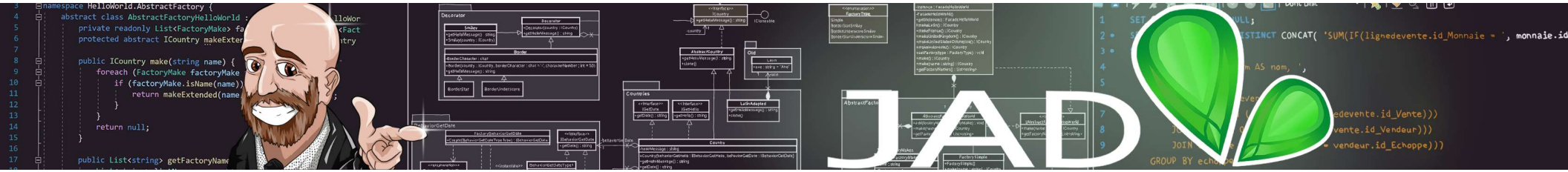


ALGORITHMIE

jeanaymeric@gmail.com

SOMMAIRE

- Définition
- Variable et type de données
- Algèbre de Boole
- Structure conditionnelle
- Structure répétitive

DÉFINITION

- Le mot algorithme vient du nom d'un mathématicien perse du IXème siècle, Al-Khwârizmî
- Larousse :

  « Ensemble de règles opératoires dont l'application permet de résoudre un problème énoncé au moyen d'un nombre fini d'opérations. Un algorithme peut être traduit, grâce à un langage de programmation, en un programme exécutable par un ordinateur. »
- Wikipédia :

  « Un algorithme est une suite finie et non ambiguë d'opérations ou d'instructions permettant de résoudre une classe de problèmes. »
- JAD :

  « Un algorithme est un mécanisme permettant à un élément de réaliser une tâche complexe qu'il ne sait pas réaliser. »

VARIABLE ET TYPE DE DONNÉES

- Wikipédia :

  « Une variable est un symbole (habituellement un nom) qui renvoie à une position de mémoire dont le contenu peut prendre successivement différentes valeurs pendant l'exécution d'un programme. »

- JAD :

  « Une variable est le nom que l'on a donné à un emplacement dans lequel on a mis une information. Cette information a un type (un nombre, un caractère, un texte, …) et ce type définit la taille de l'emplacement. Ainsi une variable ne peut généralement accueillir que des données de même type. »

VARIABLE ET TYPE DE DONNÉES
VOTRE PREMIER ALGORITHME

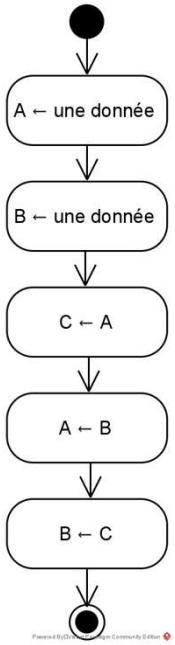
En algorithmie, le symbole ← signifie reçoit.

```
DEBUT
    A ← une donnée
    B ← une donnée
    # Complétez l'algorithme afin qu'à la fin de son exécution
    # les contenus des variables A et B soient intervertis.
FIN
```

VARIABLE ET TYPE DE DONNÉES
VOTRE PREMIER ALGORITHME

Solution

```
DEBUT
    A ← une donnée
    B ← une donnée
    C ← A
    A ← B
    B ← C
FIN
```



VARIABLE ET TYPE DE DONNÉES
VOTRE PREMIER ALGORITHME

On continue en compliquant un peu les choses.

```
DEBUT
    A est un nombre
    B est un nombre
    A ← un nombre
    B ← un nombre
    # Compléter l'algorithme mais cette fois-ci sans passer par
    # une autre variable.
FIN
```

VARIABLE ET TYPE DE DONNÉES
VOTRE PREMIER ALGORITHME

Solution

```
DEBUT
    A est un nombre
    B est un nombre
    A ← un nombre
    B ← un nombre
    A ← A + B
    B ← A - B
    A ← A - B
FIN
```

VARIABLE ET TYPE DE DONNÉES

- Différents types de données
  - Atomique
    - Entier : 1 ; 4 ; -5 ; 0 ; …
    - Réel : 1,0 ; 3,14 ; -362,78 ; $3{,}5.10^{32}$; $7{,}489.10^{-54}$
    - Caractère : ‘A’ ; ‘a’ ; ‘5’ ; ‘à’ ; ‘?’; ‘@’
    - Booléen : VRAI ; FAUX ou 0 ; 1
  - Composite
    - Chaîne de caractères : "Bonjour" ; "3,14" ; "Vrai"
    - Tableau : [1;2;3;4] ; [‘a’; ‘e’; ‘i’ ; ‘o’ ; ‘u’ ; ‘y’]; ["Lundi" ; "Mardi" ; "Mercredi" ; "Jeudi" ; "Vendredi"]
      [[1;2;3] ; [7;8;9] ; [4;5;6] ; [3;5;7] ; [0;5;9]]

- Booléen :

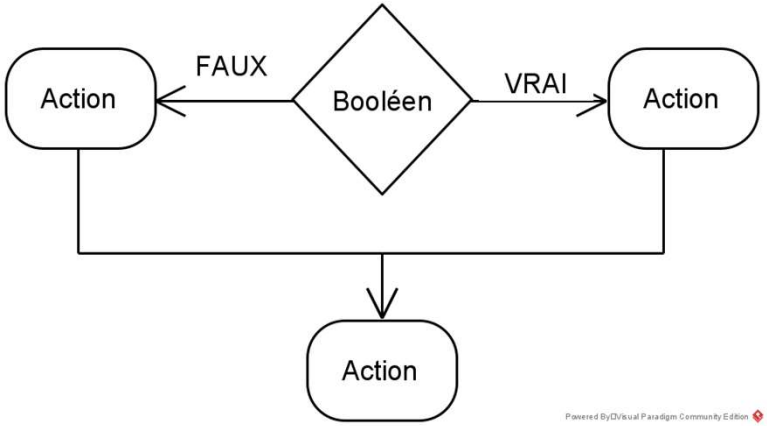
  « Un booléen est un type de variable à deux états (généralement notés vrai et faux), destiné à représenter les valeurs de vérité de la logique et l'algèbre booléenne. Il est nommé ainsi d'après George Boole, fondateur dans le milieu du xixe siècle de l'algèbre portant son nom. »

PREMIÈRE STRUCTURE CONDITIONNELLE

```
SI <Booléen> ALORS
        # Instruction exécutée si le booléen vaut VRAI
        # On peut bien entendu entrer plusieurs instructions
FINSI
```

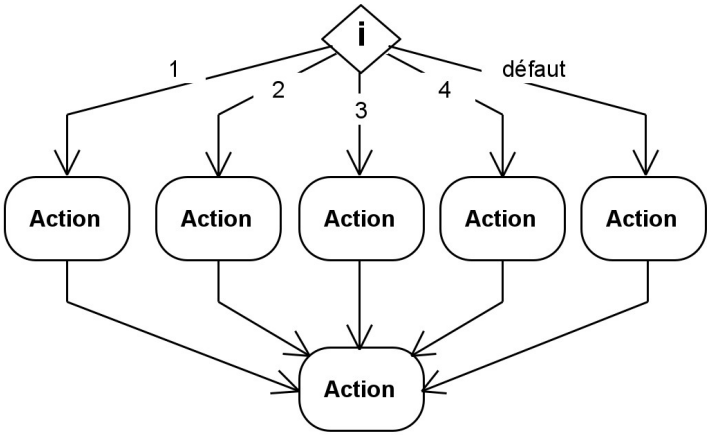
SECONDE STRUCTURE CONDITIONNELLE

```
SI <Booléen> ALORS
        # Instruction exécutée si le booléen vaut VRAI
SINON
        # Instruction exécutée si le booléen vaut FAUX
FINSI
```



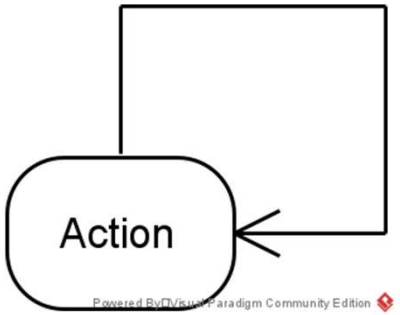
TROISIÈME STRUCTURE CONDITIONNELLE

```
SELON <variable>
        CAS <valeur 1> :
                # Instruction exécutée si variable = valeur 1
        CAS <valeur 2> :
                # Instruction exécutée si variable = valeur 2
        CAS <valeur n> :
                # Instruction exécutée si variable = valeur n
        AUTRE CAS :
                # Instruction exécutée dans tous les autres cas
FINSELON
```



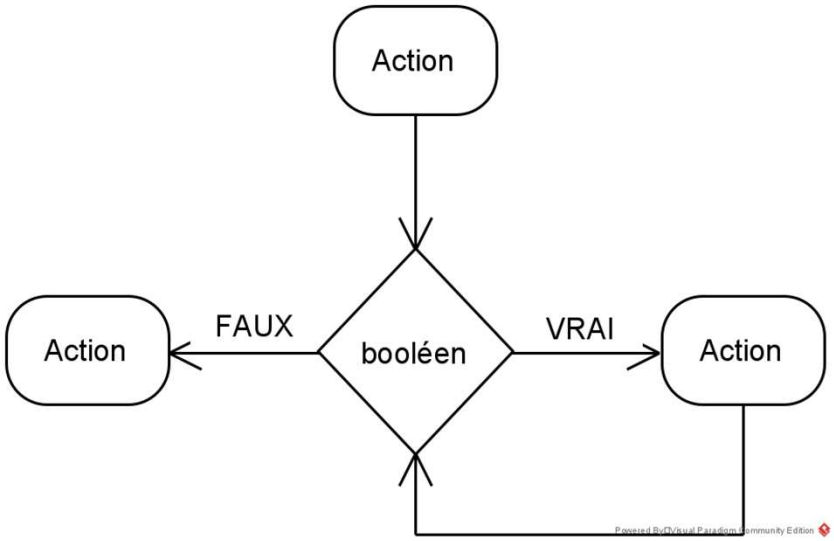
STRUCTURE RÉPÉTITIVE
LA BOUCLE REPETER

```
REPETER
        # Boucle infinie
FINREPETER
```



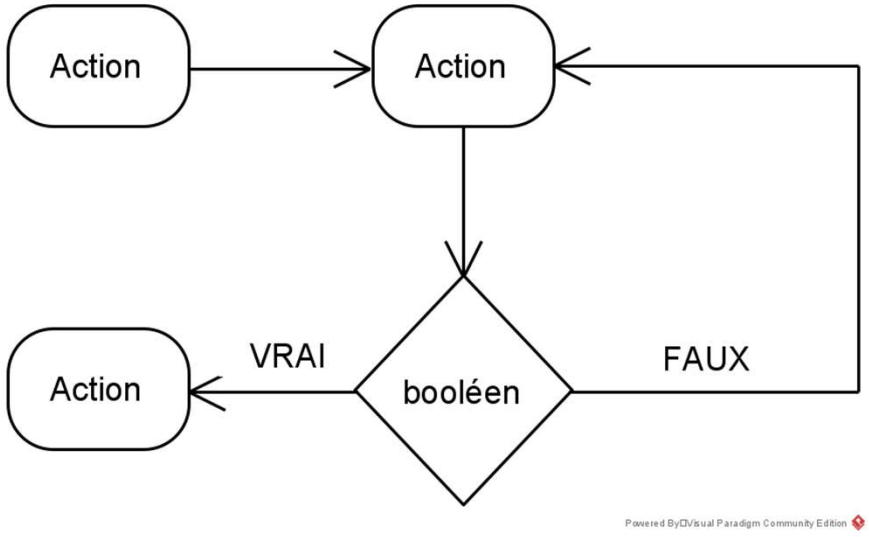
STRUCTURE RÉPÉTITIVE
LA BOUCLE TANTQUE

```
TANQUE <booléen>
        # Instruction répétée tant que booléen vaut VRAI
FINTANTQUE
```



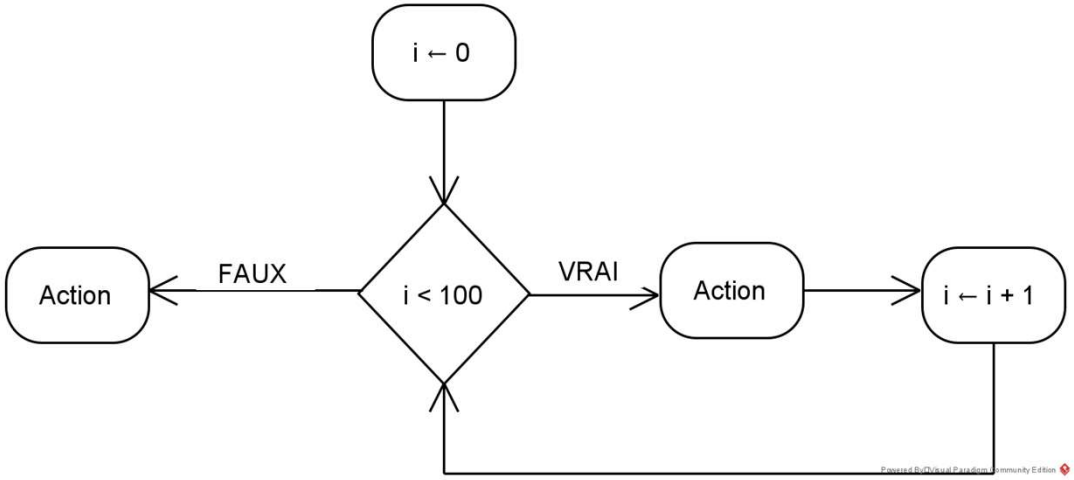
STRUCTURE RÉPÉTITIVE
LA BOUCLE REPETER JUSQU'A

```
REPETER
        # Instruction répétée jusqu'à ce que booléen valle VRAI
JUSQU'À <booléen>
```



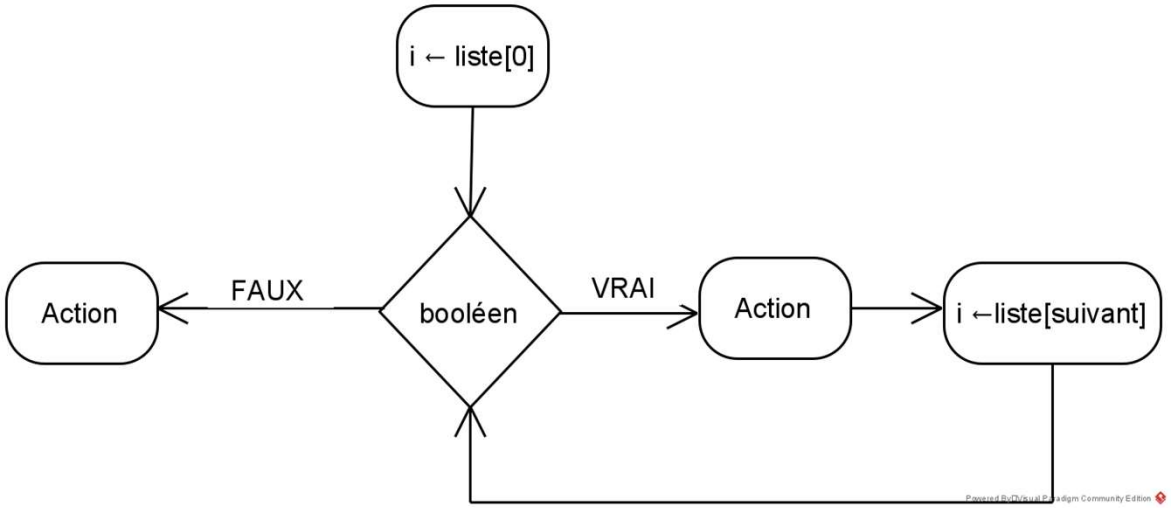
STRUCTURE RÉPÉTITIVE
LA BOUCLE POUR

```
POUR <variable> ALLANT DE <début> A <fin> PAS DE <pas> FAIRE
        # Instruction répétée plusieurs fois
        # exactement (fin - début) / pas fois
FINPOUR
```



STRUCTURE RÉPÉTITIVE
LA BOUCLE POUR CHAQUE

```
POUR CHAQUE <variable> ELEMENT DE <liste> FAIRE
        # Instruction répétée plusieurs fois
        # exactement nombre d'éléments de liste fois
FINPOUR
```



VOILÀ, C'EST TOUT !

- Vous connaissez désormais l'ensemble des structures de contrôles et l'ensembles des boucles permettant de programmer. Il n'en existe pas d'autre et toutes n'existent pas dans tous les langages de programmation.
- Au final, vous avez :
  - 3 structures conditionnelles, aussi appelées tests
  - 5 structures répétitives, aussi appelées boucles