

I. Informatique et ses fondements

3ème partie : Programmation

I.3 Programmation

1. Le langage machine
2. Langage de plus haut niveau et compilation
3. Bugs
4. Des humains et des langages

I.3 Programmation

1. **Le langage machine**
2. Langage de plus haut niveau et compilation
3. Bugs
4. Des humains et des langages

L'ordinateur est...

Gérard Berry a dit :

« **L'ordinateur est complètement con** »

L'ordinateur/le smartphone ne sont pas des objets magiques.

Ce sont des **dispositifs électroniques** très simples (mais très rapides).

Cf <http://rue89.nouvelobs.com/2016/08/26/>

[gerard-berry-lordinateur-est-completement-con-257428](http://rue89.nouvelobs.com/2016/08/26/gerard-berry-lordinateur-est-completement-con-257428)

Le langage machine

L'ordinateur étant con, il ne sait qu'obéir à des **ordres très simples** (des tâches, des instructions) qui dépendent du modèle de processeur : l'**assembleur**.

Ces instructions sont limitées :

- déplacer de la mémoire globale vers la mémoire locale
- comparer deux valeurs
- ajouter deux valeurs
- sauter à un autre point de programme
- ...

Le langage machine : exemple

```
movl    $0, -8(%rbp)
movl    $7, -4(%rbp)
cmpl    $0, -4(%rbp)
jle     .L2
movl    -4(%rbp), %eax
addl    %eax, -8(%rbp)
```

.L2:

```
movl    -8(%rbp), %eax
```

⇒ on n'a pas envie de l'écrire directement !

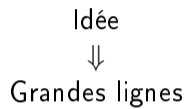
I.3 Programmation

1. Le langage machine
2. **Langage de plus haut niveau et compilation**
3. Bugs
4. Des humains et des langages

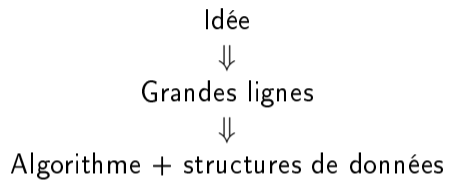
Compilation ?

Idée

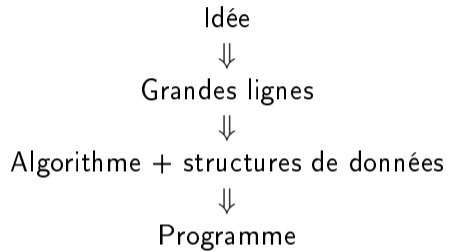
Compilation ?



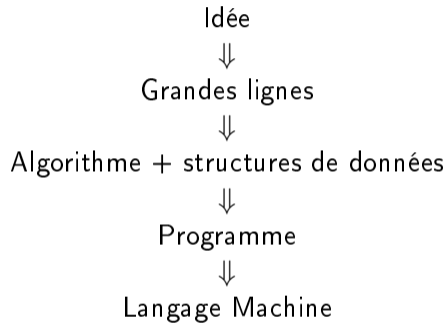
Compilation ?



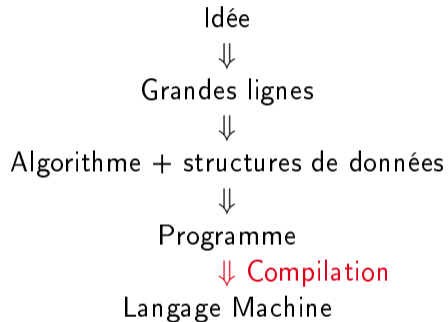
Compilation ?



Compilation ?



Compilation ?



Compilateur ?

Un compilateur est un programme !

En première approximation,

- il prend du texte dans un langage précis (C, C++, OCaml...)
- il renvoie un exécutable ou un texte en assembleur.

On a aussi des langages interprétés où le programme est exécuté dynamiquement par l'interprète.

Compilateur++

En plus, le compilateur

- vérifie qu'il n'y a pas d'erreur de syntaxe
- vérifie qu'il n'y a pas d'erreur de type (*dépend du langage*)
- peut vérifier un certain nombre de critères simples et émettre des avertissements (comme une variable non utilisée).

Bref, il vérifie (au moins) que le programme peut être transformé en un exécutable.

Compilateur++

En plus, le compilateur

- vérifie qu'il n'y a pas d'erreur de syntaxe
- vérifie qu'il n'y a pas d'erreur de type (*dépend du langage*)
- peut vérifier un certain nombre de critères simples et émettre des avertissements (comme une variable non utilisée).

Bref, il vérifie (au moins) que le programme peut être transformé en un exécutable.

Mais ça ne garantit pas que le programme fait ce qu'on veut !

I.3 Programmation

1. Le langage machine
2. Langage de plus haut niveau et compilation
3. **Bugs**
4. Des humains et des langages

USS Yorktown



En 1996, l'USS Yorktown teste le programme *Navy's Smart Ship*.

Un membre d'équipage rentre un zéro comme valeur lors de manoeuvres.

⇒ une division par zéro s'ensuit

⇒ bloquant tout le navire !

Cf [https://en.wikipedia.org/wiki/USS_Yorktown_\(CG-48\)](https://en.wikipedia.org/wiki/USS_Yorktown_(CG-48))

On teste les valeurs d'entrée !

Ariane 5



Premier vol d'Ariane 5 en 1996 :

- Ils avaient récupéré directement du code d'Ariane 4.
- Mais les valeurs d'accélération de la fusée dépassent les valeurs max prévues !

⇒ dépassement de capacité

⇒ auto-destruction

Cf https://fr.wikipedia.org/wiki/Vol_501_d%27Ariane_5

On tient compte des valeurs maximales pour éviter les dépassements de capacité (des valeurs ou des tailles de tableaux)

C'est quoi un bug ?

C'est quand l'utilisateur n'a pas ce qu'il veut !

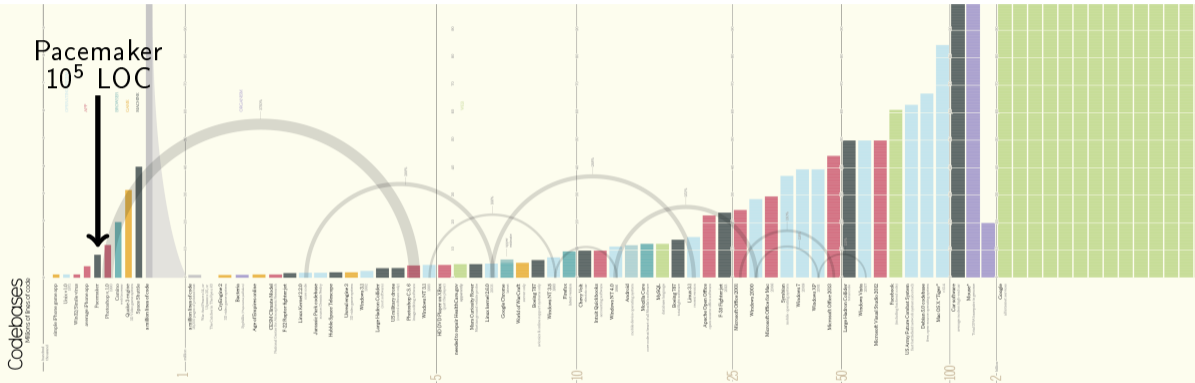
C'est quoi un bug ?

C'est quand l'utilisateur n'a pas ce qu'il veut !

- à cause d'une erreur de programmation
 - oubli d'un cas
 - typo
 - dépassement de capacité
 - accès illicite à la mémoire
 - ...
- à cause d'une erreur de communication
 - entre le programmeur et l'utilisateur
 - entre le programmeur et le programmeur !

Taille du code

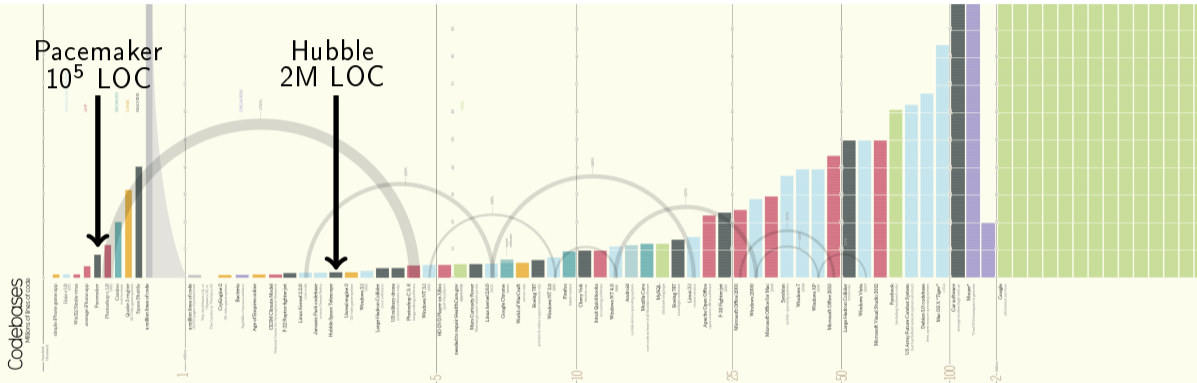
Pacemaker
 10^5 LOC



Cf <http://www.informationisbeautiful.net/visualizations/million-lines-of-code/>

[//www.informationisbeautiful.net/visualizations/million-lines-of-code/](http://www.informationisbeautiful.net/visualizations/million-lines-of-code/)

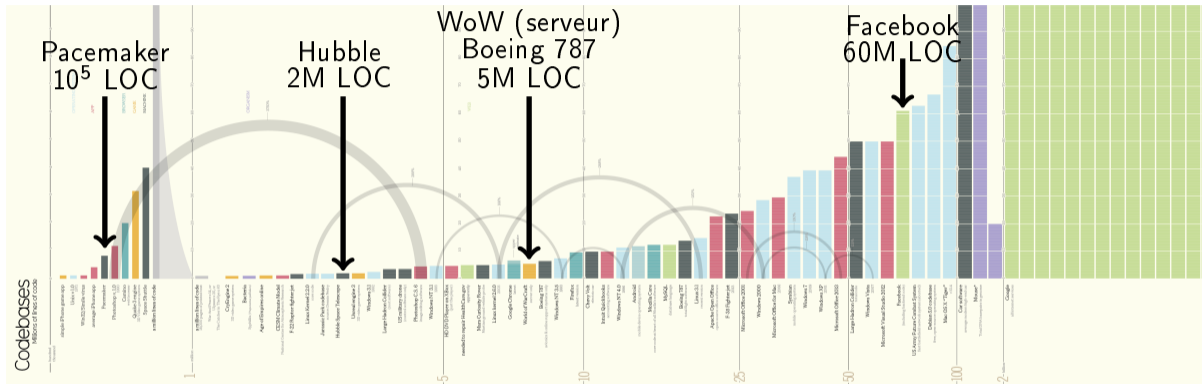
Taille du code



Cf <http://www.informationisbeautiful.net/visualizations/million-lines-of-code/>

[//www.informationisbeautiful.net/visualizations/million-lines-of-code/](http://www.informationisbeautiful.net/visualizations/million-lines-of-code/)

Taille du code



Cf <http://www.informationisbeautiful.net/visualizations/million-lines-of-code/>

[//www.informationisbeautiful.net/visualizations/million-lines-of-code/](http://www.informationisbeautiful.net/visualizations/million-lines-of-code/)

Illustrations

- Ariane 501 Cluster CC-BY-SA-3.0 Self-published work
https://fr.wikipedia.org/wiki/Vol_501_d%27Ariane_5#/media/File:Ariane_501_Cluster.svg
- Codebases concept & design : David McCandless <http://www.informationisbeautiful.net/visualizations/million-lines-of-code/>

I.3 Programmation

1. Le langage machine
2. Langage de plus haut niveau et compilation
3. Bugs
4. **Des humains et des langages**

Plusieurs langages ?

Il y a de très très nombreux langages de programmation !

- Ils ont des caractéristiques différentes.
- Mais on peut toujours tout faire !
- (mais pas forcément aussi facilement)

Paradigmes

On peut classer les langages suivant leur(s) paradigmes

- impératif (comme Fortran ou C)
- fonctionnel (comme Haskell)
- objet (comme Java)
- par contraintes (comme Prolog)
- événementiel (pour les interfaces graphiques ou les robots)
- ...

Beaucoup de langages sont multi-paradigmes ! (C++, OCaml, Python, Java...)

Exemple

On veut augmenter de 1 un ensemble de valeurs entières.

En C, on choisirait un `tableau` (`int *`) et on le parcourt.

```
int i;  
for (i=0; i < N; i++) {  
    tab[i] = tab[i] + 1;  
}
```

En OCaml, on choisirait une `liste` (`int list`) et on applique un `map`.

```
List.map (fun x => x+1) l;;
```

Choix

Choisir le langage le plus adapté dépend de ce qu'on veut programmer !

- paradigme ?
- interface avec d'autres développements ?
- bibliothèque de support ?
- connaissance du langage ?
- aide, tutorat ?

Au final, celui/celle qui programme choisit le langage qui lui plaît !