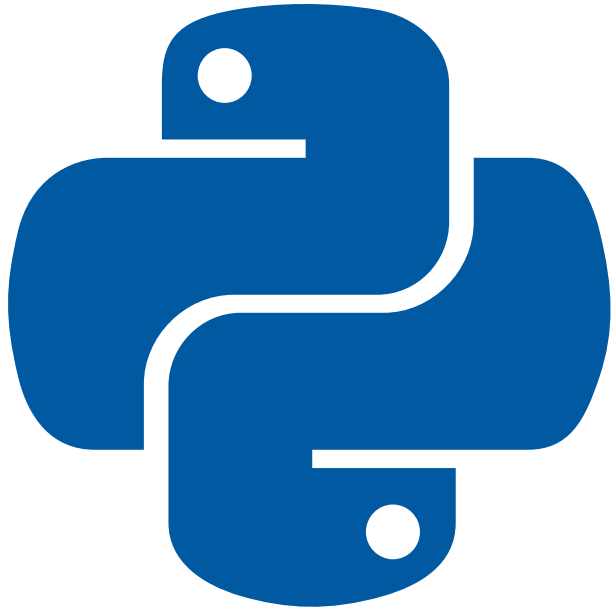
Structure séquentielle

Les Bases de l'algorithmiques

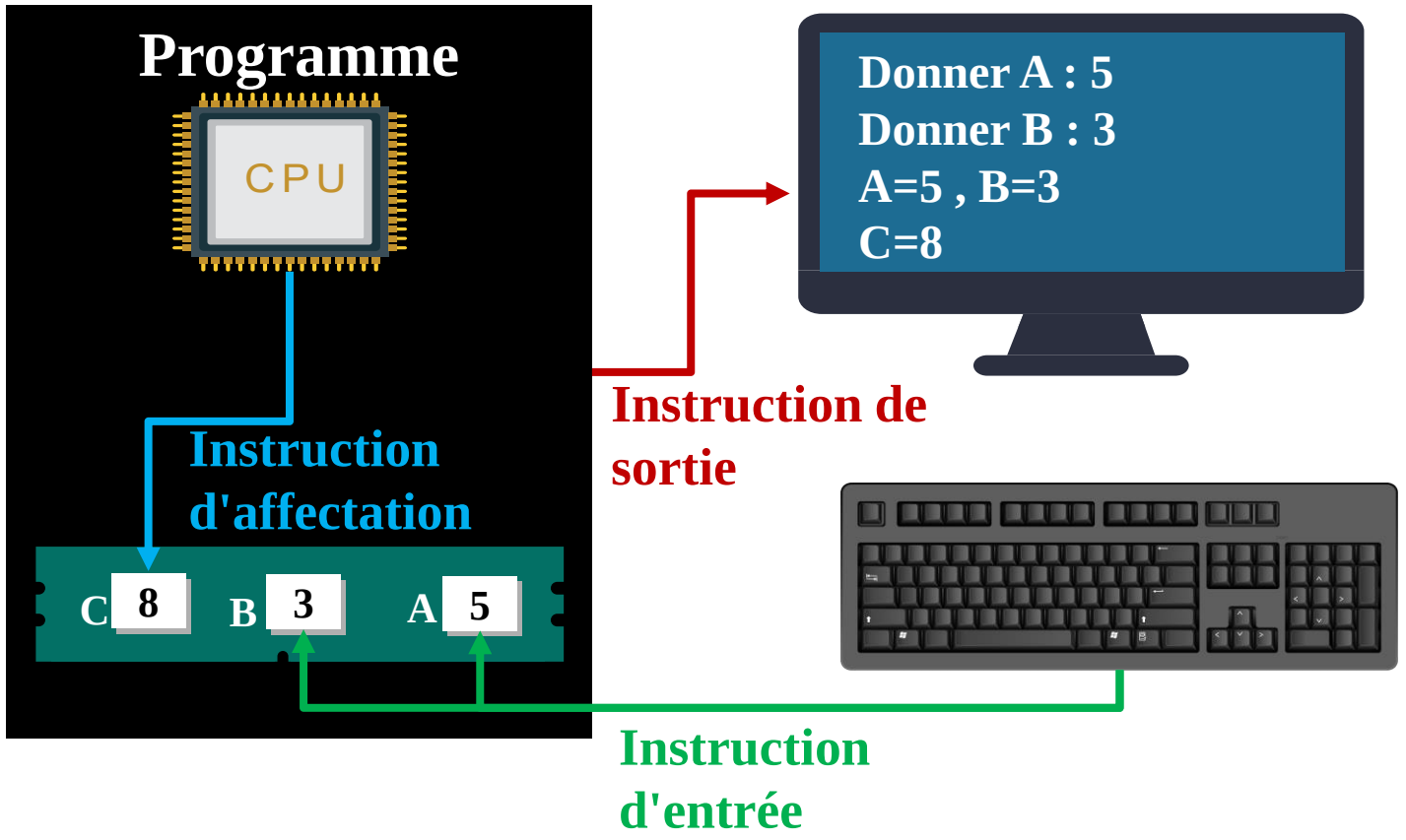
Réalisation : Omar OUGHZAL



Instructions de bases d'un programme

Une instruction : est une action élémentaire commandant à la machine un calcul ou une communication avec l'un de ses périphériques d'entrées ou sortie.

```
Algorithme    Programme
Variables A,B : Entier
Début
    Ecrire("Donner A : ")
    Lire(A)
    Ecrire("Donner B : ")
    Lire(B)
    C ← A + B
    Ecrire ("A=",A, (", B=",B)
    Ecrire("C=",C)
Fin
```



Les Expressions

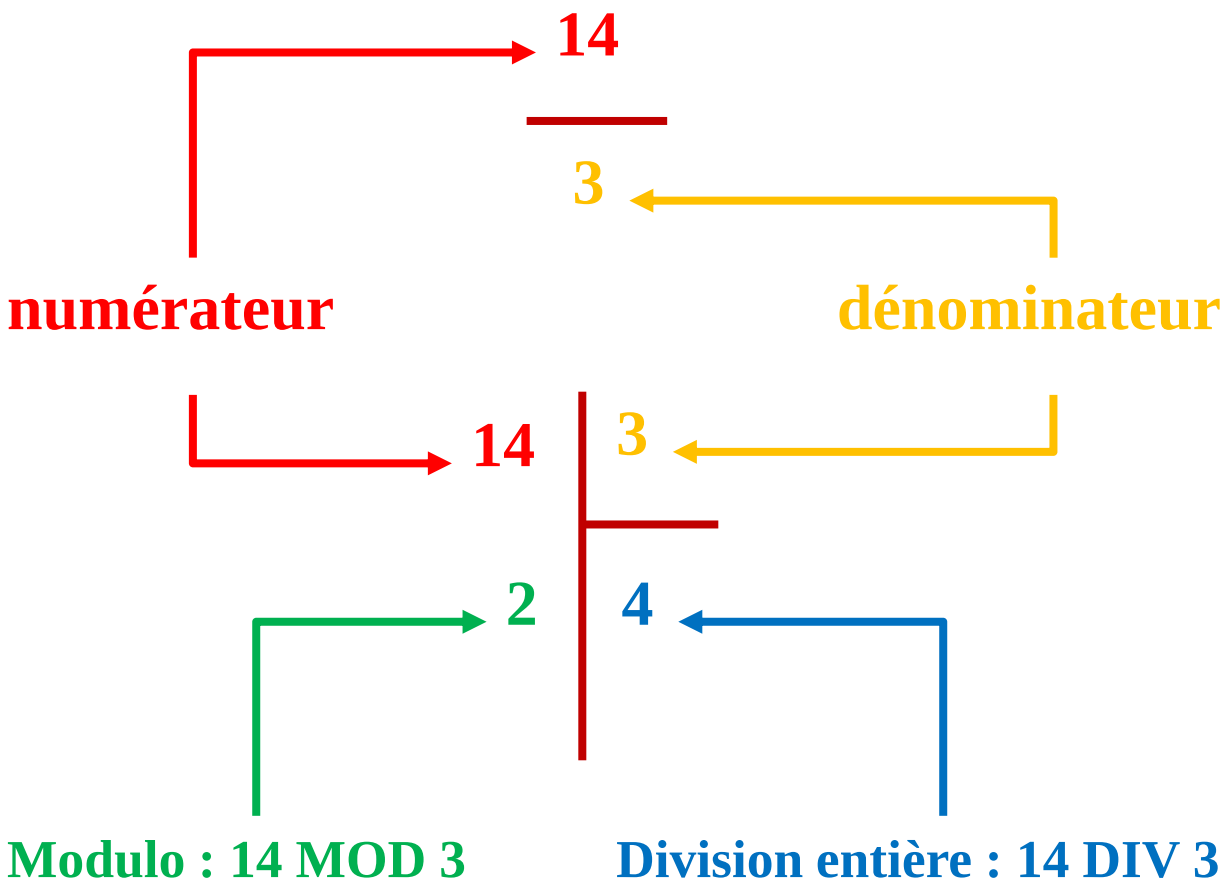
- **Une expression** : est une combinaison d'opérandes et d'opérateur qui sont évalués selon des règles particulières de précédence et d'associativité
- **Un opérateur** : est un signe que agit sur une ou deux variables pour produire un résultat

Type d'expression	Opérateurs	Exemples	Valeur évaluée
Arithmétique	+, -, *, / , DIV, MOD	A + B – C*5	Numérique
Comparaison	>, <, >=, <=, =, <>	7 >= 6	Booléenne
Logique	ET, OU, NON	A OU B	Booléenne
Concaténation	+	"Face" + "Book"	Chaîne de caractères

Opérateurs Arithmétiques

Opérateur	Symbole	Exemple	Résultat
Addition	+	3+2	5
soustraction	-	8 - 4	4
Multiplication	*	3*7	21
Division	/	11/4	2.25
Exposant	^	2^4	16
Modulo (reste de la division)	MOD	10 MOD 3	1
Division entière	DIV	20 DIV 6	3

Division entière et modulo



Exercice

Opération	Résultat
20 MOD 6	
5 DIV 2	
10 / 4	
8 MOD 2	
103 DIV 4	

Priorités des opérateurs

- Les opérateurs ont des priorités différentes
- Les opérateurs avec un nombre d'ordre petit sont calculés en premier

Ordre	
1	+, - (unaire)
2	^
3	*, /, Div, MOD
4	+, - (binaire)

Opérateurs de comparaison

Opérateur	Symbole	Exemple	Résultat
Supérieur	>	3 > 2	Vrai
Supérieur ou égale	>=	5 >= 3	Faux
Inférieur	<	3 < 7	Faux
Inférieur ou égale	<=	11 < = 11	Vrai
Différent	<>	2 <> 3	Vrai

NB : Les opérateurs logique peuvent être appliqué à tous les types des données

Comparaison des caractères et chaînes

L'ordinateur stocke les caractères sous forme de nombre selon le codage ASCCI (American Standard Code for Information Interchange) :

Exemples :

- **Espace** 32
- **0 à 9** 48 à 57
- **A à Z** 65 à 90
- **a à z** 97 à 122

La comparaison des caractères se fait selon leurs code ASCII

Exemples : 'A' < 'a' → Vrai car 65 < 95

La comparaison des chaîne se fais caractère par caractère

Exemple : '**Maro**c' < '**Marr**akech → Vrai

Opérateurs logiques

A	B	Non A	A et B	A ou B
Vrai	Vrai	Faux	Vrai	Vrai
Vrai	Faux	Faux	Faux	Vrai
Faux	Vrai	Vrai	Faux	Vrai
Faux	Faux	Vrai	Faux	Faux

Exercice

Donner le résultat et le type des expressions suivantes :

- **7+2**
- **5*4**
- **3^3**
- **7 Div 3**
- **10 MOD 4**
- **4.5 / 3**
- **7/3**
- **5+3/2**
- **(3+5)/2**
- **3 >= 6**
- **9 MOD 2 = 0**
- **‘A’ > ‘B’**
- **Vrai et Faux**
- **Vrai Ou Faux**
- **Non Vrai**
- **‘ID’ & ‘1A’**
- **9 MOD 3 = 1**
- **Non (Vrai et Faux)**
- **4^0.5**
- **8 <>2**
- **(4 < 6) OU (9 > 2)**
- **NON(NON(5 < 6))**

L'instruction d'affectation

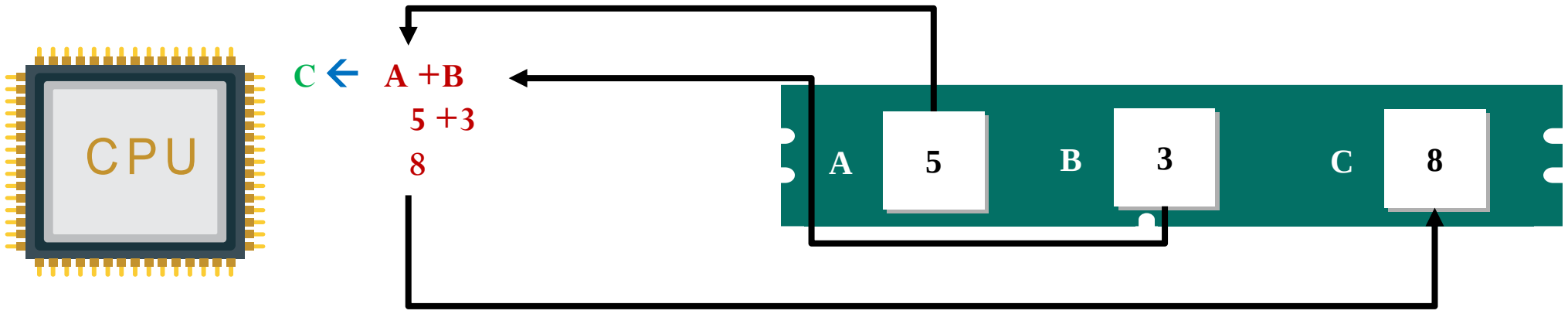
L'affectation : Permet d'affecter une valeur à une variable.

Syntaxe :

Variable ← Expression

Sémantique : une affectation peut être définie en deux étapes :

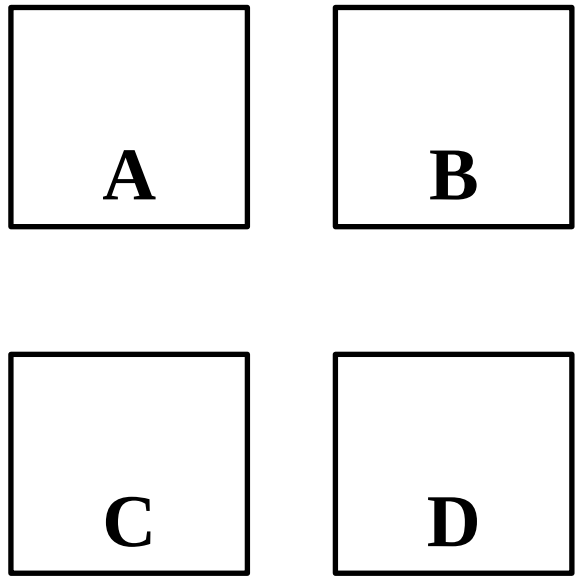
1. **Evaluation la valeur de l'expression qui se trouve à droit de l'affectation**
2. **Placer cette valeur dans la variable**



Exemple d'affectation

Donner les valeurs des variables le long d'exécution de cet algorithme

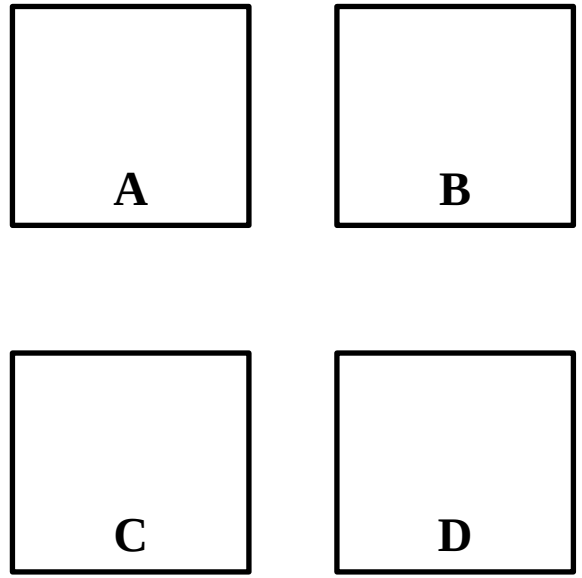
```
Algorithme Calcul
  Variables A, B, C, D : Entier
Début
   A ← 10
   B ← 30
   C ← A + B
   D ← C * A
Fin
```



Exemple d'affectation

Donner les valeurs des variables le long d'exécution de cet algorithme

Algorithme logique
Variables A, B, C,D : Booléen
Début
A ← Vrai
B ← Faux
C ← A et B
D ← A ou B
Fin



L'instruction de sortie

L'instruction de sortie (d'écriture) permet d'afficher des informations à l'écran.

Syntaxe : **Ecrire (Argument1, Argument2, Argument2, Argument2, …)**

Chaque argument est une expression.

Exemple :

A ← 5

B ← 2

S ← ''Monsieur''

Ecrire (A)

Ecrire ("La valeur de A est :", A)

Ecrire (A," + ", B, " = ", A+B)

Ecrire ("Bonjour ", S)

Exercice d'instruction de sortie

```
Algorithme Affichage
Variables
        a, b :entier
Début
        a ← 5
        b ← 2
        ……….
Fin
```

Compléter l'algorithme pour avoir la sortie suivante :

```
La valeur de la variable a est : 5
La somme de 5 et 2 égale à 7
5 * 2 = 10
```

L’instruction d’entrée (ou de lecture)

L’instruction d’entrée : permet de lire une valeur du clavier puis l'affecter à une variable.

Syntaxe : **Lire (Identificateur)**

Exemple : **Lire (a)**

Remarque : Avant de lire une variable, il est conseillé d’écrire un message à l’écran, afin de prévenir l’utilisateur de ce qu’il doit taper

```
Ecrire (’’Donner la valeur de a :’’)
Lire (a)
```

Les commentaires

Les commentaires : sont des lignes destinées aux personnes qui lisent l'algorithme pour expliquer le fonctionnement de l'algorithme

Exemple :

```
// commentaire sur une seule ligne
/* commentaire Sur plusieurs
Lignes  */
```

L'ordinateur ne prenne en compte les commentaires lors de l'exécution du programme.

NB : Les commentaire peuvent être insérer à n'importe quel emplacement dans un algorithme

Exemple :

```
/* Cet algorithme est un teste pour les
commentaires */
Algorithme Commentaire
 // Déclaration des variables
VAR
    N : Entier // N est le nombre des étudiants
Début
    // demander à l'utilisateur le nombre N
    Ecrire("Donner N : ")
    Lire(N) // la valeur saisie sera stockée dans la
    variable N
Fin
```

Exercice

Ecrire un algorithme qui demande à l'utilisateur le rayon d'un disque, calcule la surface et le périmètre et les affiche à l'écran.

N.B : la surface d'un disque est πR^2, Le périmètre d'un cercle est $2 \pi R$

Traduire un Algorithme en python

```
Algorithme Nom_Algo
Var A : Entier
Début
    Ecrire("Entrez A :")
    Lire(A)
    Ecrire(A,"^2 = " , A^2)
Fin
```

```
A = int(input("Entrez A : "))
print(A,"^2 = " , A**2)
```

- Pour traduit un algorithme en Python, il suffit de traduit le corps de l'algorithme
- Pas de d'entête ni de la partie déclarative, ni les mot-clé début et Fin

Traduction des instructions de base

```
Ecrire("Entrez A :")
Lire(A)
Ecrire("Entrez B :")
Lire(B)
C <-- A + B
Ecrire("La Somme est ",C)
```

```
A = int(input("Donner A : "))
B = int(input("Donner B : "))
C = A + B
print("La somme est", C)
```

Instruction	Algorithmiques	Python
affectation	←	=
Sortie	Ecrire(exp1,exp2,…)	print(exp1,exp2,…))
Entrée	Lire(Variable)	Variable = input()

Traduction des commentaires

```
// commentaire sur une seule ligne

/*
   commentaire sur plusieurs lignes
*/


# commentaire sur une seule ligne

"""
   commentaire sur plusieurs lignes
"""
```

Les variables

variable = expression

- Une variable : est un nom pour un emplacement mémoire qui sert à stocker des valeurs
- Une variable est caractérisée par : son nom, son type et sa valeur
- Python est un langage de typage implicite (on ne déclare pas les variables) et dynamique (une variable peur changer de type)
- Le type d'une variable est déterminé par le type de la valeur qui lui est affecté
- Le nom d'une variable doit commencer par une lettre ou "_"
- Python est sensible à la case : Python fait la différence entre la minuscule et la majuscule
- Une variable est créée lorsqu'on lui affecte une valeur
- Les types des variables en python : int, float, str et bool

Traduction de l'instruction d'affectation

Variable ← Expression

```
variable = Expression
Var1,var2,... = Expression1, Expression2,...
Var1,var2 = var2, var1 # échanger les valeurs de var1 et var2
```

- Une expression est une combinaison d'opérandes et d'opérateurs évaluée pour produire une valeur
- Un opérande peut être : une valeur, une variable, un appel à une fonction ou une sous-expression
- Python accepte l'affectation multiple de plusieurs expressions à plusieurs variables en une seule ligne

Traduire les opérateurs

Opérateur	Algorithmiques	Python
Arithmétiques	+, -, *, /	+, -, *, /
Modulo	MOD	%
Division Entière	DIV	//
Exposant	^	**
comparaison	>, >=, <, <=	>, >=, <, <=
égale	=	==
Différent	<>	!=
Et logique	ET	and
Ou logique	OU	or
négation	NON	not
La valeur Vrai	Vrai	True
La valeur Faux	Faux	False

Traduire l'instruction Ecrire

```
Ecrire(Expression1,Expression2,Expression3,…)

print(Expression1,Expression2,Expression3,…,sep=" ",end="\n")
```

- Les arguments des fonctions **Ecrire** et **print** sont des expressions
- Les arguments d'un fonction python peuvent être : **positionnels** (déterminés par leur ordre) ou **mot-clé** (définis par un **mot-clé** sans tenir en compte leur ordre)
- La fonction print a deux arguments mots-clés : **sep** (défini la chaîne de caractères qui sépare les arguments lors de l'affichage, par défaut un espace) et **end** (défini la chaine de caractère ajoutée à la fin des arguments, par défaut "\n" le retour à la ligne

Traduire l'instruction Lire

```
Lire(Variable)
```

```
variable = input("Message")
```

- La fonction input peut prendre un argument qui s'affiche lors de son exécution
- La fonction input retourne la valeur saisie au clavier sous forme de chaîne de caractères, utiliser la fonction **int(input("Message"))** pour lire un **Entier** et **float(input("Message"))** pour lire un **réel**