxe et vous donne des indications sur le type de l'erreur. Une réponse en bleu signifie que l'instruction s'est bien exécutée ce qui ne veut pas dire qu'elle a fait ce que vous pensiez qu'elle ferait. Il faut toujours lire et chercher à comprendre cette réponse il NE SERT JAMAIS À RIEN DE L'EFFACER. Si la réponse n'est pas celle que vous attendez, corrigez l'instruction et exécutez à nouveau.
Les instructions sont sensibles aux majuscules.
Si vous voulez exécuter successivement plusieurs instructions. Il est conseillé de les placer dans une *zone d'exécution*. Pour cela, à la fin d'une ligne, au lieu d'appuyer sur "entrée" pour aller à la ligne (ce qui provoquerait l'exécution) il faut appuher sur "shift (flèche vers le haut comme pour les majuscules) entrée". Vous allez alors à la ligne sans exécuter. La zone d'exécution est matérialisée par une ligne verticale sur la gauche. Pour exécuter les instructions, placer le curseur n'importe où dans la zone et appuyer sur "entrée".
Maple n'interprète pas les espaces dans une commande ce qui permet l'*indentation*. Il s'agit d'une règle de présentation du code qui consiste aligner à gauche des lignes de même niveau d'imbrication et à ajouter des espaces (par exemple deux espaces ou une tabulation) en début de ligne chaque fois que l'on augmente le niveau.
Vous devrez OBLIGATOIREMENT indenter votre code.
Voila quelques éléments de syntaxe utiles au débutant :

\# commentaire. Un commentaire n'est pas exécuté.

; fin d'instruction avec affichage.

: fin d'instruction sans affichage.

:= assignation. Le code pour le symbole ⟵ des diagrammes.

Vous pouvez aussi commencer à manipuler les éléments suivants.

pi la lettre. Utilisée seule, elle est par défaut le nom d'une variable

Pi le nom (réservé par Maple) du nombre célèbre

Digits la variable (réservé par Maple) désignant le nombre de décimales utilisé dans les calculs en virgule flottante (décimaux).

evalf(expression) la fonction d'évaluation décimale (floating) d'une expression.

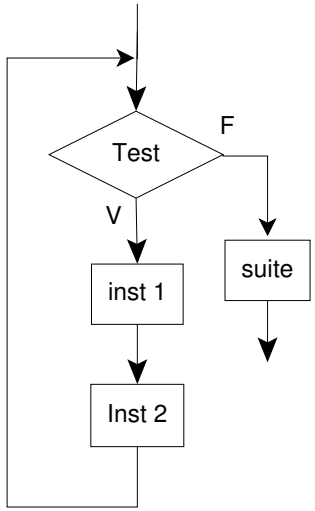
irem(a,b) le reste de la division euclidienne de a par b. Il s'agit de la division des nombres entiers (i : integer, rem : remainder).

iquo(a,b) le quotient de la division euclidienne entière de a par b.

Exécutez les commandes suivantes

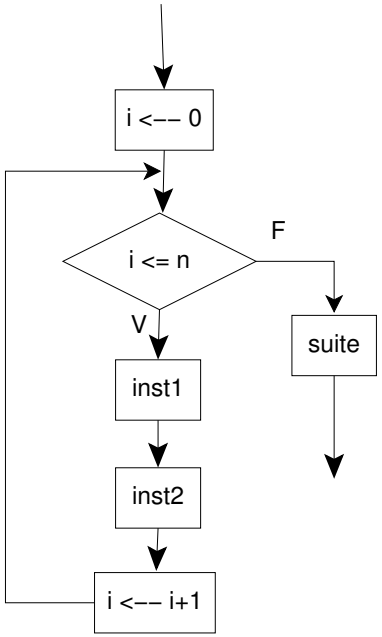
```
> Pi; cos(x); cos(Pi);
> evalf(Pi); evalf(pi);
> Digits :=2; evalf(Pi);
> Digits :=200; evalf(Pi);
> Pi := 3.14;
```

et comprenez les réponses.




```
while test do
  inst1;
  inst2;
od;
suite;
```

FIG. 2 – boucle "while"




```
for i from 0 to n do
  inst1;
  inst2;
od;
suite;
```

FIG. 3 – boucle "for"

Implémentation de la structure de contrôle : "if"

Certaines configurations d'enchaînements d'instructions se retrouvent souvent. La structure de contrôle "if" est implémentée dans la figure 1 en schéma conventionnel et en syntaxe maple.

Implémentation des boucles : "while", "for"

La figure 2 implémente la *boucle* "while".
La boucle "for" est en fait d'une boucle "while" particulière dite non conditionnelle.

On sait à l'avance combien de fois on doit exécuter une séquence d'instructions donnée et on utilise un compteur qui est incrémenté automatiquement. On peut toujours l'implémenter aussi avec une boucle "while" :

```
i:= 0;
while i <= n do
  inst1;
  inst2;
  i := i+1;
od;
suite;
```