Algorithmique et Développement
Initiation à la Programmation
Swing – 2020/2021

Collège Alain Fournier / Technologie pour le FUN – MOOC N°4 : Algo. / Prog.

Bibliographie : *« Initiation à la programmation », DELANNOY.*

Plan

- Le rôle de l’ordinateur.
- La notion de codage.
- Le fonctionnement de l’ordinateur.
- Dialoguer avec l’ordinateur.
- La programmation.
- Utiliser le logiciel Swing.

Rôle de l'ordinateur

- Un ordinateur est ***une machine*** qui :
  - Saisit (périphériques d'entrée),
  - Stocke (mémoire),
  - Traite (programme), et
  - Restitue (périphériques de sortie) ***des informations.***

Rôle de l'ordinateur

- Un programme, c'est une source de diversité car à chaque tâche correspond un programme ! (factures, stocks, paye, jeux …)
- L'ordinateur est capable de mettre en mémoire un programme puis de l'exécuter.
- Un programme est constitué d'une suite d'instructions.
- Une instruction spécifie :
  - Les opérations élémentaires à effectuer,
  - La façon dont elles vont s'enchaîner.
- La vitesse d'exécution de l'ordinateur fait sa puissance,
- Le programme lui donne sa souplesse.

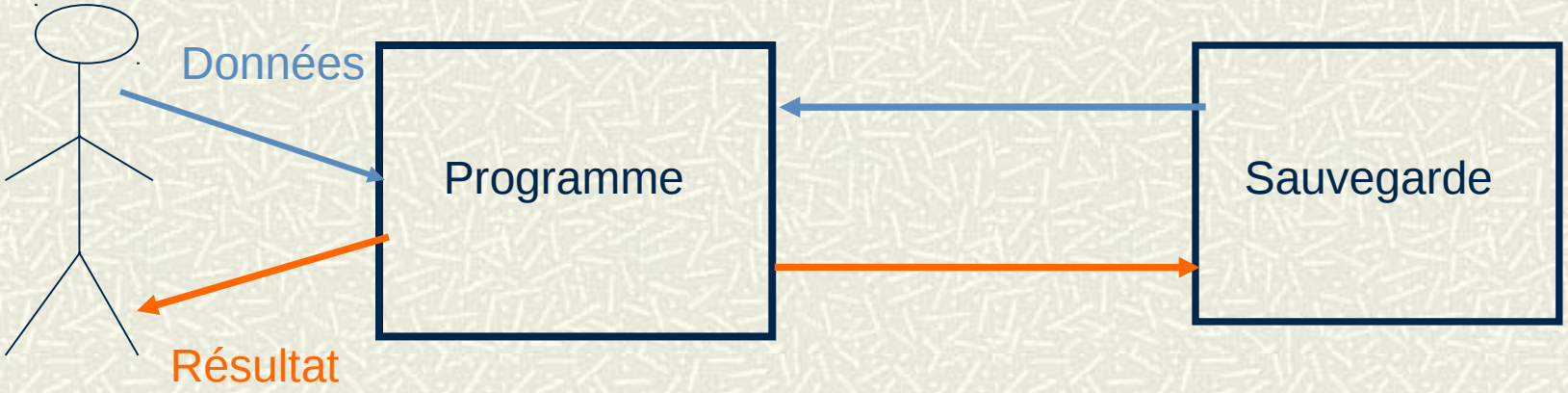
Données du programme et résultat(s)

- Exemple 1: on dispose d'un programme qui calcule la moyenne des notes :
  - Celui-ci a besoin qu'on lui fournisse les notes (données),
  - Pour qu'il nous retourne la moyenne (résultat).

- Exemple 2: établissement d'une fiche de paye :
  - Données : NSS, nombre d'heures, grade, …
  - Résultats : salaire brut, retenues, salaire net, …

Communication ou Sauvegarde ?

- D’où viennent les données ? Où vont les résultats ?



Notion de codage

- Toutes les informations traitées par l'ordinateur sont en binaire.
  - Quand on tape sur une touche du clavier, l'ordinateur la transforme en une suite de 0 et de 1.
  - Quand l'ordinateur affiche sur l'écran un résultat, il fait l'opération inverse.
- Nous aussi, nous utilisons le codage. Par exemple :
  - 11, onze, XI.
- Nous avons interprété XI comme étant le nombre 11. Comment a-t-on pu dire qu'il ne s'agissait pas des lettres X et I ?
- Pour interpréter les informations, l'ordinateur a besoin du type de données fournies.

Fonctionnement de l'ordinateur

- L'ordinateur traite l'information grâce à un programme qu'il mémorise. Il communique et sauvegarde des données.

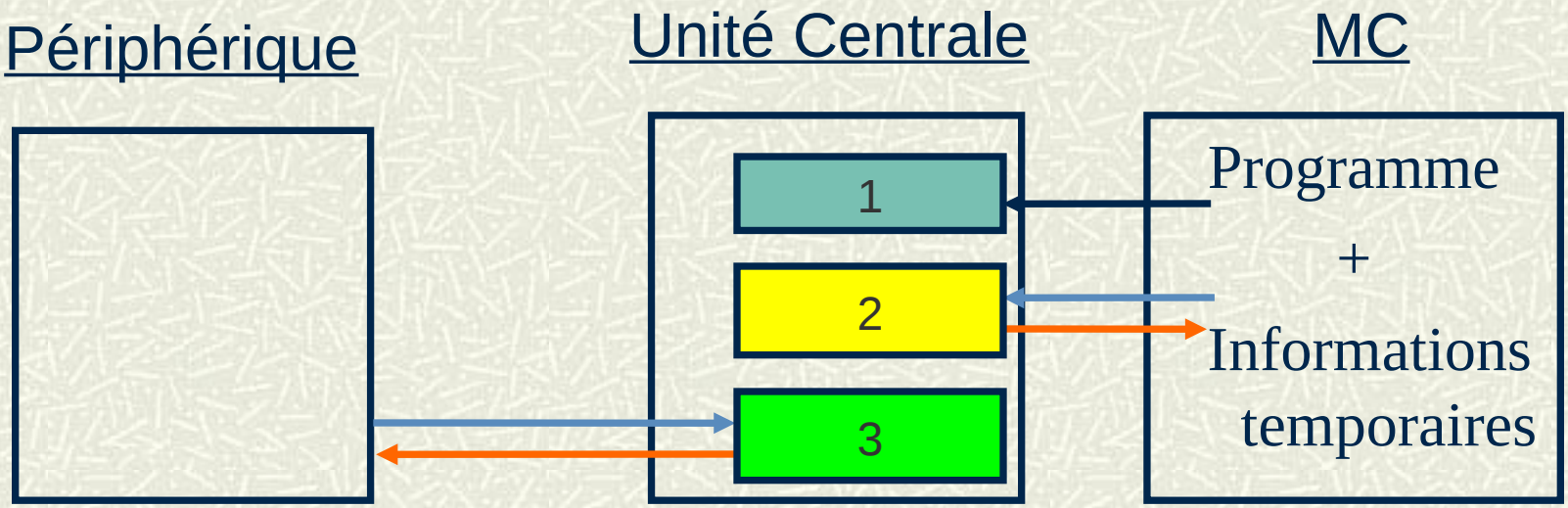
- Mémoire centrale : Programme + Informations temporaires.
- Unité centrale : chargée de prélever une à une les instructions du programme.

  - Deux types d'instructions :

    - Opérations internes (addition, soustraction, …),
    - Opérations de communication (affichage, sauvegarde, …).

- Périphériques : d'entrée (clavier, …), de sortie (imprimante, …), d'entrée/sortie (clé USB, …).

Fonctionnement de l'ordinateur



1. Prélèvement d'une instruction en Mémoire Centrale (MC).
2. Exécution de l'instruction avec possibilité d'échange avec la MC.
3. Exécution d'une instruction d'échange avec un Périphérique.

Organisation de la mémoire centrale

- C’est une ’grille’ où chaque ’case’ peut prendre la valeur 0 ou 1 (un bit).
- On ne manipule pas des cases mais des ensembles de cases qu’on appelle mots.
- En informatique, un mot correspond à un octet (8 bits = 8 binary digits).
- Chaque mot a une adresse.

L’unité centrale

- Elle sait exécuter des opérations très simples :
  - addition, soustraction, multiplication, division,
  - comparaison, communication élémentaire (un caractère).
- Chaque instruction du programme doit préciser :
  - la nature de l’opération (son code binaire),
  - la ou les adresses sur lesquelles porte l’opération.
- Les instructions sont exécutées l’une à la suite de l’autre
  - sauf si l’on rencontre une opération de branchement.

Traduction des programmes

- L'ordinateur ne comprend que le binaire mais doit-on pour autant écrire les programmes en utilisant une succession de 0 et de 1 ?

- Non, car il existe des langages de programmation dits « évolués » (proches du langage courant) et

- pour chaque langage évolué, il existe un programme qui le « traduit » en langage binaire !

- Attention : un programmeur efficace doit savoir « optimiser » les ressources informatiques en terme de taille de fichier et de vitesse de traitement (pas de place gaspillée ET pas de temps perdu !).

Exemple de traduction

- Soit : A = 5 et B = 7
- En langage évolué : Ajouter A et B
- En langage assembleur : ADD A, B
- En langage machine : 0101 010011 011010
- 0101 correspond à ADD
- 010011 correspond à l'adresse binaire de A
- 011010 correspond à l'adresse binaire de B

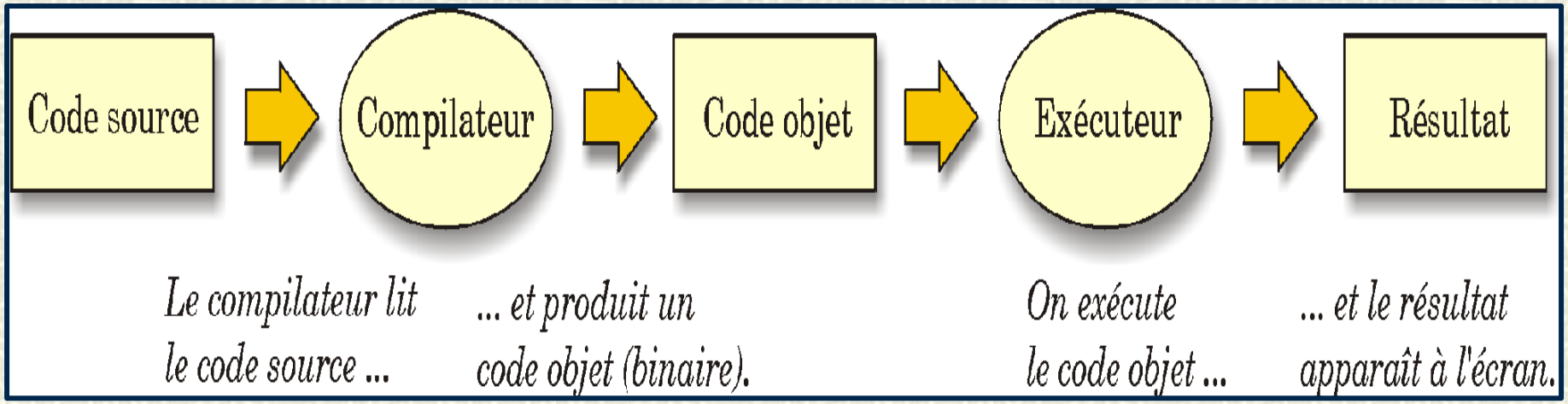
Traduction des programmes

Il existe essentiellement deux modes de traduction :

- la compilation : la traduction se fait une fois pour toutes.
- l'interprétation : à chaque fois que l'on veut exécuter le programme, l'interprète traduit une instruction à la fois. Une fois que celle-ci est exécutée, il passe à l'instruction suivante.

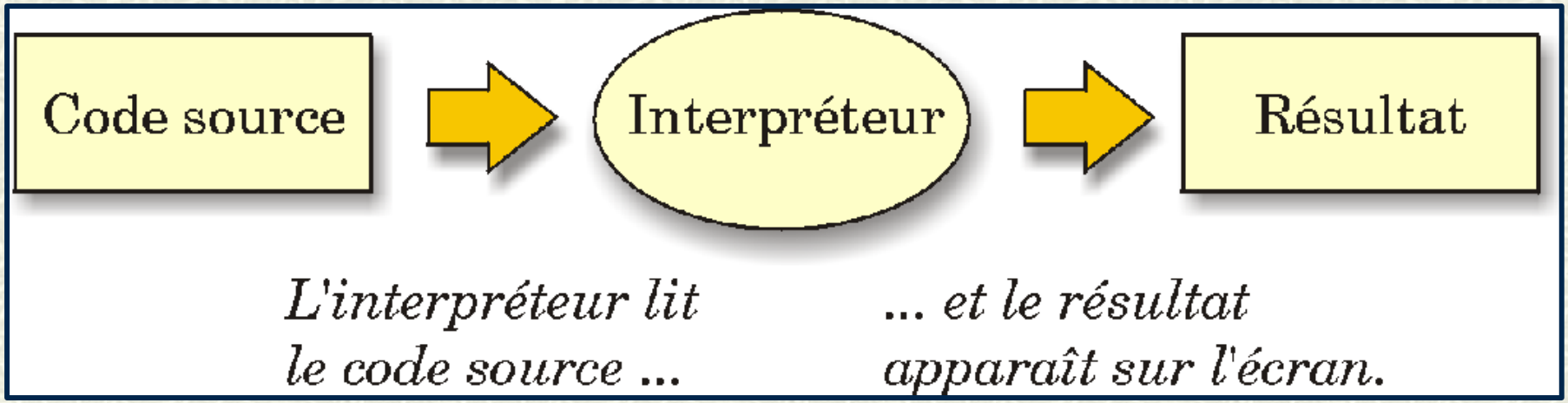
La compilation

- Schéma de principe :



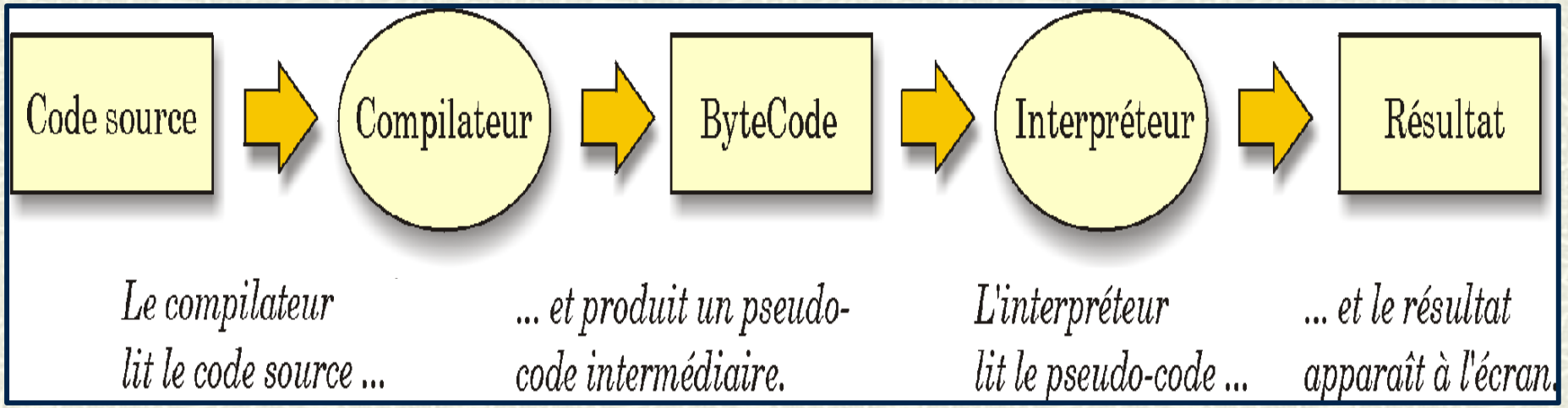
L’interprétation

- Schéma de principe :



Compilation ET interprétation

- Schéma de principe :



Le logiciel Swing fonctionne selon ce schéma et additionne les performances de chacune de ces deux techniques.

Programmation

- En pratique, il existe un large éventail de langages de programmation (C, Java, Pascal, Visual Basic, Cobol, Fortran, Python, Perl, Ruby, Caml, PHP, C++ …).
- Or, s'il existe plusieurs langages, cela veut-il dire qu'il existe aussi plusieurs sortes de programmation ?
- Non, car la plupart des langages utilisent les mêmes concepts. Dans ce cours, nous utiliserons une notation particulière: la notation algorithmique. On l'appelle aussi : pseudo-code.

Langages précurseurs

Le logiciel Swing a été écrit en langage B+. Ce langage a trois ancêtres : le CPL, le BCPL et le B.

CPL : Le **C**ombined **P**rogramming **L**anguage a été conçu au début des années 1960 - Universités de Londres et de Cambridge. Langage trop complexe → disparition dans les années 70.

BCPL : Le **B**asic **CPL (**Cambridge en 1966 par Martin Richards). Version simplifiée du CPL → Écriture d'un premier compilateur et de divers systèmes d'exploitation.

B : Ken Thompson vers 1970 dans les laboratoires Bell → version simplifiée du BCPL.

B+ : Version industrielle du langage B (années 80 et suivantes).

C : Développé par un collègue de Ken Thompson, Dennis Ritchie (années 75 et suivantes).

Pour programmer

- On suivra toujours deux étapes :

1. Analyse du problème c-à-d recherche du moyen d'aboutir au résultat à partir des données dont on dispose (=> écriture d'un algorithme),
2. Traduction de l'algorithme dans un langage de programmation.
3. Attention : apprendre à programmer, ce n'est pas apprendre un langage de programmation !

Algorithme *Al-Khawarizmi (780-850 après J.C.) - « Al-Jabr »*

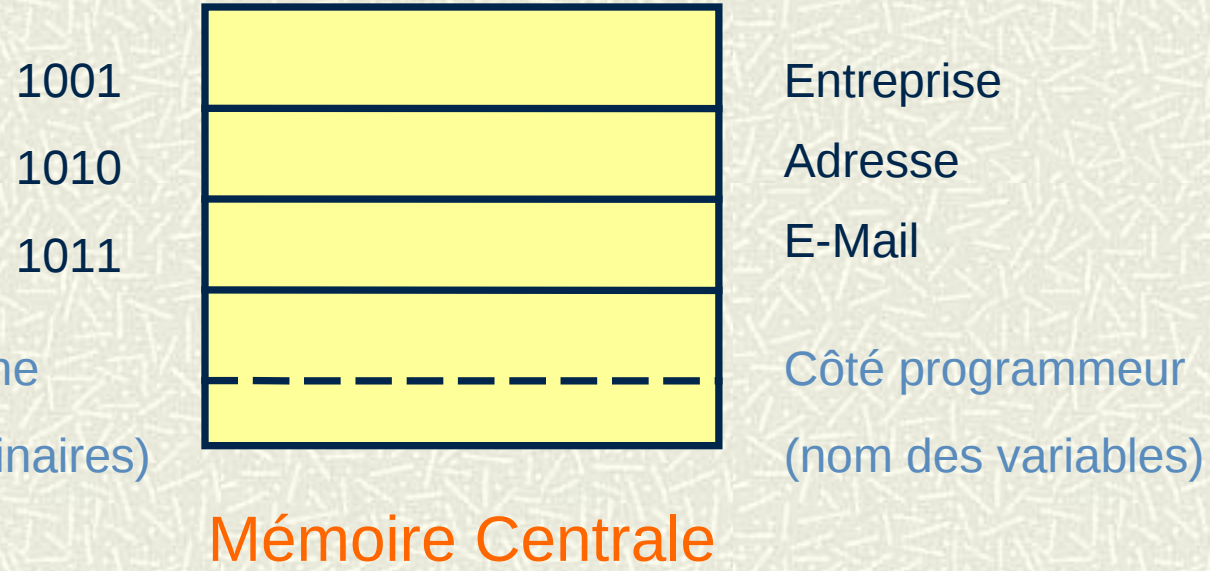
- C’est la description claire, précise et ordonnée des étapes élémentaires permettant de résoudre un problème donné.

- Exemple : classer deux nombres A et B, du plus grand au plus petit. Afficher le plus grand en premier.

1. Connaître les valeurs de A et de B.
2. Comparer A et B.
3. Si A > B alors afficher A en premier et B en second.
4. Si A = B alors afficher A B ou B A.
5. Si A < B alors afficher B en premier et A en second.

La notion de variable

- Les variables servent à « nommer » des emplacements ou des adresses de la mémoire.
- Elles permettent de « manipuler » des valeurs sans pour cela connaître leur emplacement exact.



Le type d'une variable

- Le type d'une variable permet de savoir :
  - Quelle est la nature de l'information représentée dans la variable (caractère ou numérique).
  - Quelles sont les opérations autorisées sur la variable (traitement de caractères ou calculs numériques).

- Déclaration d'une variable dans un algorithme :
  - variable Nom_de_la_variable : type
  - Exemple :
    - variables Classe, Nom : caractères
    - variables Note, Moyenne : numériques

Instruction d'affectation

- Rôle : mettre une valeur dans un emplacement mémoire désigné par son nom.
- Syntaxe :

1. Nom_de_la_variable ← valeur

   Exemple : *Note ← 15*

2. Nom_de_la_variable1 ← Nom_de_la_variable2

   Exemple : *Note1 ← Note2*

3. Nom_de_la_variable ← expression

   Exemple : *Moyenne ← ((Note1 + Note2) / 2)*

Instruction d'affectation

- Si la variable Note est égale à 10, à quoi sera-t-elle égale après l'exécution de :

  Note ← Note + 5 ? Note = 15

- A quoi seront égales les variables A et B après l'exécution de la suite d'instructions suivantes :

1. A ← 5 ? A = 5
2. B ← A + 4 ? A = 5 ; B = 9
3. A ← A + 1 ? A = 6 ; B = 9
4. B ← A - 4 ? A = 6 ; B = 2

Déroulement d'un algorithme

Instruction	Valeur de A	Valeur de B
0: DEBUT	---	---
1: A ← 5	5	---
2: B ← A + 4	5	9
3: A ← A + 1	6	9
4: B ← A - 4	6	2

A la fin du déroulement de l'algorithme : A = 6 et B = 2

Algorithme de permutation de A et B

Instruction	Valeur de A	Valeur de B	Valeur de C
0: DEBUT	10	20	0
1: C ← A	10	20	10
2: A ← B	20	20	10
3: B ← C	20	10	10

A la fin de l'algorithme : A = 20 ; B = 10 ; C = 10

Instruction de lecture

- Rôle : permet de fournir des données au programme. En pratique, on lit une valeur au clavier ou dans un fichier.
- Syntaxe : lire Variable

  Exemple : *lire Note*
- Effet :
  - A la rencontre de cette instruction, l'ordinateur arrête l'exécution du programme et attend une valeur.
  - On termine la saisie en appuyant sur la touche Entrée.
  - La valeur qu'on tape est affectée à la variable lue.
  - La lecture dans un fichier est automatique.
- Attention : *lire valeur* et *lire expression* n'ont aucun sens.

Instruction d'écriture

- Rôle : permet d'écrire sur un périphérique. En pratique, cela consiste à afficher sur un écran ou à éditer sur une imprimante.
- Syntaxe :

1. écrire Variable

   Exemple : *écrire Note*

2. écrire expression

   Exemple : *écrire « Moyenne = », Moyenne*

3. écrire expression

   Exemple : *écrire « Moyenne= », ((Note1 + Note2) / 2)*

- Attention : écrire Note n'est pas équivalent à :

  écrire « Note »

L’écriture d’un algorithme

- Syntaxe :

Algorithme *Nom_de_l’algorithme*

Déclaration des variables

Début

Suite d’instructions

Fin

Un exemple d'algorithme

Algorithme Addition

variables X, Y : numériques

Début

X ← 4

écrire « Entrez la valeur de Y : »

lire Y

écrire « X + Y = », (X + Y)

Fin

2 variables numériques sont déclarées par cette instruction

4 instructions forment le corps de l'algorithme

Instruction de choix simple

- Rôle : permet d'exécuter une ou plusieurs instructions quand une condition est vérifiée.
- Syntaxe :

si

condition

alors

suite d'instructions

Instruction de choix simple

- Exemple : on veut afficher un message quand X est positif.

si

X > 0

alors

écrire « X est positif. »

Instruction de choix simple

- La condition peut être composée en utilisant des ‘OU’ et des ‘ET’.

Exemple :

si

((X = 3) OU (Y < 4)) ET (Z > 0))

alors

suite d’instructions

Instruction de choix avec alternative

- Rôle : permet de spécifier aussi ce qu'il faut faire dans le cas où la condition n'est pas vérifiée.
- Syntaxe :

  si

  condition

  alors

  suite d'instructions

  sinon

  autre suite d'instructions

Instruction de choix avec alternative

- Exemple : on a saisi une valeur pour X

si

X < 0

alors

écrire « X est négatif. »

sinon

écrire « X n'est pas négatif. »

Instruction de répétition

- Rôle : permet d'accomplir *n* fois un traitement en boucle *(n > 1)*.
- Syntaxe :

x ← 1	Initialisation
accomplir *n* fois	Répétition
suite d'instructions	Traitement
x ← x + 1	Incrémentation
boucler	Exécution

Instruction de répétition

- Exemple : on saisit 2 nombres pour en faire l'addition.

```
Total ← 0
x ← 1
accomplir  2  fois
        écrire « Entrez un nombre : »
lire Nbre
Total ← Total + Nbre
x ← x + 1
boucler
écrire « La somme des deux nombres saisis vaut : », Total
```

Déroulement de l'algorithme

Instruction	Valeur de Nbre	Valeur de Total
Total ← 0	---	0
X ← 1 : lire Nbre	10	0
Total ← Total + Nbre	10	10
X ← 2 : lire Nbre	20	10
Total ← Total + Nbre	20	30

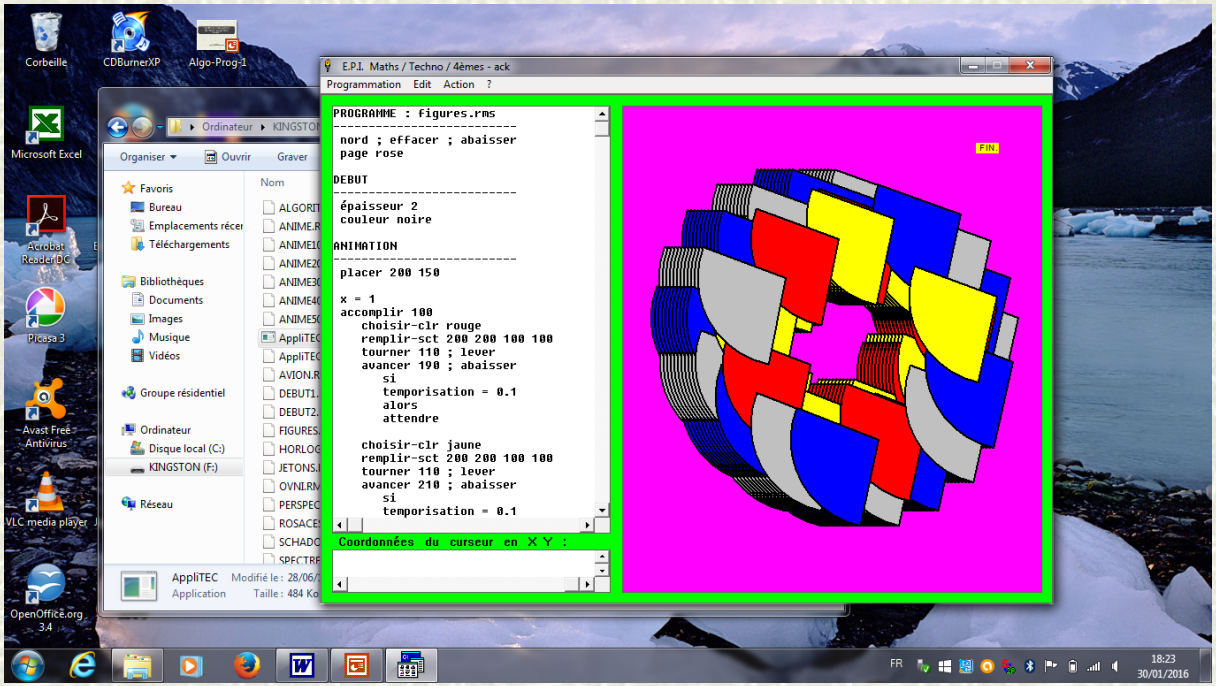
Le logiciel Swing …

- a été conçu pour découvrir, au collège, la programmation d'Applications intégrant Textes Et Commandes graphiques, ou Applied Technology (Technologie Appliquée),
- il dispose d'un Editeur de Liens Dynamiques (D.L.E) qui permet d'optimiser l'espace de stockage (tables et fichiers) ainsi que la vitesse de traitement (compilation et exécution) *[Swing fonctionne avec 2,34 Méga-Octets seulement !]*,
- il utilise un jeu limité d'instructions simple, facile à utiliser mais aussi compatible avec Windows 7 Professionnel © *[Version originale sous Linux Mint Cinnamon 18.3]*

Utiliser le logiciel SWING

G.U.I = Graphical User Interface

Interface Graphique pour l'Utilisateur = Interface Homme-Machine (I.H.M)



Utiliser le logiciel SWING

- Et maintenant, une découverte pratique du logiciel devant l'ordinateur...