5IPRO

Principes algorithmiques et programmation

Benjamin Delbar

Benjamin Delbar

Cours 12

Feedback interro

Introduction aux différents langages de programmation

Feedback interro

Moyenne générale 14.5/20 (17.5 sans les échecs) => Bravo!
Théorie : 8/10
Pratique : 14/20

5 résultat de 20(+)/20
6 échecs malheureusement

Feedback interro

Points d'attention

Attention à la **lecture des énoncés**

Q3
Inverser le tableau => ne veut pas dire seulement faire un console.log
Inverser le tableau => ne veut pas dire trier le tableau

Q4
Remplacer dans le tableau => ne veut pas dire seulement faire un console.log
Oubli de l'index 15 => Affichez les 2 messages (multiple de 3 et de 5)

Feedback interro

Focus sur l’entraide

Plus que nécessaire pour mener à bien votre bachelier

Déjà présente (et c’est génial)

Améliorer cela pour aider ceux qui ont plus de mal

Pas de honte à demander de l’aide

Très bon exercice pour ceux qui sont plus à l’aise => expliquer la matière force à la maîtriser d’abord

Benjamin Delbar

Introduction aux différents langages de programmation

Concepts importants

- Turing-complete
- Interprété - Compilé - JIT
- Bas niveau - Haut niveau

Fiches explicatives

Exercices

Concepts importants

Turing-complete

Alan Turing, 1936

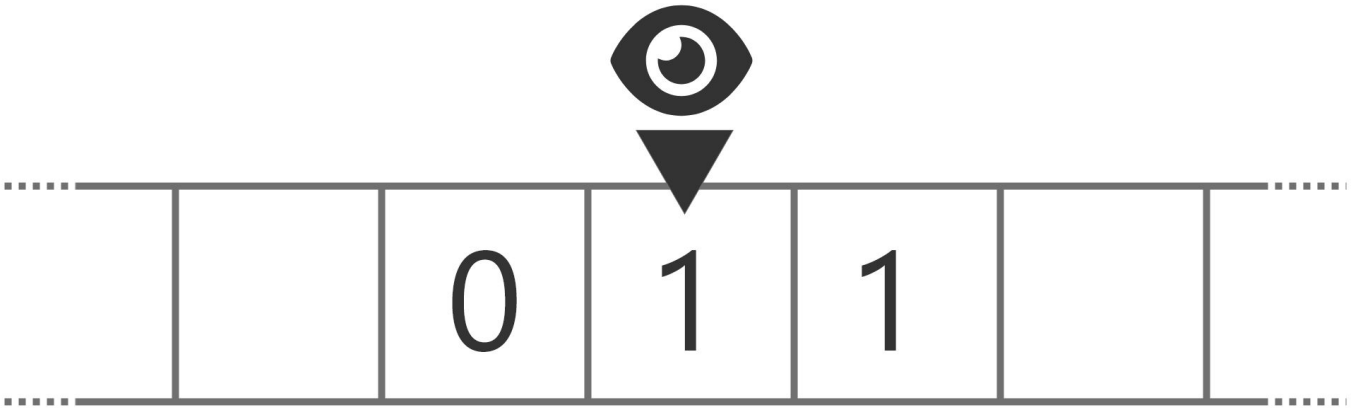
La majorité des langages de programmation sont “Turing-complete”, càd qu’ils sont capable de faire tout ce qu’une machine de Turing sait faire…

Qu’est ce qu’une machine de Turing du coup?

Visualiser le comme une bande de ruban aussi longue que nécessaire, avec une tête capable de **lire et écrire** sur ce même ruban
=> Rien que de pouvoir écrire et lire des 0 et des 1 par ex. est suffisant pour calculer a peu près n’importe quoi

Concepts importants

Visualisation du “ruban” avec la tête de lecture/écriture



Concepts importants

Machine de Turing (TM)

Conditions (conditional branching)
Si ce que je regarde sur mon ruban est un 0 je fais ceci, si c'est un 1 je fais cela à la place. => *If, else*
=> Go to : Permettre de se déplacer et d'aller (go to) un autre endroit du ruban en fonction d'une condition

Une quantité suffisante de mémoire
Notre RAM, il nous faut suffisamment de "ruban" pour pouvoir traiter le problème. Le modèle de la TM en dispose d'une infinité, ce qui n'est pas possible avec nos ordinateurs actuels

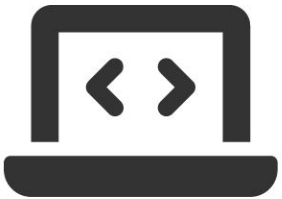
Concepts importants

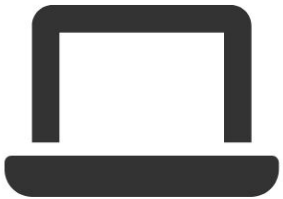
Interprété - Compilé - Just in Time (JIT)

Problème de base : Le code source doit pouvoir être utilisé par une machine

3 approches différentes

Concepts importants - Compilé





Concepts importants - Compilé

Concepts importants - Compilé

Concepts importants

Compilé

Transformation d’un code source en un code machine (exécutable), par un compilateur

Le code source n’est pas transmis, uniquement le code compilé

Concepts importants - Interprété

Concepts importants - Interprété

Concepts importants - Interprété

Concepts importants - Pour & Contre

Compilé

+++

Prêt à être exécuté
Souvent plus rapide
Code source reste privé

Compilé pour un système (.exe => ! Mac)
Peu flexible (x32 / x64)
Compilation nécessaire

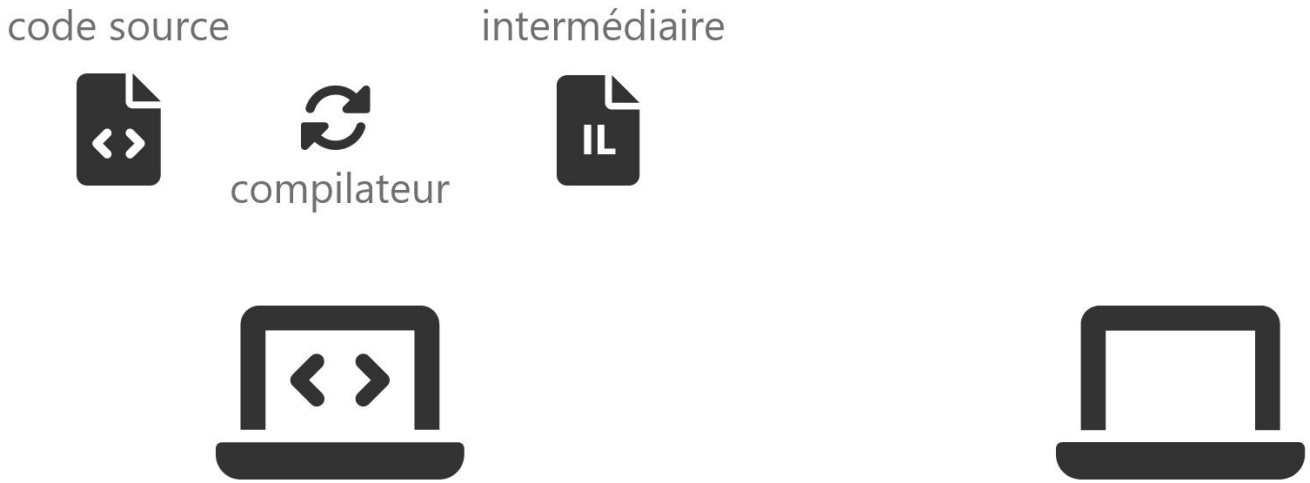
Interprété

+++

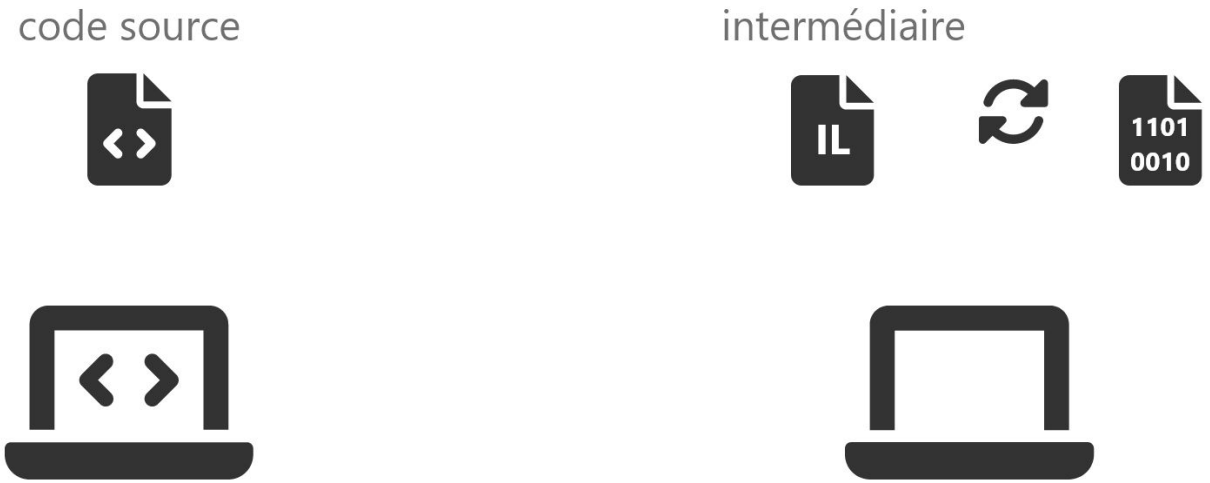
Cross platform
Facile à tester
Facile à débug

Nécessite un interpréteur
Souvent plus lent
Code source fourni

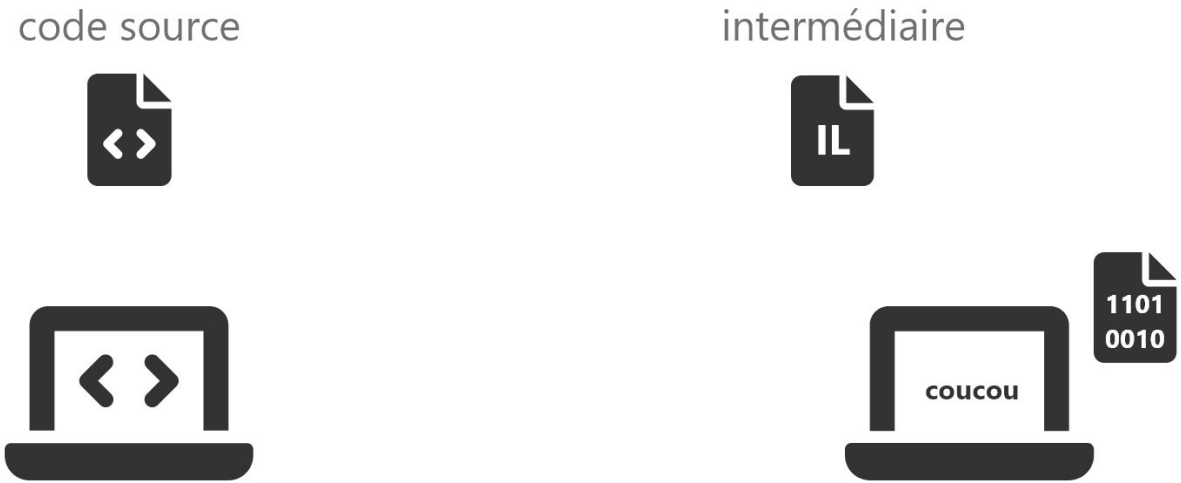
Concepts importants - JIT



Concepts importants - JIT



Concepts importants - JIT



Concepts importants

Just in time (JIT)

Transformation d’un code source en un “bytecode”, par un compilateur

Bytecode adapté en code machine par le receveur

Concepts importants

Bas niveau - Haut niveau
cf Image

Quantité d'abstraction entre le langage de programmation et le langage machine

Bas niveau :
Au plus un langage est bas niveau au plus il est proche du hardware et contrôle plus de choses (allocation mémoire etc)

Plus dur à utiliser et plus fastidieux pour développer
=> Créer des firmwares / codes embarqués / Operating systems

Concepts importants

Bas niveau - Haut niveau

Haut niveau :
Plus grande abstraction => Plus facile à lire, écrire et à comprendre

Les programmeurs ne doivent pas s’occuper de certaines choses,

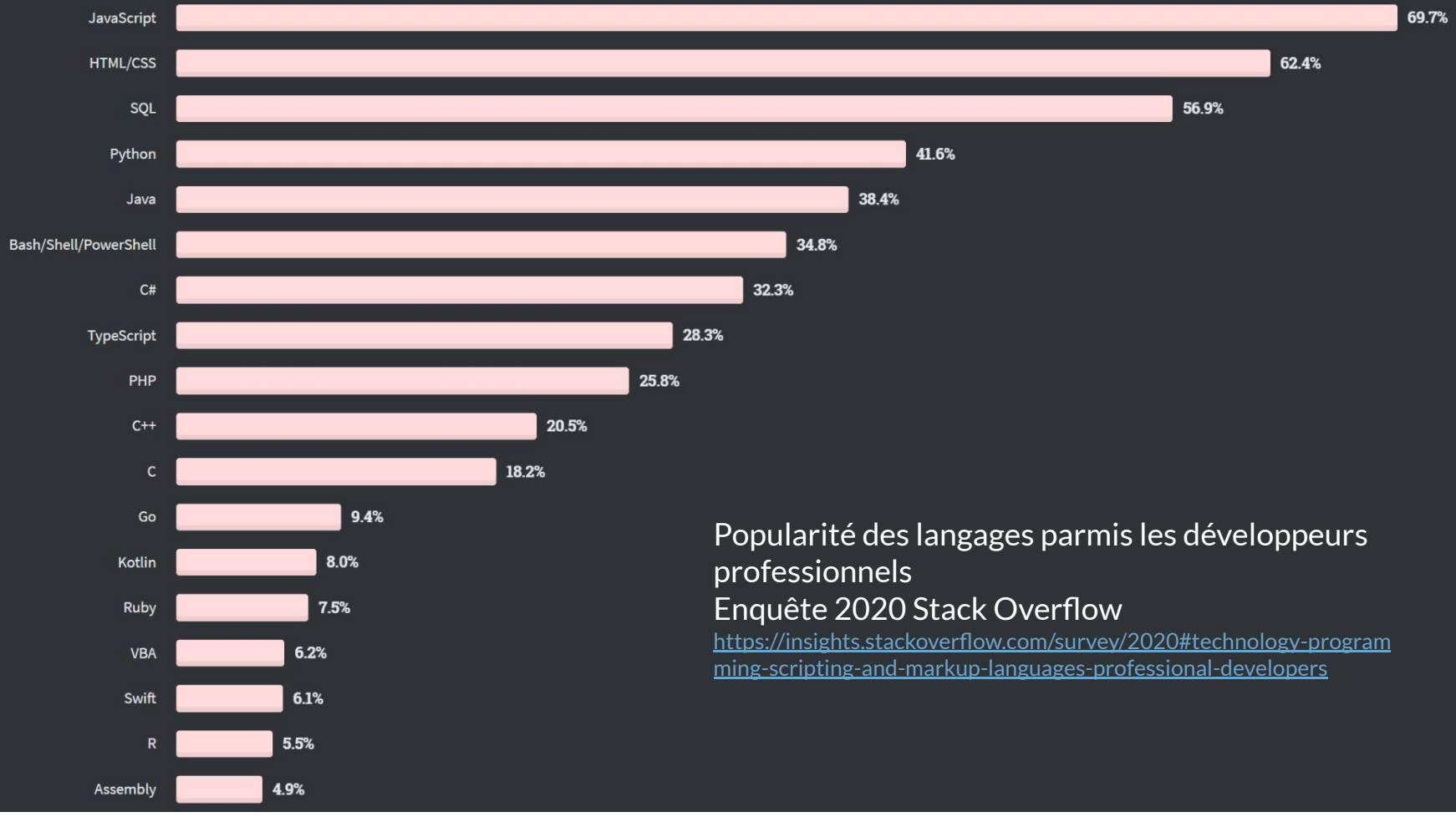
Le code est moins dépendant du hardware

Concepts importants

Bas niveau - Haut niveau

Un langage se situe sur ce spectre et non pas de manière binaire

C’est une question de comparaison entre langages aussi



Popularité des langages parmis les développeurs professionnels
Enquête 2020 Stack Overflow
https://insights.stackoverflow.com/survey/2020#technology-programming-scripting-and-markup-languages-professional-developers

Fiches explicatives

Javascript

Création	1995
Créateur	Brendan Eich
Turing complete	Oui
Niveau d'abstraction	Élevé (Haut niveau)
Type	Langage interprété (script)
Utilisation principale	Web, côté client, côté serveur (nodeJS), applications (electron), ...
Popularité	Très élevée #1
Version actuelle	ECMAScript 2021 (Juin 2021)

Fiches explicatives

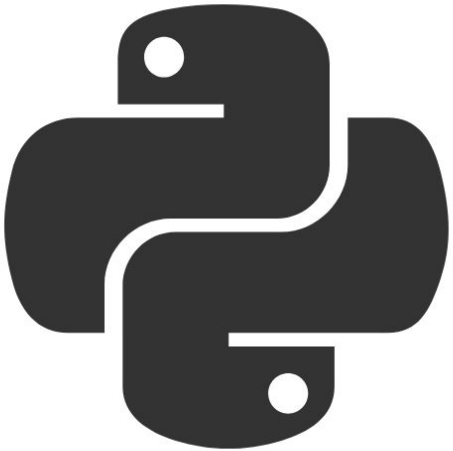
SQL (Structured Query Language)

Création	1974
Créateur	Donald D. Chamberlin
Turing complete	Oui
Niveau d'abstraction	Élevé (Haut niveau)
Type	Langage interprété (mostly)
Utilisation principale	Base de données
Popularité	Très élevée #3

Fiches explicatives

Python

Création	1991
Créateur	Guido van Rossum
Turing complete	Oui
Niveau d'abstraction	Élevé (Haut niveau)
Type	Langage interprété (mostly)
Utilisation principale	Raspberry PI, IoT, web (Django), ...
Popularité	Très élevée #4
Version	3.10 (Oct 2021)



Fiches explicatives

Java

Création	1995
Créateur	James Gosling, Patrick Naughton
Turing complete	Oui
Niveau d'abstraction	Élevé (Haut niveau)
Type	JIT
Utilisation principale	Software d'entreprise, back-end, ...
Popularité	Très élevée #5
Version	Java 17 (2021)

Fiches explicatives

PHP

Création	1994
Créateur	Rasmus Lerdorf
Turing complete	Oui
Niveau d'abstraction	Élevé (Haut niveau)
Type	Interprété
Utilisation principale	Back end (web)
Popularité	Élevée #9
Version	8.1.3 (Fev 2022)

Fiches explicatives

C++

Création	1985
Créateur	Bjarne Stroustrup
Turing complete	Oui
Niveau d'abstraction	Moyen / Bas
Type	Compilé
Utilisation principale	Software, Jeux, OS
Popularité	Moyenne #10
Version	C++ 20 (Déc 2020)

Fiches explicatives

C

Création	1972
Créateur	Dennis Ritchie
Turing complete	Oui
Niveau d’abstraction	Bas
Type	Compilé
Utilisation principale	OS, code bas niveau, gestion hardware, embed
Popularité	Moyenne #11
Version	C17 (2017)

Fiches explicatives

Assembly

Création	1949
Créateur	David J. Wheeler
Turing complete	Oui
Niveau d’abstraction	Bas niveau
Type	Langage machine
Utilisation principale	Manipulation hardware
Popularité	Basse #18

Hello world!

Javascript (.js)

```
console.log('hello world!');
```

SQL (.sql)

```
SELECT 'hello world!' FROM dual
```

Python (.py)

```
print('hello world!')
```

Java (.java)

```
public class Main {
    public static void main(String[] args) {
        System.out.println('hello world!');
    }
}
```

PHP (.php)

```
<?php
echo "hello world!";
?>
```

Hello world!

C++ (.cpp)

```
#include <iostream>
int main () {
    std::cout << “hello world!”;
    return 0;
}
```

C (.c)

```
#include <stdio.h>
void main () {
    printf(“hello world!”);
    return 0;
}
```

Assembly (.asm)

Benjamin Delbar

Hello world!

Assembly

```
    .equ    LAST_RAM_WORD,   0x007FFFFC
    .equ    JTAG_UART_BASE,  0x10001000
    .equ    DATA_OFFSET,     0
    .equ    STATUS_OFFSET,   4
    .equ    WSPACE_MASK,     0xFFFF

    .text
    .global _start
    .org    0x00000000

_start:
    movia       sp, LAST_RAM_WORD
    movi        r2, '\n'
    call        PrintChar
    movia       r2, MSG
    call        PrintString
_end:
    br          _end

PrintChar:
    subi        sp, sp, 8
    stw         r3, 4(sp)
    stw         r4, 0(sp)
    movia       r3, JTAG_UART_BASE
pc_loop:
    ldwio       r4, STATUS_OFFSET(r3)
    andhi       r4, r4, WSPACE_MASK
    beq         r4, r0, pc_loop
    stwio       r2, DATA_OFFSET(r3)
    ldw         r3, 4(sp)
    ldw         r4, 0(sp)
    addi        sp, sp, 8
    ret

PrintString:
    subi        sp, sp, 12
    stw         ra, 8(sp)
    stw         r3, 4(sp)
    stw         r2, 0(sp)
    mov         r3, r2
ps_loop:
    ldb         r2, 0(r3)
    beq         r2, r0, end_ps_loop
    call        PrintChar
    addi        r3, r3, 1
    br          ps_loop
end_ps_loop:
    ldw         ra, 8(sp)
    ldw         r3, 4(sp)
    ldw         r2, 0(sp)
    addi        sp, sp, 12
    ret

        .org        0x1000
MSG:    .asciz      "Hello World\n"
        .end
```

Autres exemples

Comparaison syntaxique entre C et Python

```c
#include <stdio.h>

int main()
{
    int A = 12;
    int B = 24;
    if (A > B) {
       printf("A est plus grand : %d", A);
    } else if (B > A) {
        printf("B est plus grand : %d", B);
    } else {
        printf("B et A sont égaux : %d", A);
    }
}
```

```python
a = 12
b = 24

if a > b:
    print("A est plus grand :", a)
elif b > a:
    print("B est plus grand :", b)
else:
    print("A et B sont égaux : ", a)
```

Exercices

Par groupe de 2 => en pair programming

dans un maximum de langages vus (**au moins 2)** :

Réalisez l'exercice d'affichage de 1 à 100 (coucou % 3, hibou %5)
Réalisez l'exercice PP / PG (plus petit, plus grand)
Réalisez un tri à bulle / par insertion / par sélection (au choix)
Réalisez le jeu de Nim (bâtonnets)

Exercices

Compilateurs / Interpréteurs en ligne

Python	https://www.programiz.com/python-programming/online-compiler/
C	https://www.onlinegdb.com/online_c_compiler
Java	https://www.jdoodle.com/online-java-compiler/
C++	https://www.programiz.com/cpp-programming/online-compiler/
PHP	https://www.w3schools.com/php/phptryit.asp?filename=tryphp_compiler

Exercices

Quels conclusions tirez vous de vos différents codes?