



Séance 4 : Boucles et conditions avec le langage *Logo*

1. Les instructions

Pour donner un ordre à un robot ou à une machine, il faut utiliser un langage qui soit proche de la langue des humains (le français, l'anglais...) **et** qui soit compréhensible par la machine sans pour autant lui parler en utilisant directement son langage (des suites de « 0 » et de « 1 »). Nous allons voir ici un langage simple permettant de communiquer avec la machine : le langage *Logo*. Ce langage est composé d'un certain nombre d'ordres (des morceaux de phrase) que l'on peut donner à la machine, on appelle ces ordres des *instructions*.

2. La boucle répéter

Si nous voulons apprendre à notre robot à monter une marche, il peut se rendre à pieds à un étage (de dix marches par exemple). Il va devoir exécuter dix fois l'instruction (l'ordre) qui lui permet de monter une marche afin d'atteindre un étage. **On répète l'instruction « monte une marche » autant de fois qu'il y a de marches dans l'étage à monter (10 fois dans notre exemple).**





- En langage *Logo*, l'instruction (l'ordre) qui permet de faire **plusieurs fois** une série d'instructions correspond à

```
repeat n [ instructions ]
```

- Ceci signifie que l'on répète ("repeat") les ordres qui sont donnés dans [instructions] *n* fois. Ici *n* est donc un nombre.
- Dans notre exemple nous avons donc :
 - "*n*" qui correspond au nombre de marches de l'étage (10 marches)
 - "*instructions*" qui correspond à l'action « monte une marche ». Attention, dans notre exemple « instructions » ne donne qu'un ordre à répéter, il pourrait aussi y en avoir plusieurs.

Application 1 : exemple du carré

Nous te donnons quelques *instructions* en langage *Logo* :

- **Faire avancer la tortue représentée par un triangle** 
`forward n` : fait avancer la tortue de *n* pas. *n* est un nombre.
- **Faire tourner la tortue** 
`right 90` : fait tourner la tortue à droite. 
`left 90` : fait tourner la tortue à gauche. 

A toi de jouer maintenant...

Dans la partie gauche du tableau ci-dessous, nous te donnons un *programme Logo* qui dessine un carré. Ce programme possède de nombreuses *instructions* qui se répètent. Dans la partie droite du tableau, écris un programme qui dessine ce même carré **mais en utilisant la boucle répéter (repeat)**.

Sans boucle 'répéter'	Avec boucle 'répéter'
<pre>forward 50 right 90 forward 50 right 90 forward 50 right 90 forward 50 right 90</pre>	

Tu vas maintenant tester ton programme avec le *langage logo*.

Lance le logiciel Logo en cliquant deux fois sur l'icône :



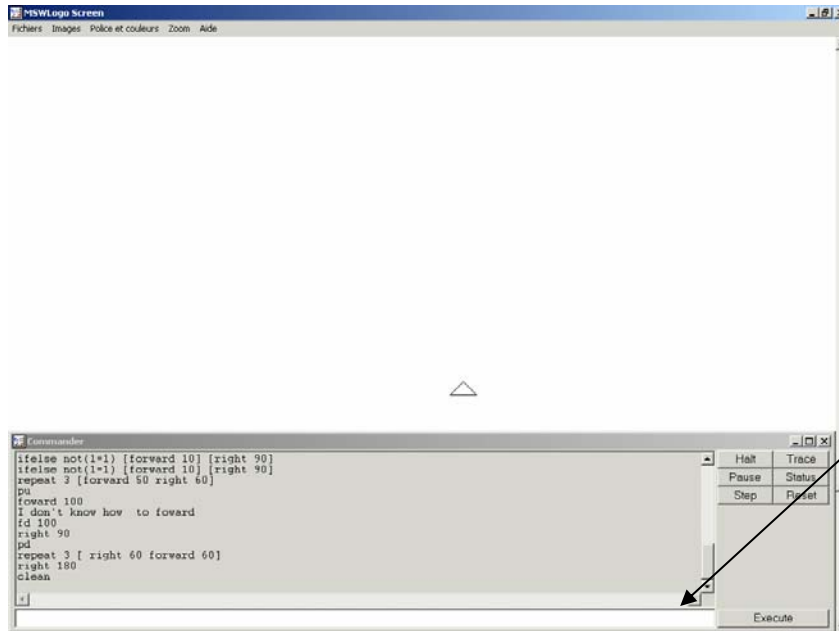
Le logiciel est lancé.

Tu vois dans la partie supérieure un écran blanc dans lequel il y a la tortue représentée par un triangle .



La partie inférieure représente les **commandes du programme**.

Pour tester ton programme, tu vas l'écrire sur la ligne blanche la plus basse.



Lorsque tu as écrit ton programme (attention de ne pas faire de fautes, le logiciel ne comprendrait pas !), clique sur « execute » (bouton en bas à droite de ton écran). Si la tortue a dessiné un carré alors, c'est gagné !!!! Bravo, tu as écrit ton premier programme en *langage Logo*.

Application 2 : exemple du rectangle

Maintenant tu vas écrire un *programme logo* pour dessiner un rectangle de **largeur 50** et de **longueur 90**.

- 1) Dans un premier temps, tu vas écrire ton programme dans le tableau ci-dessous. Tu vas d'abord écrire un programme **sans utiliser la boucle répéter** (partie gauche du tableau) puis **en utilisant la boucle répéter** (partie droite du tableau).

Sans boucle 'répéter'	Avec boucle 'répéter'

2) Maintenant tu vas tester ton programme qui utilise la boucle **répéter** avec le logiciel *Logo* pour vérifier que ta tortue dessine bien le rectangle souhaité. Ainsi tu tapes ton programme et tu cliques sur « execute » comme dans l'exercice précédent.

3. Conditions

Lorsqu'on souhaite livrer un paquet chez Monsieur Dupond qui habite dans un immeuble, on peut prendre l'ascenseur ou monter à pieds. Ce choix est réalisé mentalement, en prenant en compte des raisons comme l'étage où il faut se rendre, la forme physique dans laquelle on est, le poids du paquet...

On pourrait traduire les réflexions par : **si** Monsieur Dupond habite au premier étage, **alors** je monterais à pieds, **sinon** je prendrais l'ascenseur.

si condition alors {instructions1} sinon {instructions2}

La « condition » est évaluée comme étant *vraie* ou *fausse*, si la condition est vraie, alors on exécutera les ordres qui sont dans « instructions1 » sinon on exécutera les ordres qui sont dans « instructions2 ». La **condition** est souvent le résultat de comparaisons :

Opérateur	Signification	Exemple
étage = 1	étage égal à 1 ?	si (étage = 1) alors (monter à pieds)
not (étage = 1)	étage différent de 1	si not (étage = 1) alors (prendre l'ascenseur)
étage < 2	étage inférieur à 2	si (étage < 2) alors (monter à pieds)
étage > 1	étage supérieur à 1 ?	si (étage > 1) alors (prendre l'ascenseur)



Condition



instruction

NB : On peut aussi écrire des conditions plus compliquées comme, par exemple, « si (**étage = 1 et paquet_est_léger**) alors (monter à pieds) », nous reverrons cela lors d'une prochaine séance ☺

- En *langage Logo*, la condition s'exprime de la manière suivante :

ifelse condition [instructions1] [instructions2]
--

- Ceci signifie que l'on exécute "instructions1" si "condition" est respectée. Dans l'autre cas, on effectue "instructions2".

Dessine ce que trace la tortue lorsque tu exécutes le programme ci-dessous :

```
ifelse 3+2=6 [forward 100] [right 90 forward 100]
```



Dessine ce que trace la tortue lorsque tu exécutes le programme ci-dessous :

```
ifelse 3+2=5 [forward 100] [right 90 forward 100]
```

