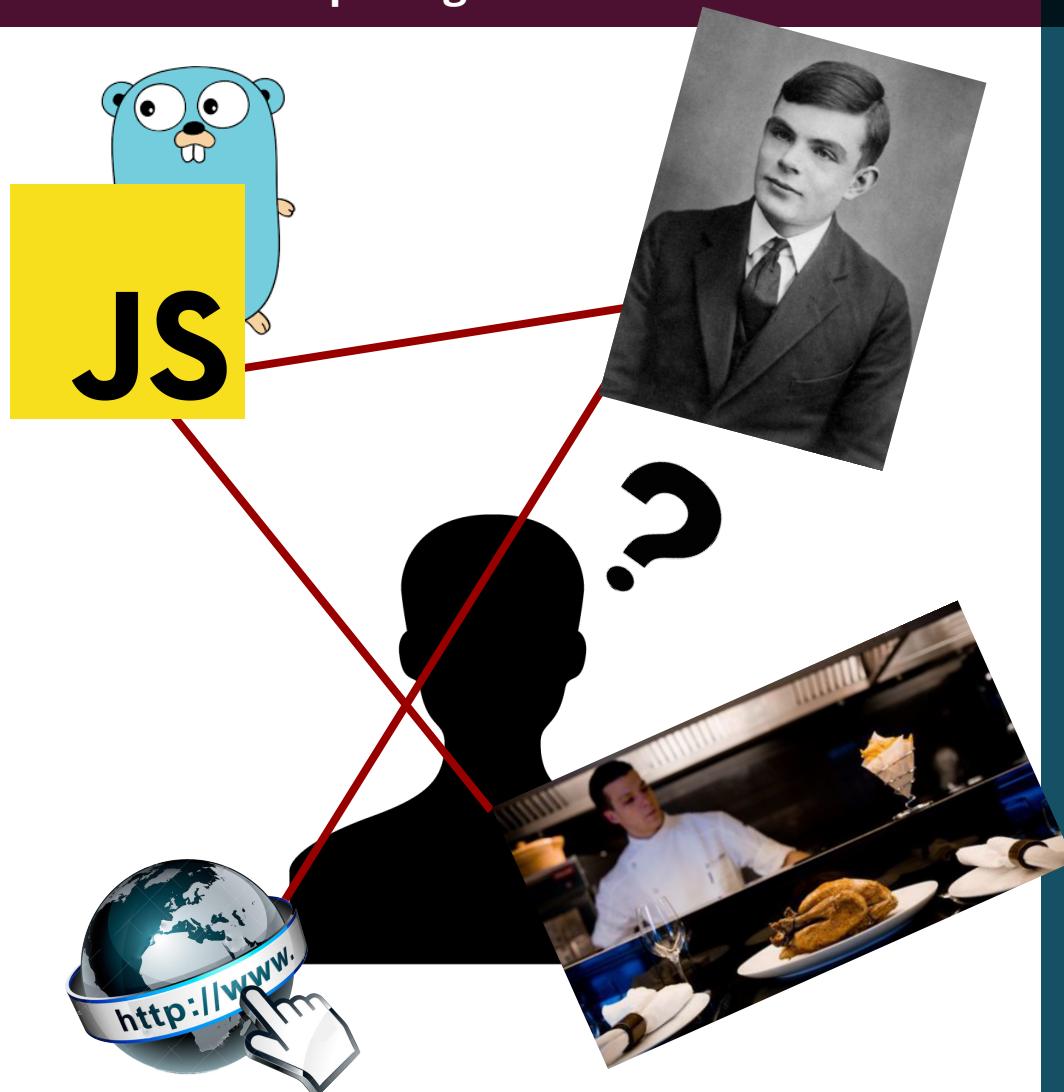


A long long time ago in a  
galaxy far, far away...

Mission révision, correction et découverte

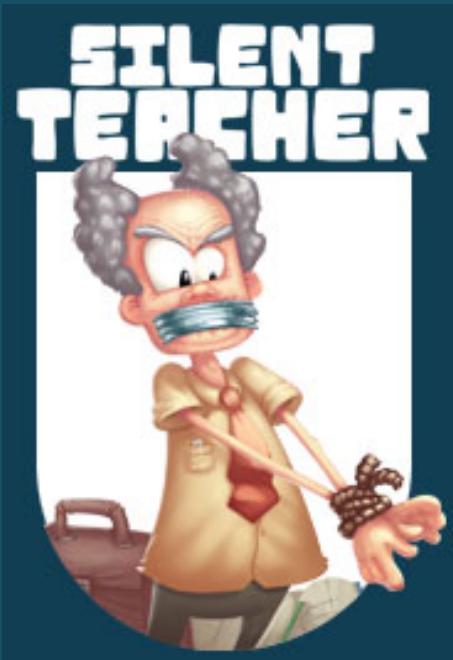
# Episode II - Jenilee la nouvelle padawan

Mission partage de connaissances



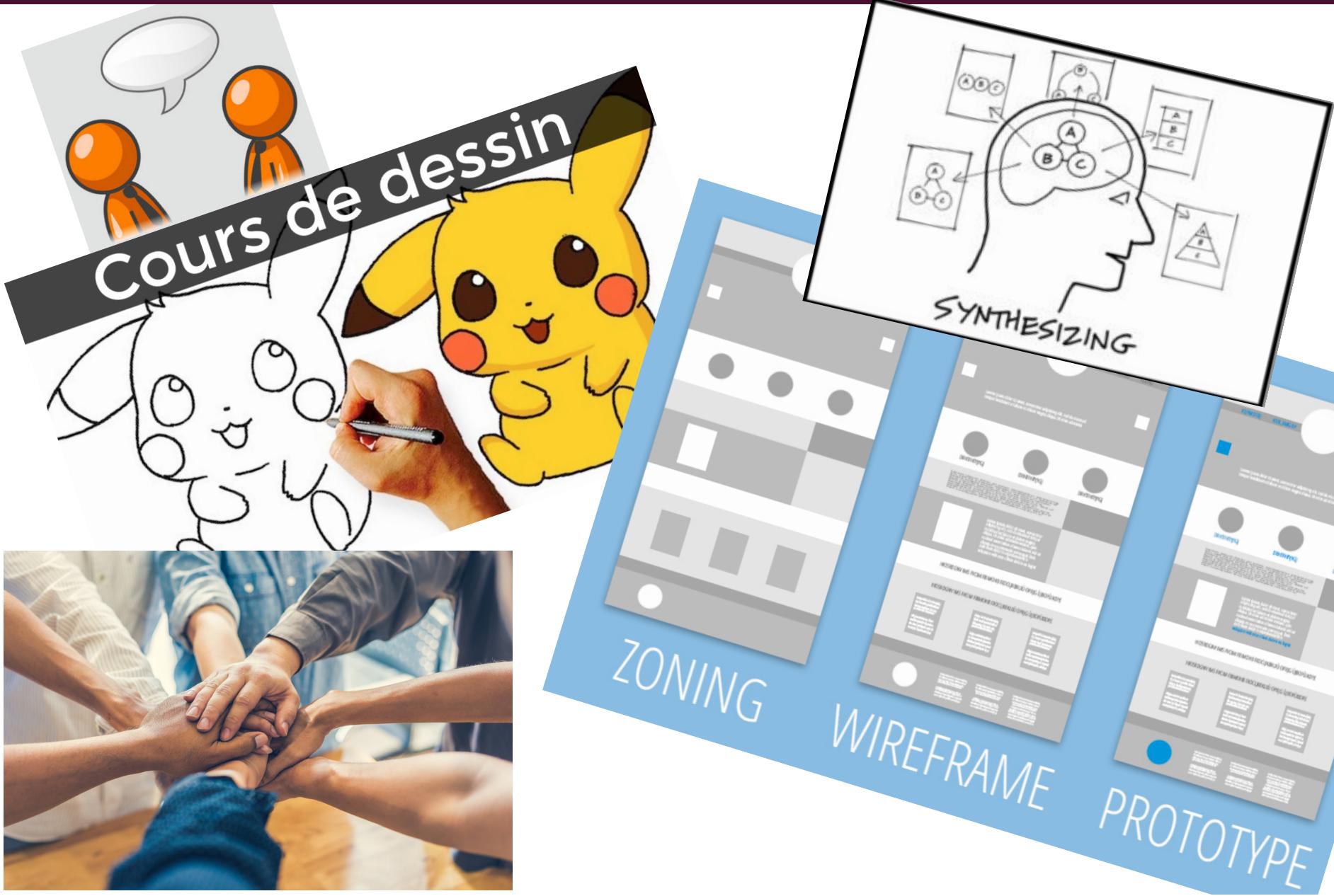
Mission Algo

Jenilee puis correction

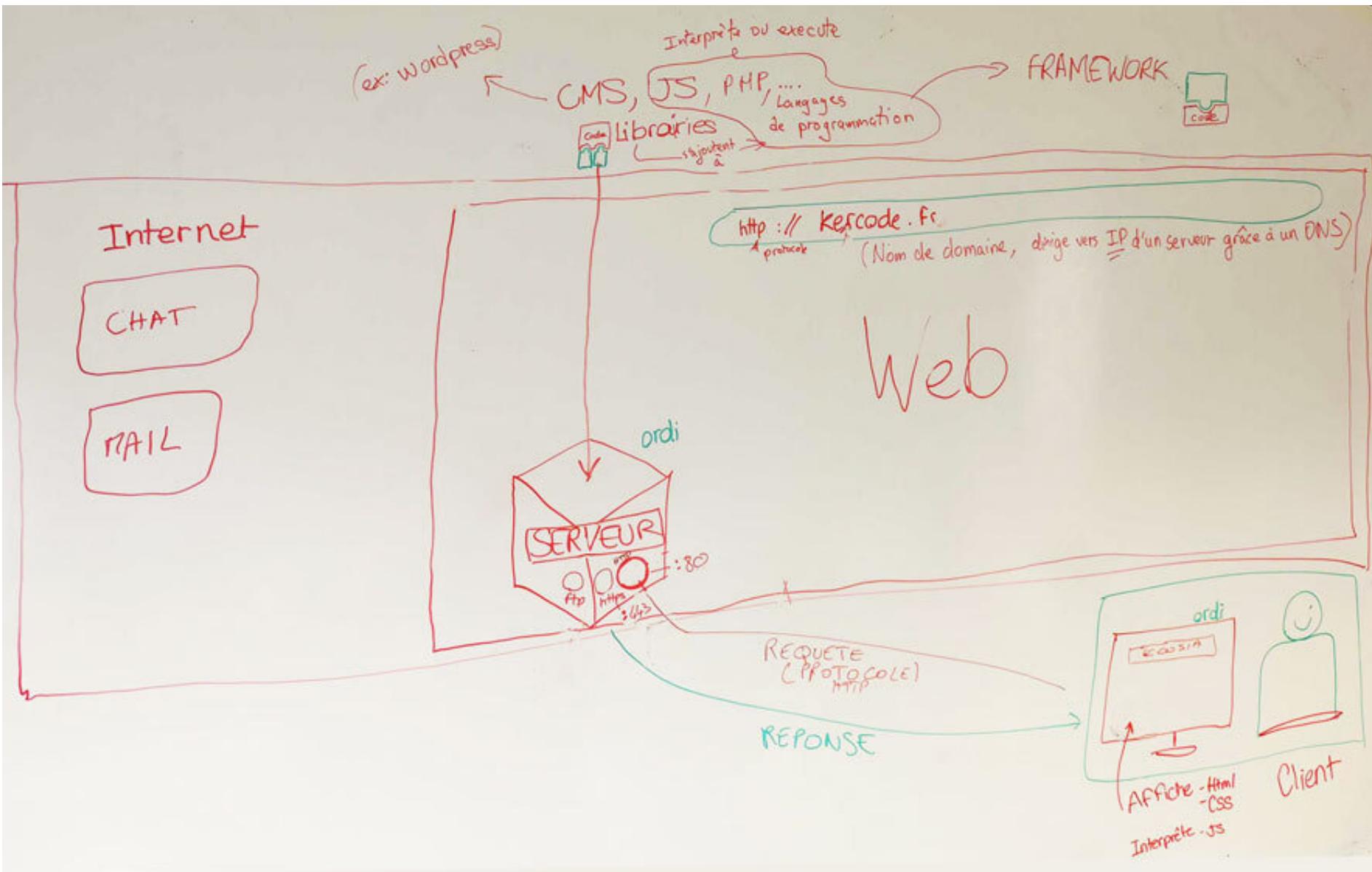


<http://silentteacher.toxicode.fr/>

# Mission partage de connaissances



# Mission partage de connaissances



# Correction Silent Teacher - Basics

5 + 6

6

12

11

5

?

`var a = 1`

`a + 4`

1

9

3

8

6

5

4

2

7

?

`var a = 'wf'`

`var b = '67'`

`a + b`

68

66

67

'wf67'

'f'

'wf'

'w'

?

`var a = 5`

`var b = 3`

`a + b`

13

17

3

5

6

16

10

2

7

14

8

15

9

?

# Correction Silent Teacher - Basics

5 + 6

6

12

11

5

?

`var a = 1`

`a + 4`

1

9

3

8

6

5

4

2

7

?

`var a = 'wf'`

`var b = '67'`

`a + b`

68

66

67

'wf67'

'f'

'wf'

'w'

?

`var a = 5`

`var b = 3`

`a + b`

13

17

3

5

6

16

10

2

7

14

8

15

9

?

## Les variables



Une variable est une boîte qui nous permet de stocker et de restituer de l'information

```
1 var maBoite = "un ordinateur pour coder";
```

On accède à une variable en écrivant son nom

```
1 maBoite  
2 // Retournera "un ordinateur pour coder"
```

### Information présente dans la boîte :

- D'un **type** (entier, texte, tableau, etc)
- D'une valeur

# Correction Silent Teacher - Basics

## Opérateurs +, =, -, \* et /

```
1 // Addition  
2 var a = 0;  
3 var b = 0;  
4  
5 a + b;  
6  
7 // retourne 0  
8  
9  
10 // Concaténation  
11 var c = "La tête ";  
12 var d = "à toto";  
13  
14 c + d;  
15  
16 // retourne La tête à toto
```

L'opérateur **+** est dit d'**addition** lorsqu'on le positionne entre deux nombres

L'opérateur **+** est dit de **concaténation** lorsqu'il un des deux n'est pas un nombre

L'opérateur **=** est dit d'**assiguation**



= "Chaussures"

```
1 var maBoite = "Chaussures";
```

Enfin les opérateurs **-** (**soustraction**), **\*** (**multiplication**) et **/** (**division**) permettent de réaliser les opérations les plus courantes

# Correction Silent Teacher - Plus d'opérateurs

```
var a = 23  
a === 4
```

true    '==='    false    19    4    23    27

?

15 !== 15

30    15    false    '==='    0    true    ?

21 < 14

14    7    35    21    '==='    false    true

?

# Correction Silent Teacher - Plus d'opérateurs

```
var a = 23  
a === 4
```

The Silent Teacher interface displays a question in pink text at the top, followed by a row of seven answer options in light blue boxes. The fourth option, 'false', is highlighted in yellow. Below this row is a light blue input field containing a question mark.

var a = 23  
a === 4

true    '==='    **false**    19    4    23    27

?

15 !== 15

The second question is displayed in pink text above a row of seven answer options. The third option, 'false', is highlighted in yellow. Below this row is a light blue input field containing a question mark.

15 !== 15

30    15    **false**    '==='    0    true    ?

21 < 14

The third question is displayed in pink text above a row of seven answer options. The sixth option, 'false', is highlighted in yellow. Below this row is a light blue input field containing a question mark.

21 < 14

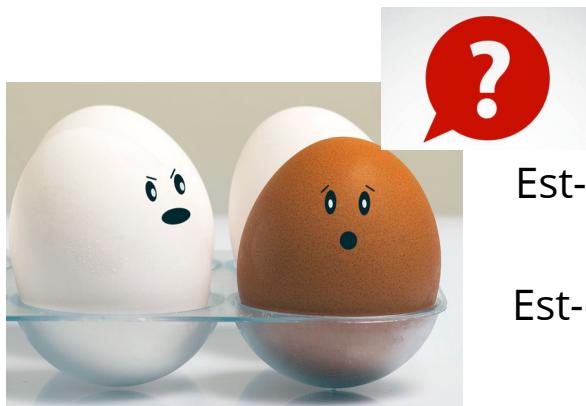
14    7    35    21    '==='    **false**    true

?

# Correction Silent Teacher - Plus d'opérateurs

## Opérateurs ==, !=, <, >

Faisons parler des oeufs

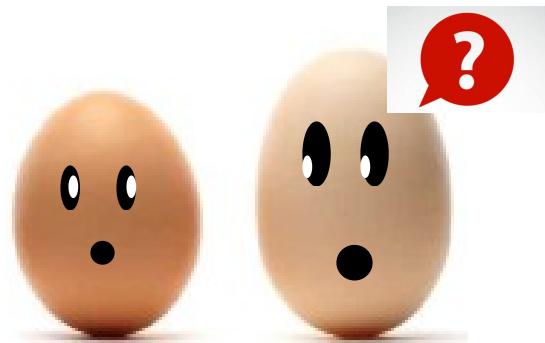


Est-on de la **même** couleur ? `color_egg_a === color_egg_b`

Est-on d'une couleur **différente** ? `color_egg_a != color_egg_b`

OG est-il **plus grand que** OD ? `left_egg > right_egg`

OG est-il **plus petit que** OD ? `left_egg < right_egg`



*La réponse à ces questions est nécessairement oui **true** ou non **false** qui sont des valeurs de type Booléen **Boolean** !*

# Correction Silent Teacher - Les conditions

```
if (3 < 5) {  
    var a = 1  
} else {  
    var a = 3  
}  
a + 2
```

2    4    6    5    3    1    ?

```
if (7 < 15) {  
    var a = 5  
} else if (5 < 4) {  
    var a = 7  
} else {  
    var a = 4  
}  
a + 1
```

7    1    4    6    5    8    ?

```
if (5 === 5) {  
    var a = 4  
} else {  
    var a = 1  
}  
a + 2
```

1    3    7    5    2    4    6  
?

# Correction Silent Teacher - Les conditions

```
if (3 < 5) {  
    var a = 1  
} else {  
    var a = 3  
}  
a + 2
```

-

2    4    6    5    3    1    ?

```
if (7 < 15) {  
    var a = 5  
} else if (5 < 4) {  
    var a = 7  
} else {  
    var a = 4  
}  
a + 1
```

7    1    4    6    5    8    ?

```
if (5 === 5) {  
    var a = 4  
} else {  
    var a = 1  
}  
a + 2
```

1    3    7    5    2    4    6  
? ?

# Correction Silent Teacher - Les conditions

```
1 var jaiFaim = true;  
2  
3 if (jaiFaim) {  
4     return "Miam, je mange";  
5 } else {  
6     return "Je bois un café";  
7 }
```

On écrit une condition à l'aide de l'instruction "**if**"

L'alternative "**else**" est facultative

Pour écrire d'autres options, on écrit "**else if**", et "**else**" pour la dernière option

**On exécute les instructions d'un seul bloc (if ou else)**

# Correction Silent Teacher - Les conditions

```
if (3 < 5) {  
    var a = 1  
} else {  
    var a = 3  
}  
a + 2
```



**Si 3 est inférieur à 5 je défini la variable a avec comme valeur 1 puis je retourne la valeur de a + 2**

On remarque que notre **structure de contrôle conditionnelle utilise un opérateur qui retourne un boolean**

# Correction Silent Teacher - Les fonctions

```
function hi (a, b) {  
    return a * b  
}
```

hi(0, 1)

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 7 | 6 | 4 | 9 | 5 | 2 | 1 |
| 3 | 8 | 0 |   |   |   |   |

```
function hi (a, b) {  
    if (a < b) {  
        return a + b  
    } else {  
        return a * b  
    }  
}
```

```
function hi (a, b) {  
    return a * b  
}
```

hi(3, 3)

|    |   |   |   |   |    |   |
|----|---|---|---|---|----|---|
| 5  | 4 | 7 | 9 | 1 | 11 | 3 |
| 10 | 2 | 8 | 0 |   | ?  |   |

```
function hello (a, b) {  
    return hi(a, b + 1)  
}
```

hello(3, 3)

|    |   |    |   |   |   |   |
|----|---|----|---|---|---|---|
| 3  | 8 | 12 | 7 | 4 | 2 | 5 |
| 10 | 1 | 6  | 9 | ? |   |   |



Des fonctions dans des fonctions

# Correction Silent Teacher - Les fonctions

```
function hi (a, b) {  
    return a * b  
}
```

hi(0, 1)

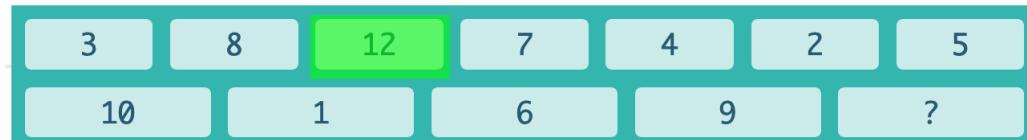
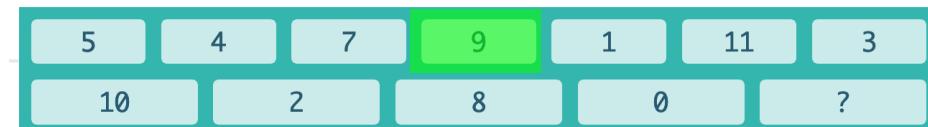


```
function hi (a, b) {  
    if (a < b) {  
        return a + b  
    } else {  
        return a * b  
    }  
}
```

```
function hi (a, b) {  
    return a * b  
}
```

```
function hello (a, b) {  
    return hi(a, b + 1)  
}
```

hello(3, 3)



Mais bien sûr...

# Correction Silent Teacher - Les fonctions

## Step by step

```
function hi (a, b) {  
    return a * b  
}
```

```
function hello (a, b) {  
    return hi(a, b + 1)  
}
```

hello(3, 3)

# Correction Silent Teacher - Les fonctions

À l'instar d'un robot, une fonction permet d'exécuter une série d'instructions autant de fois qu'on le souhaite.



```
1 function faireDuJus(nombreOranges) {  
2     // Instruction 1 : Découper l'orange  
3     // Instruction 2 : Presser les deux parties de l'orange  
4     // Instruction 3 : Répéter les instructions précédentes  
5     //     autant de fois qu'il y a d'oranges  
6 }  
7  
8 faireDuJus(6);
```

Je suis un robot qui fait une quantité de jus d'orange  
en fonction du nombre d'oranges qu'on me donne

# Correction Silent Teacher - Les fonctions

Une fonction commence par le mot clé "function" suivi de son nom et de ses **paramètres**

Les paramètres sont des variables disponibles uniquement à l'intérieur de la fonction

```
1 function faireUnGateau(nombreOeufs) {  
2     // Ici nombreOeufs vaut 3  
3 }  
4  
5 faireUnGateau(3)  
6 // Ici nombreOeufs vaut undefined
```

# Correction Silent Teacher - Les fonctions

Il est possible de stocker le résultat d'une fonction dans une variable

```
1 function helloWorld() {  
2     return "Hello World !";  
3 }  
4  
5 var message = helloWorld();  
6 // Ici, message vaut "Hello World !"
```

Ok mais c'est quoi un algo ?

# C'est quoi un algorithme ?



Des ingrédients



Un chef heureux



Des ustensiles



Un petit gâteau au yaourt

| Ingrédients                          | Préparation  |
|--------------------------------------|--|
| Nombre de personnes<br><br>6         | <b>Temps Total : 45 min</b><br><br><b>Préparation : 15 min   Cuisson : 30 min</b>                          |
| 3 œufs                               | <b>1</b> Mettre dans cet ordre un pot de yaourt nature, la farine, le sucre, le sucre vanillé et mélanger. |
| 1 yaourt nature                      | <b>2</b> Rajouter les 3 œufs, mélanger.  |
| 1 sachet de levure en poudre (5,5 g) | <b>3</b> Mettre l'huile, mélanger et ajouter le sachet de levure.  |
| 2,5 pots de yaourt vides de farine   | <b>4</b> Mélanger encore, la pâte doit être lisse.   |
| 1,5 pot de yaourt vide de sucre      | <b>5</b> Beurrer un moule à manqué et y verser la pâte.  |
| 1 sachet de sucre vanillé (7,5 g)    |  |

Une recette de cuisine de gateau au yaourt

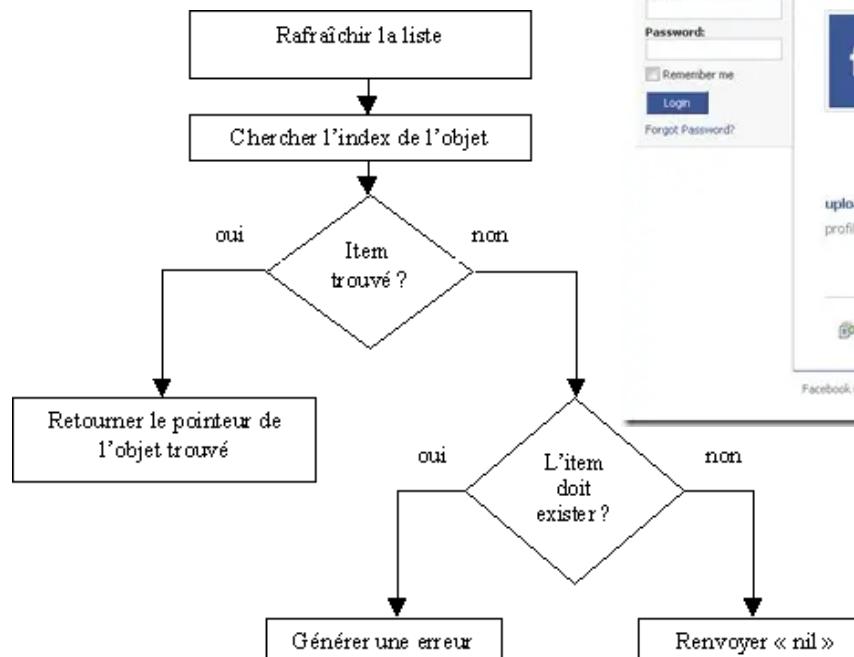
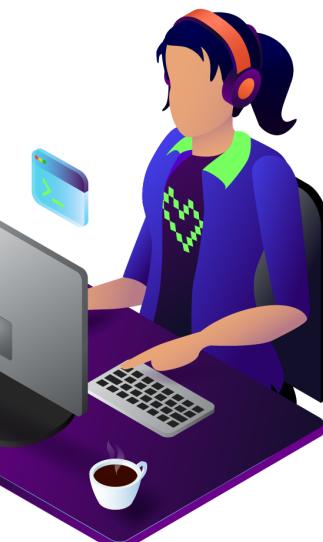
# C'est quoi un algorithme ?

```
1 var jaiFaim = true;  
2 var notesDeLaClasse = [20, 19, 17, 18, ...];
```

```
1 if (jaiFaim) {  
2   return "Miam, je mange";  
3 } else {  
4   return "Je bois un café";  
5 }
```

La donnée

Structures de contrôle / opérateurs...



Un·e dev

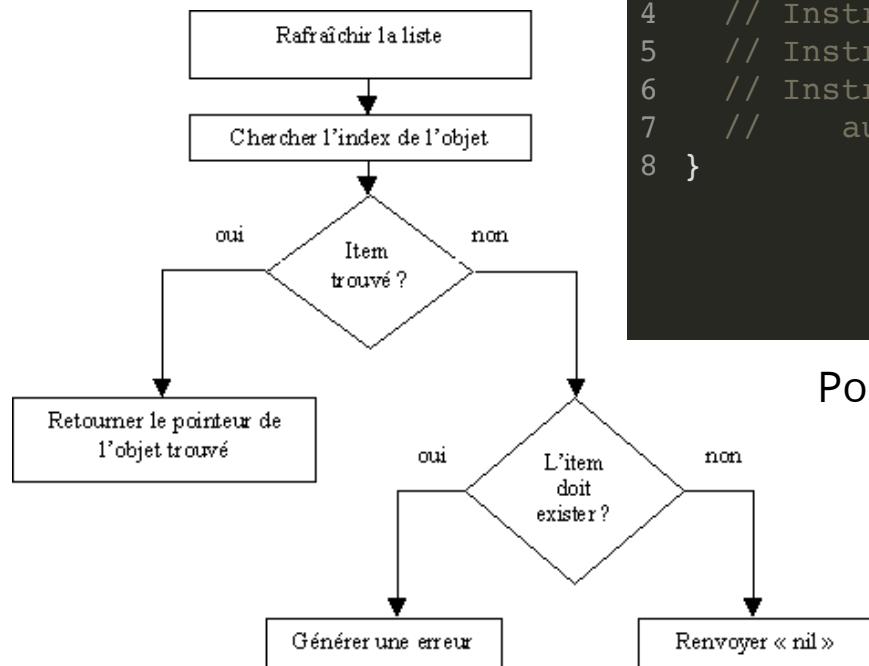
Un algorithme



Un logiciel

# C'est quoi un algorithme ?

Un algorithme est la séquence d'instructions manipulant la donnée et qui permet de réaliser un logiciel.



```
1 function faireDuJus(nombreOranges) {  
2     // TODO :  
3     //  
4     // Instruction 1 : Découper l'orange  
5     // Instruction 2 : Presser les deux parties de l'orange  
6     // Instruction 3 : Répéter les instructions précédentes  
7     //    autant de fois qu'il y a d'oranges  
8 }
```

Pour les petits algos, s'aider des commentaires

Pour les algos plus conséquents vous pouvez vous aider de diagrammes

**Des questions ?**

**Go pratique !**