Tutoriel pour le langage ToySanta, version 1.0...

En ToySanta, une ligne de code est composé de trois types d'éléments:

FONCTION [PARAM] **ARGUMENT1**, ARGUMENT2, ARGUMENT3...

Exemple simple de ligne de code:

```
print "bonjour..."
```

Maintenant, on va essayer quelque chose de plus compliqué:

```
= 7

loop [
     print "Bonjour, number ", text $
/]
```

Sur la ligne 1, le symbole '**=**' est une fonction, et l'argument '**7**' est ce qui est mis dans la variable [PARAM]. Puisque [PARAM] est facultatif, une variable spéciale, '$' est utilisée lorsqu'une fonction a besoin de modifier une variable. Donc, ici, la fonction **=** mets la valeur **7** dans la variable $.

Sur la prochaine ligne, **loop** est la fonction, suivi d'une parenthèse carrée. Cette parenthèse se nomme 'post-argument', et elle permet de délimiter un bloc de lignes de code. La fonction **loop** est en fait une boucle, qui décrémente son paramètre jusqu'à ce qu'il soit égale à zéro. S'il n'y a pas de paramètre, alors c'est $ qui sera décrémenté.

La prochaine ligne doit être familière, c'est **print**. Mais cette fois-ci, elle a deux arguments. Le second argument commence par '**text**', parce que la variable $ doit être convertie en format '**text**' avant de pouvoir être accepté par la fonction **print**.

La dernière ligne, est la fonction '**/]**', qui termine la délimitation. Une fois la délimitation terminée, la boucle peut donc exister. La fonction **loop** teste alors la variable spéciale $. Si la variable est supérieure à 0, alors la boucle sera exécuté, et $ sera décrémentée. Lorsque $ devient 0, la boucle s'arrête.

On va maintenant faire un peu plus difficile:

```
macro <message> <
     text <msg>
     print msg
/>
```

```
text <travail>

=7|loop [
     =<travail> "Bonjour, number ", text $
     message msg travail
/]
```

La première fonction, **macro**, crée une nouvelle fonction qui se nomme **message**. La fonction **message** imprime à l'écran le contenu de la variable **msg**, qui est créé à chaque fois que la fonction **message** est utilisée. Notez, chaque variable créée dans une fonction est détruite lorsque la fonction se termine. C'est imprudent de créer des variables à l'intérieur d'une **loop**, parce que la destruction peut survenir plus tard qu'après que la **loop** se soit terminée.

Après la **macro**, nous avons une fonction **text**, qui crée une variable **travail**. La variable **travail** est donc une variable de type '**text**', qui peut contenir un texte.

Ensuite, on sait ce que **=7** fait, et ce que **loop** [fait. Le | entre les deux permet d'avoir plus d'une ligne de commande par ligne de texte.

Dans la **loop**, la fonction **=** copie dans son paramètre, le message **"Bonjour, number "**, suivi du nombre contenu dans la variable $, sous forme de texte. Notez ici, que les guillemets qui délimitent le paramètre de la fonction **=** doivent être les mêmes que ceux utilisées pour créer la variable. Si par exemple, la variable **travail** avait été créée avec des parenthèses carrées, alors **=** devra aussi avoir des parenthèses carrées. Comme ceci:

```
text [travail]
...
= [travail] ...
```

Ensuite, dans la **loop**, après la fonction d'assignation **=**, la fonction **message** est appelée. Souvenez-vous qui vous avez précédemment créé cette fonction. La fonction message peut prendre n'importe quel paramètres, mais ces paramètres doivent être précédées du nom de la variable dans la **macro**. Notez que c'est valide que pour les variables de la **macro** qui sont créées entre parenthèses triangulaires. Les variables à l'intérieur de la **macro**, qui sont créés avec des parenthèses carrées, ne sont pas reconnus par les arguments. Mais une variable avec des parenthèses carrées peuvent toutefois être utilisées comme arguments d'une **macro**.

Alors, le contenu de la variable **travail** est copié dans la variable **msg** de la fonction **message**. Et ensuite, la fonction **message** imprime le contenu de la variable **msg**.