


<http://licence-info.univ-lyon1.fr/LIFAMI>

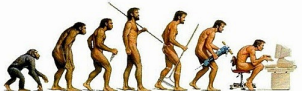


## LIFAMI

### APPLICATIONS EN MATH ET INFO

CULTURE GÉNÉRALE DE PROGRAMMATION

Alexandre Meyer  
Equipe SAARA, Laboratoire LIRIS  
Université Lyon 1



## LIFAMI


### APPLICATIONS EN MATH ET INFO

Culture générale de programmation




## Programmer (ou coder) ?

- Pourquoi faire ? Résoudre des problèmes
- Quel type de problème ?
  - Un ordinateur est meilleur qu'un humain (ordinaire) dans peu de chose
    - Plus grande mémoire (mais moins bien organisée)
    - Tâches répétitives sans se lasser
  - Mais il ne comprend rien au monde
- L'ordinateur est un (bon?) outils pour l'homme
- Finalement c'est quoi coder ?



## Programmer (ou coder) ?

- Finalement c'est quoi coder ?
  - Il faut tout expliquer à un ordinateur par une série de commandes, d'instructions que l'on peut répéter facilement (**boucle**)
  - On peut regrouper des instructions dans des groupes d'instructions=des **fonctions/procédures**
  - Il faut organiser les données, par ex. avec des **structures/tableaux**
- Ceci se fait dans un langage précis
  - Chaque langage aura ses spécificités et sa cible d'applications
- Il faut du temps pour apprendre à coder
  - Etre maniaque, ordonnée, précis
  - Aimer ça (voir ça comme un jeu)



## IEEE : top 10 des meilleurs langages de programmation de l'année 2016

Web Mobile Enterprise Embedded

Language Rank	Types	Spectrum Ranking	Custom Ranking
1. C	Web, Mobile, Enterprise, Embedded	100.0	100.0
2. Java	Web, Enterprise	98.1	99.7
3. Python	Web, Mobile, Enterprise	98.0	99.1
4. C++	Mobile, Enterprise, Embedded	95.9	96.8
5. R	Enterprise, Embedded	87.9	91.0
6. C#	Web, Mobile, Enterprise	86.7	84.1
7. PHP	Web, Mobile, Enterprise	82.8	83.6
8. JavaScript	Web, Mobile	82.2	82.6
9. Ruby	Web, Mobile, Enterprise	74.5	74.8
10. Go	Web, Mobile, Enterprise	71.9	73.3

## IEEE: Top 10 des langages de programmation en forte croissance

Web Mobile Enterprise Embedded

Language Rank	Types	Trending Ranking
1. C	Web, Mobile, Enterprise, Embedded	100.0
2. C++	Mobile, Enterprise, Embedded	97.6
3. Python	Web, Mobile, Enterprise	97.2
4. Java	Web, Mobile, Enterprise	94.4
5. Swift	Mobile, Enterprise	88.9
6. R	Enterprise, Embedded	85.5
7. JavaScript	Web, Mobile	84.7
8. Ruby	Web, Mobile, Enterprise	81.1
9. Go	Web, Mobile, Enterprise	80.6
10. C#	Web, Mobile, Enterprise	78.9

## IEEE : Top 10 des langages les plus demandés par les employeurs

Web Mobile Enterprise Embedded			
Language Rank	Types	Jobs Ranking	
1. C		100.0	
2. Java		98.4	
3. Python		96.9	
4. C++		93.0	
5. JavaScript		88.8	
6. C#		86.4	
7. PHP		81.8	
8. Ruby		79.5	
9. HTML		78.9	
10. Swift		74.9	

## IEEE : Top 10 des langages les plus populaires dans la communauté open source

Web Mobile Enterprise Embedded			
Language Rank	Types	Open Ranking	
1. C++		100.0	
2. Python		98.7	
3. C		97.9	
4. Java		95.9	
5. Swift		91.6	
6. JavaScript		87.6	
7. C#		85.8	
8. Ruby		85.7	
9. PHP		84.8	
10. R		84.7	

## Les langages indispensables d'après

<http://dailygeekshow.com/programmation-internet-langage/>

1. **Le Java** L3 : LIFAP7 et beaucoup en M1
2. **Le C** L1 : LIFAP1, LIFAMI
3. **Le C++** L2 : LIFAP3, LIFAP4; L3 : LIFAP6
4. **L'Objective-C** (propriétaire apple, proche du C++)
5. **Le C#** (propriétaire microsoft, proche du java)
6. **Le PHP** L1 : LIFASR2; L2 : LIFBDW1
7. **Le Python**
8. **Le Ruby**
9. **Le SQL** L3 : LIFBDW2
10. **Le JavaScript** L2 : LIFAP5

## Les langages indispensables d'après

<http://dailygeekshow.com/programmation-internet-langage/>

1. Le Java L3 : LIFAP7 et beaucoup en M1
2. Le C L1 : LIFAP1, LIFAMI
3. Le C++ L2 : LIFAP3, LIFAP4; L3 : LIFAP6
4. L'Objective-C
5. Le C#

Ces 5 langages ont

- une syntaxe assez proche : {}, for, etc.
- sont typés (les variables doivent avoir un type)

## Les langages en Licence Info (Lyon 1)

- Le Java L3 : LIFAP7 et beaucoup en M1
- Le C L1 : LIFAP1, LIFAMI
- Le C++ L2 : LIFAP3, LIFAP4; L3 : LIFAP6
- Le PHP L1 : LIFASR2; L2 : LIFBDW1
- Le SQL L3 : LIFBDW2
- Le JavaScript L2 : LIFAP5
- Shell L1 : LIFASR1
- Scheme L1 : LIFAP2
- Assembleur L1 : LIFASR3
- ProLog L2 : LIFPROLOG (en option)

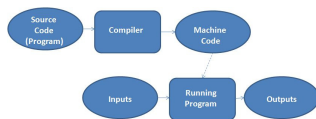
➔ Pas idiot de savoir raisonner en algo (pseudo-langage)

## Les langages en Licence Info (Lyon 1)

- Le C L1 : LIFAP1, LIFAMI
- Assembleur L1 : LIFASR3
- Le C++ L2 : LIFAP3, LIFAP4; L3 : LIFAP6
- ➔ Plutôt proche de la machine (bonne chose pour vous former)
- Shell L1 : LIFASR1 ➔ Spécifique au script
- Scheme L1 : LIFAP2 ➔ Programmation récursive
- Le PHP L1 : LIFASR2; L2 : LIFBDW1 ➔ pour le WEB
- Le SQL L2 : LIFBDW1; L3 : LIFBDW2 ➔ pour les Base de données
- Le JavaScript L2 : LIFAP5 ➔ pour le WEB
- ProLog L2 : LIFPROLOG (en option)
- Le C++ M1
- Le Java L3 : LIFAP7 et beaucoup en M1
- ➔ Plutôt haut niveau

## Compilé ou interprété ?

- **Compilateur** (cas du C/C++)
  - = un programme qui traduit un code source (.cpp pour nous) → code assembleur → exécutable
  - + .exe (sous windows) peut s'exécuter seul
  - .exe compilé pour un processeur-X/OS-Y ne tournera pas ailleurs
  - + programme rapide



## Compilé ou interprété ?



- **Interpréteur** (sera le cas de Scheme, shell, etc.)
  - = un programme qui exécute à la volée un code
  - + code tournera sur toutes les machines disposant de l'interpréteur
  - plus lent

## Paradigme de programmation

Un **paradigme de programmation** est un style fondamental de [programmation informatique](#)

- C'est une notion bien plus importante que les querelles de langages
- Un langage offre souvent plusieurs paradigmes de programmation



## Paradigme de programmation

- **Programmation impérative**
  - Procédurale : LIFAP1, LIFAM1, etc.
  - Des procédures et des fonctions avec des boucles et des structures
- **Programmation déclarative**
  - Fonctionnelle : LIFAP3
    - Uniquement des fonctions, appels récursifs pour les itérations
  - Logique : LIFPROLOG
    - Description du problèmes et non de la recette menant à la solution
- **Programmation orientée objet**
  - En LIFAP3, LIFAP4, LIFAP6, LIFAP7, etc.
  - Les fonctions/procédures sont regroupées dans les structures
- **Programmation évènementielle** : LIFAP4, LIFIHM, etc.

## Savoir débiter quand on débute

- Difficile quand on commence, voici quelques idées
- Il faut savoir ce qu'il y a dans les variables et par où on passe
    - faire des cout de debug
  - **Programmation incrémentale**
    - **Ecrire une fonction/procédure à la fois**
    - **Faire un main minimal**
- tester avec ce qu'on a, faire plusieurs tests  
Puis seulement passer à la suite si tout semble ok



Bug

= un insecte

1<sup>er</sup> plantage informatique était dû à un insecte coincé dans l'ordinateur

## Savoir débiter quand on débute

Si on a déjà écrit beaucoup de code sans avoir testé

- **C'est mal, très mal !!!**
  - Revient à trouver une aiguille dans une botte de foin
- Mettre en commentaire tout le code sauf une fonction et lancer le code comme ça, pour voir si ok ou non
- Décommenter la 2<sup>e</sup> fonction, seulement quand on est sûr que la 1<sup>ere</sup> fonction fonctionne
- Etc.

