Introduction à la programmation

2 mai 2022

1 Objectifs

1. À savoir
   (a) Un **programme** est une liste d'instructions.
   (b) Ces instructions sont exécutées / suivies par une **machine**.
2. À comprendre
   (a) Les machines (et donc les programmes) peuvent être conçues de manières très variées.
   (b) Ces machines ne tolèrent aucune erreur dans la programmation.
   (c) La clé de la programmation est la **composition** – A voir.

2 Un peu d'histoire

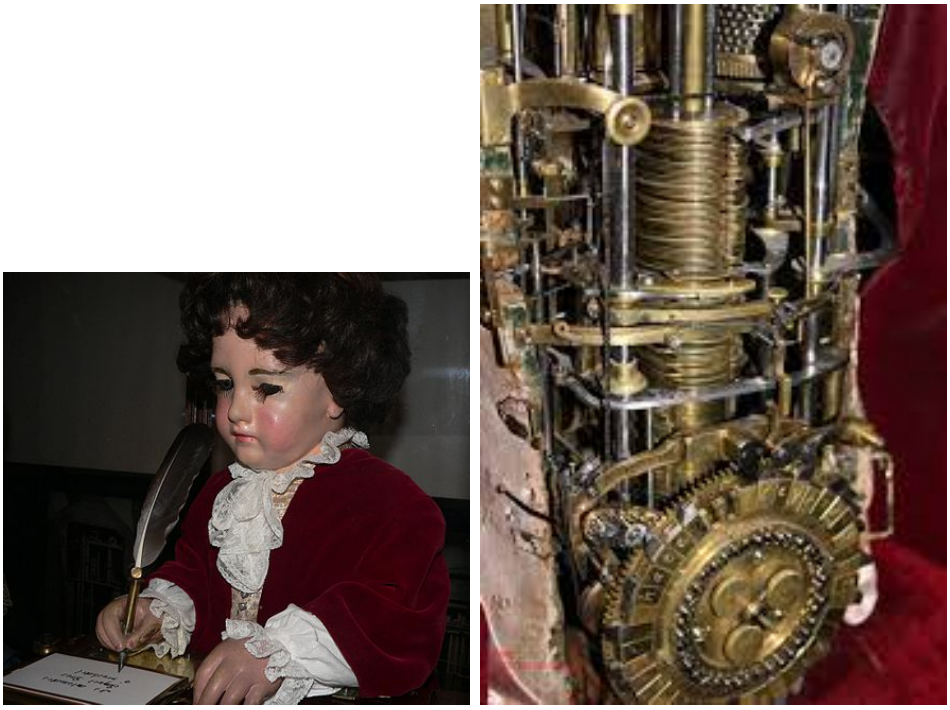


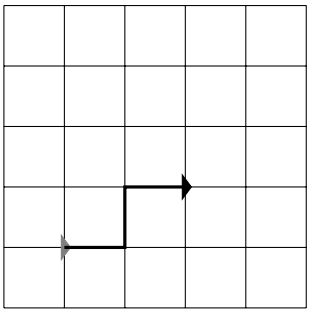
Figure 1: Automaton: Construit par la famille Jaquet-Droz (des horologiers de la Jura) autour de 1770.

1. Automata – 1770, Jaquet-Droz
2. Difference Machine/Analytic Engine – 1820/1830 Charles Babbage
3. La Bombe – 1939/1940, Alan Turing et al.
4. Ordinateurs électroniques – 1960/1970 ...

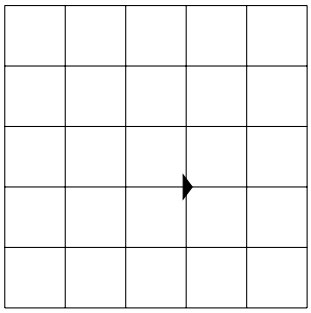
3 Débranché – un exemple

Vous êtes la machine, ou plutôt votre stylo sur le papier. Vous allez déplacer le stylo selon les instructions du programme pour produire des dessins. À tout moment le stylo "pointe" dans une certaine direction (indiqué par la tête d'une flèche); cette direction est aussi modifiée par le programme. Le programme est construit avec les instructions suivantes:

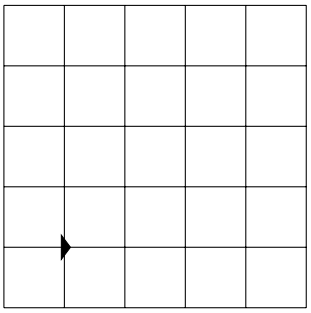
1. `F` – Avancez une unité
2. `+` – Tournez 45° sens anti-horaire
3. `-` – Tournez 45° sens horaire



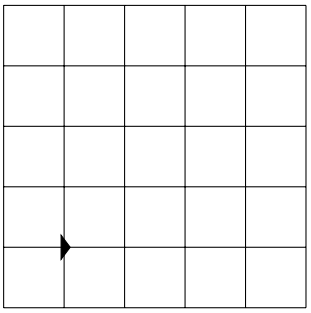
`"F++F--F"`



`"F++FF++FF++FF++F"`



`"FFF++FF++F++F--FF++F"`



`"FF++FFF++FF++F++F--F--FF--FFF--FFFF"`

4 Replit

Instructions pour accéder le travail:

1. Entrer cette URL dans la barre d'adresse du browser

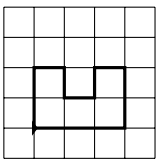
   `https://tinyurl.com/4hmwbsfu`

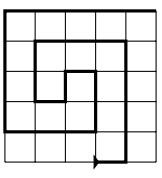
2. Choisissez un pseudonyme
3. Mot de passe : 8 caractères, 1 majuscule, 1 nombre, 1 spéciale ($ etc.)

Voici ce que vous allez voir dans l'onglet central:

```
# main.py
import machine

M = machine.Machine()
#M.test()

#### Ne changez rien au-dessus ####

p = "F++FF"

M.do(p)
```

1. Suivre le prof pour reproduire ses exemples.
2. Reproduire les dessins faits à la main dans la section précédent.
3. Écrire des programmes pour dessiner les formes suivantes:

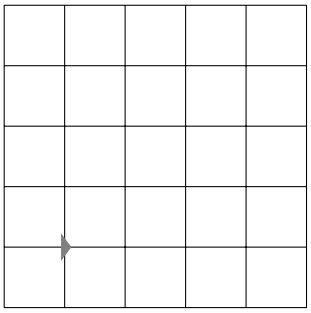
(a) 

(b) 

5 Un langage plus grand

Nous pouvons améliorer notre machine en introduisant de nouvelles instructions. À la place d'une seule lettre, nous pouvons mettre maintenant deux nouvelles instructions que nous mettons dans des parenthèses :

1. Répétitions: `"(5*F)"` veut dire `"FFFFF"`, `"(3*+)"` veut dire `"+++"`, etc.



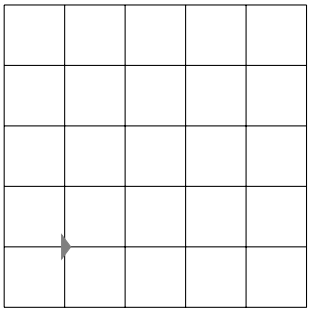
Exécutez les programmes suivants dans le cadre à côté:

   (a) `"(3*F)"`

   (b) `"(4*F)++(4*F)++F++(3*F)"`

   (c) `"F++(4*F)--F--(5*F)++F"`

2. Définitions: `"(a=F+F-)"` veut dire que chaque occurrence subséquente de `"a"` sera remplacé par `"F+F-"`. On appelle `A` une `macro`. La définition d'une macro ne change pas de tout le dessin; il faut l'**appeler** pour que ça fasse un effet.



Exécutez les programmes suivants dans le cadre à côté:

   (a) `"(a=F++)aaaa"`

   (b) `"(b=F++F--)bbbb"`

   (c) `"(l=++)FlFlFlF"`

3. Une répétition peut utiliser une macro, par exemple `"(3*a)"` où `"a"` est déjà défini.

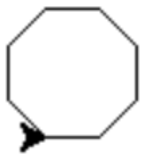
4. Une définition **ne peut pas** contenir une répétition !

5. La manière d'écrire les instructions s'appelle la **syntaxe** et ça doit être respecté absolument autrement le programme ne marchera pas.

6 Exercices

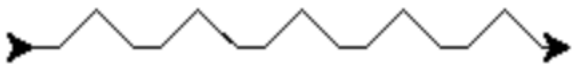
Completez ces programmes pour produire chacun des dessins ci-dessous (utilisez `Replit` pour les tester):

1. Octogone:



   Program: `"(a=F+)_____"`

2. "Shark's teeth":

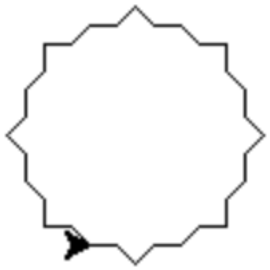


   Program: `"(a=_____)(5*a)"`

3. Tesselation:

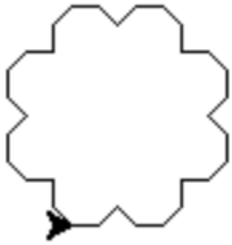
   Program: `"(a=_____)F+a--a++a--a+F"`

4. Un octogone tesselé



Program: _____

5. Un octogone tesselé autrement



Program: _____

6. Certains des programmes ci-dessous contiennent des bogues, c'est à dire des erreurs de syntaxe. Indiquer quels programmes ont une erreur et identifier l'erreur.

   (a) `"F+F-G-F"`

   (b) `"(a=F+)aaa"`

   (c) `"(a=F+)(b=F-)aba"`

   (d) `"(a=F+)3*a"`

   (e) `"(a=F+F)(b=a-a)bbb"`

   (f) `"(a=(4*F)++)a-a-a"`

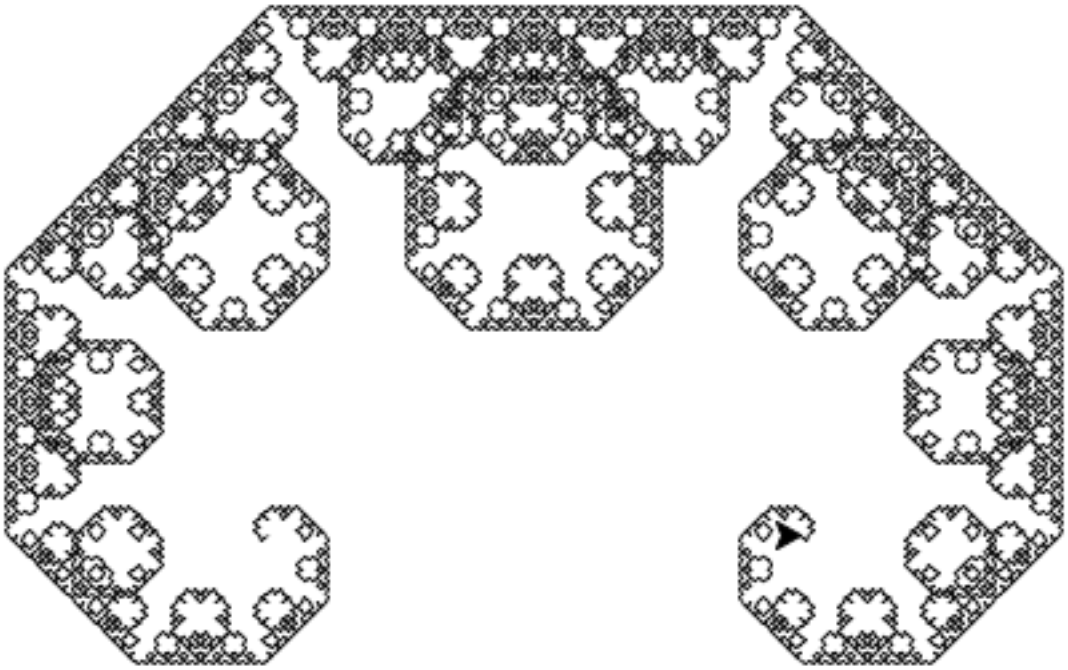


Figure 2: Le "Dragon" – Rechercher "ViHart Dragon"