

## 2 - Introduction à la programmation

**Héctor Satizábal**

IICT- HEIG-VD

2018 - 2019

- ▶ Environnement: `ca`
- ▶ Création: `crt`
- ▶ Mouvement: `fd`, `bk`, `rt`, `lt`
- ▶ Position: `setxy`, `set heading`
- ▶ Dessin: `pd`, `pu`

Adapté à partir du matériel du cours créé par **Prof. Andres Perez-Uribe**

# Comment programmer un ordinateur



# Compilation et assemblage

```
var
  a,b,c : Integer;
begin
  a := 5;
  b := 8;
  c := a + b;
end
```

```
load $4, 50($0)
load $5, 54($0)
add  $2, $4, $5
store $2, 58($0)
```

```
1010 1111 1011 1111
0100 0000 0011 0010
1010 1111 1011 1111
0101 0000 0011 0110
0010 0111 1011 1101
1111 1111 1110 0000
1010 0000 1011 1111
0010 0000 0011 1010
```

Langage de  
haut niveau

→  
compilation

Langage  
assembleur

→  
assemblage

Langage  
machine

# Exemples de “Hello world”

## Assembleur

```
.data
str:    .ascii "Hello World!\n"
.text
main:   li      $v0, 4
        la      $a0, str
        syscall
        li      $v0, 10
        syscall
```

## Ada

```
-- Hello World
with Ada.Text_IO;
use Ada.Text_IO;

procedure HelloWorld is
begin
    Put_Line("Hello World");
end HelloWorld;
```

## C++

```
#include <iostream.h>

int main(int argc, char *argv[])
{
    cout << "Hello World" << endl;
    return 0;
}
```

## Java

```
public class HelloWorld
{
    public static void main(String args[])
    {
        System.out.println("Hello World");
    }
}
```

## Pascal

```
PROGRAM HelloWorld;

BEGIN
    WRITELN('Hello World');
END.
```

## Lisp

```
; LISP
(DEFUN HELLO_WORLD ()
  (PRINT (LIST 'HELLO 'WORLD)))
```

# La programmation, qu'est-ce que c'est?

- ▶ La programmation dans le domaine **informatique** est l'ensemble des activités qui permettent l'écriture des **programmes informatiques** ([fr.wikipedia.org](https://fr.wikipedia.org))
- ▶ Un programme informatique indique à l'ordinateur ce qu'il doit faire (sous la forme d'une série d'opérations à effectuer) pour résoudre un problème
- ▶ La suite d'opérations nécessaires pour résoudre un problème est appelée algorithme
- ▶ Un algorithme est mis en oeuvre lorsqu'on le décrit à l'aide d'un langage de programmation

# Algorithmme

Le nom du mathématicien perse Al-Khwārizmī, latinisé au Moyen Âge en Algoritmi, puis en Algorisme par les Européens, est à l'origine du mot algorithme, qui veut dire “procédure”



Mais, le principe des algorithmes était connu depuis l'antiquité (algorithme d'Euclide)

# Un exemple d'algorithme

Mettre ces mots dans l'ordre alphabétique:

algorithme, magique, informatique, école, heig-vd, assembleur

# Un exemple d'algorithme

Mettre ces mots dans l'ordre alphabétique:

algorithme, magique, informatique, école, heig-vd, assembleur

Comment avez-vous fait?

1. Chercher des mots qui commencent par 'a'
2. S'il y a un mot qui commence par 'a', le placer au début de la liste
3. Chercher s'il y a un autre mot qui commence par 'a'
4. S'il y a un autre mot qui commence par 'a', comparer sa deuxième lettre avec la deuxième lettre du premier mot qui commençait par 'a'
5. Si ces deux lettres sont différentes, placer ces mots par ordre alphabétique selon la deuxième lettre, sinon procéder de la même façon avec la troisième lettre, et ainsi de suite
6. Refaire la même procédure pour 'b' et chaque lettre selon l'ordre alphabétique



# Encore un exemple d'algorithme

- ▶ Compter le nombre de mots dans la phrase:  
“la programmation peut être très amusante”

# Encore un exemple d'algorithme

- ▶ Compter le nombre de mots dans la phrase:  
"la programmation peut être très amusante"
- ▶ Comment avez-vous fait?

Vous savez qu'un mot est une séquence de lettres et que les mots sont séparés par des espaces, ainsi vous avez trouvé 6 mots dans la phrase en question

- Maintenant, combien de mots y a t-il dans la phrase:

“to be or not to be,that is the question”

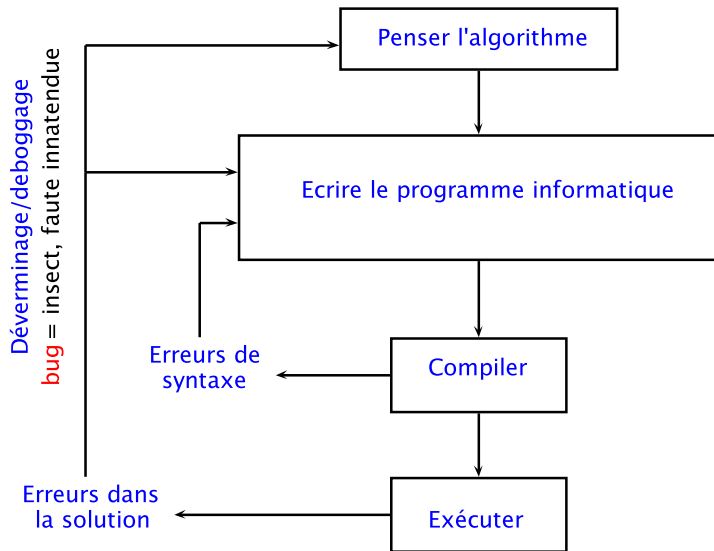
- Maintenant, combien de mots y a t-il dans la phrase:

“to be or not to be,that is the question”

- Il y a bien 10 mots, si nous prenons en compte que les mots dans une phrase ne sont pas séparés que par des espaces, mais aussi par des virgules!
- Finalement, la règle peut être complétée en considérant des traits d'union, des sauts de ligne, etc... comme séparateurs de mots

- ▶ Puisque les ordinateurs sont “idiots”, la programmation implique de rendre explicite toutes ces règles qu’on applique presque inconsciemment pour résoudre des problèmes, tels que l’ordonnancement des mots par ordre alphabétique, ou le comptage des mots dans une phrase
- ▶ L’avantage des ordinateurs est leur complète obéissance et leur rapidité!

# Procédure pour programmer une application



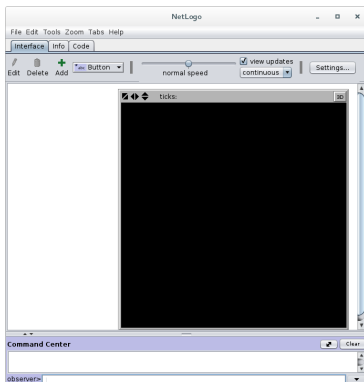
# Environnement de programmation

## Outils:

- ▶ Editeur (pour écrire les programmes)
- ▶ Compilateur (pour compiler)
- ▶ Dévermineur (pour debugger)

## Aide à la programmation:

- ▶ Indique la ligne de code qui génère des erreurs lors de la compilation
- ▶ Documents sur le langage
- ▶ Couleurs pour le code
- ▶ Exécution instruction par instruction
- ▶ etc...



# Netlogo (Uri Wilensky, Northwestern University)



- ▶ Langage simple mais puissant
- ▶ Environnement graphique et interactif
- ▶ Conçu pour la modélisation de systèmes décentralisés (tels que le comportement d'une fourmilière, des embouteillages de voitures ou le marché économique)
- ▶ Il se prête bien à l'enseignement de la programmation



# Origine de Netlogo

- ▶ Netlogo est un langage issue du langage Logo
- ▶ Logo avait été crée dans les années 60 au laboratoire d'intelligence artificielle du Massachusetts Institute of Technology MIT par S. Papert (qui avait travaillé avec Piaget à Genève) et M. Minski
- ▶ Leur souhait était de créer un langage de programmation afin d'utiliser la puissance de l'outil informatique dans les tâches d'enseignement
- ▶ Plus que tout autre langage, Logo a été conçu dans le but de démystifier les ordinateurs et la programmation



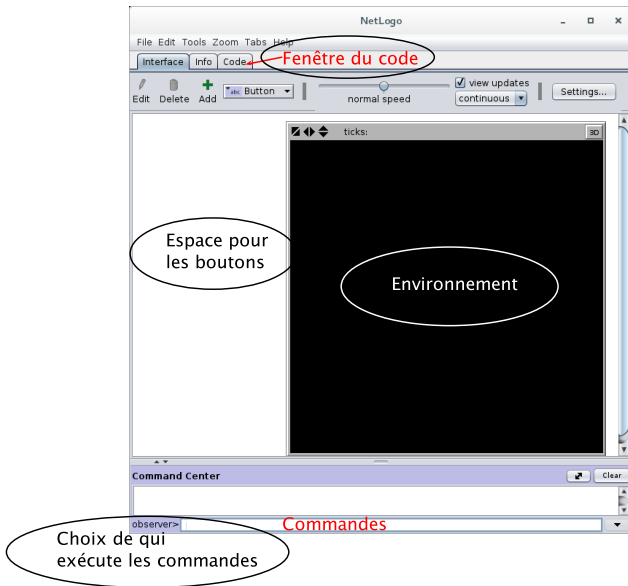
- ▶ C'est Seymour Papert qui imagina se servir de ce langage pour tracer des graphiques
- ▶ Logo était utilisé avec un ordinateur qui commandait les déplacements d'un petit robot, une "tortue" capable de laisser une trace de ses trajets grâce à un crayon dont le maniement était prévu dans le langage

# Netlogo: protagonistes du langage




- ▶ Les “tortues” (fourmis, voitures, “agents”) = éléments principaux
- ▶ Les patches = environnement des tortues : nourriture, obstacles, etc
- ▶ L'observateur = il peut superviser ce qu'il se passe et ordonner aux tortues de faire certaines choses. Il permet également d'intervenir alors même que le programme est déjà lancé


# Netlogo: fenêtres




# La tortue commence à bouger...

Sélectionner “observer” pour exécuter les commandes, et écrire:


clear-all 


create-turtles 1 

ou ca 

ou crt 1 

Sélectionner “turtles” pour exécuter les commandes, et écrire:

forward 10 

back 10 

right 45 

back 10 

left 180 

ou fd 10

ou bk 10

ou rt 45

ou fd -10

ou lt 180

# Et la tortue commence à dessiner...

Sélectionner “turtles” pour exécuter les commandes, et écrire:

<code>setxy 10 10</code>	;bouger à la position (10, 10) de l'environnement
<code>set heading 0</code>	;set heading: s'orienter vers le nord
<code>pd</code>	;pendown: placer le stylo par terre
<code>fd 5</code>	
<code>rt 45</code>	
<code>rt 15</code>	
<code>fd 5</code>	
<code>pu</code>	;penup: soulever le stylo
<code>set heading 45</code>	
<code>fd 50</code>	

- ▶ Qu'est qui se passe après exécution de l'instruction `fd 50`?
- ▶ Quelle est la différence entre `rt`, `lt` et `set heading`?

## Exercices:

1. Utiliser une tortue pour dessiner la lettre E
2. Utiliser une tortue pour dessiner la lettre A
3. Dessiner une maisonnette sans soulever le stylo:

