

L'algorithmique

Variables, constantes et instructions

Une **variable** est une **zone mémoire** qu'un programme va utiliser pour **stocker temporairement** une valeur.

- Une variable est un **triplet** (**identificateur**, **type**, **valeur**).
 - ✓ **L'identificateur** c'est son **nom** qui permet de la **désigner tout au long** de l'algorithme.
 - ✓ Le type est **fixe**.
 - ✓ En revanche la valeur **variera** au cours de l'exécution de l'algorithme.

Une constante est :

- ✓ soit une valeur "**brute**" ;
- ✓ soit une zone mémoire ;

qu'un programme va utiliser pour **stocker temporairement** une valeur **qui ne changera pas** durant l'exécution du programme.

Une constante est une valeur prise dans l'ensemble des valeurs du type.

- **28** est une constante de type **ENTIER** ;
- **3,14** est une constante de type **REEL** ;
- **"A"** est une constante de type **CAR** ;
- **"AZERTY"** est une constante de type **CHAINE** ;
- **VRAI** est une constante de type **BOOLEEN**

Les instructions :

- qui permettent de modifier d'état des variables

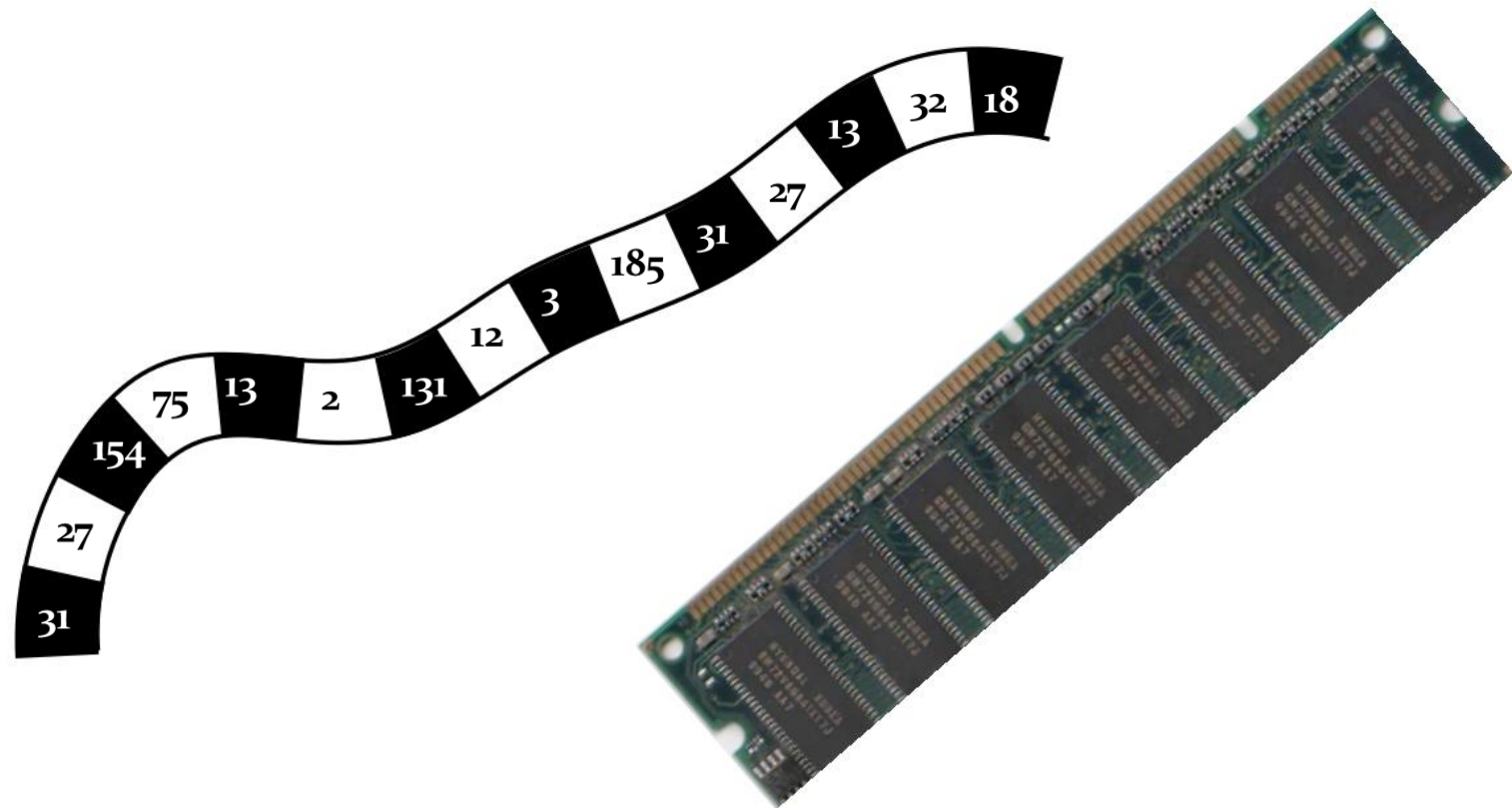
Un algorithme est donc composé d'une **suite** d'instructions qui :

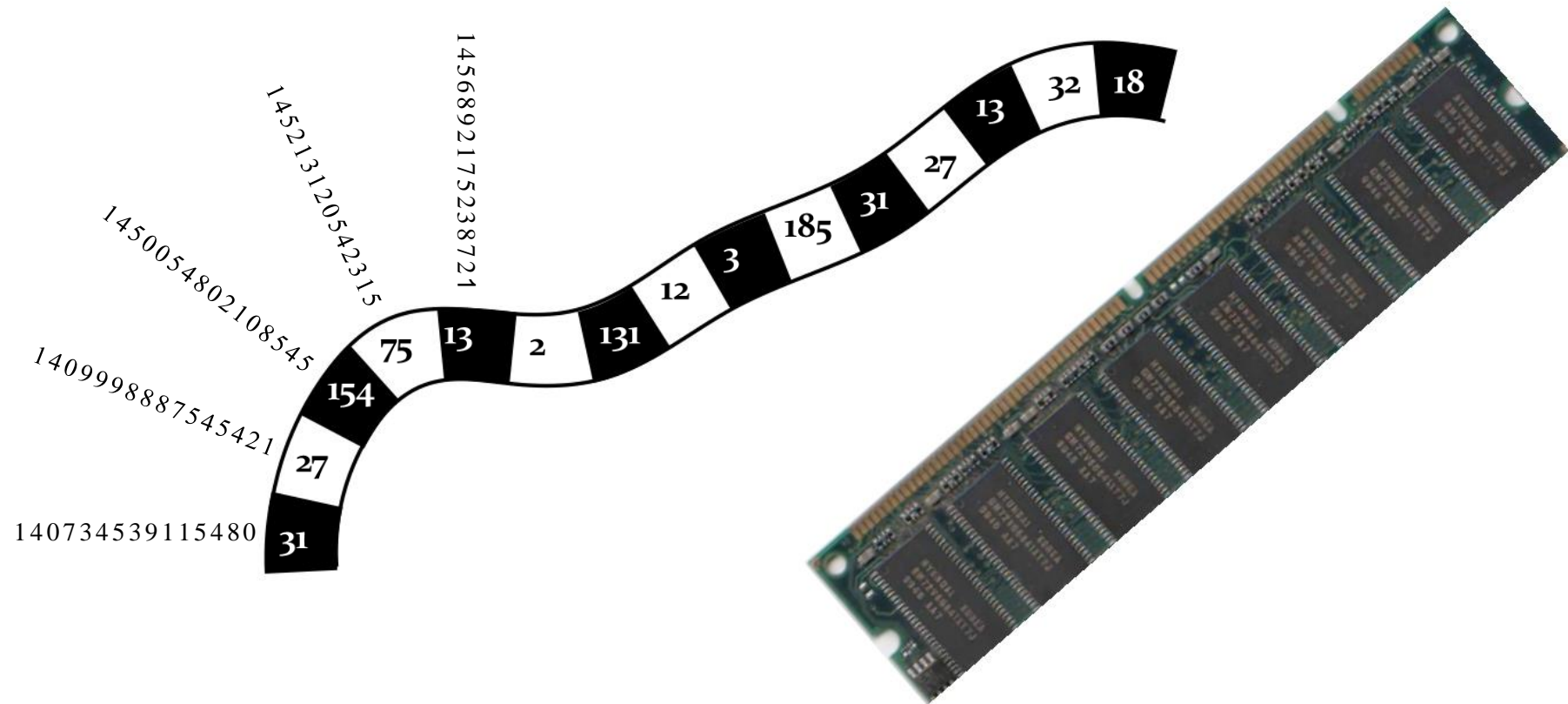
- partant d'une **description** en **mémoire** d'un problème **non résolu** ;
- donnent les **modifications** de la mémoire ;
- permettant d'arriver à une description en mémoire du problème **résolu**.

Les Variables : Nom, types, etc....

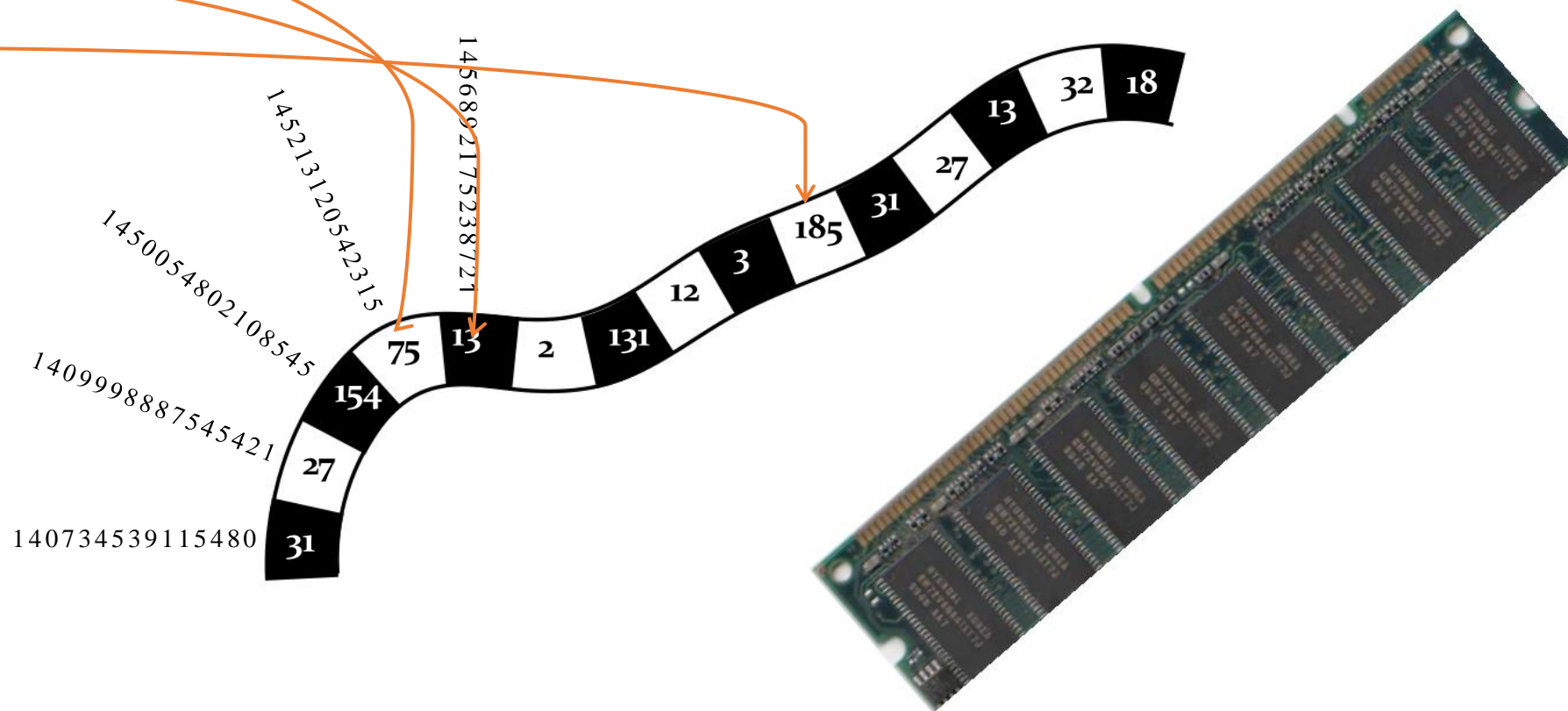
Rappel : Accéder à la mémoire de l'ordinateur....

- ✓ Mémoire de l'ordinateur = **très grand** espace **découpé** en **cases** ;
- ✓ **Chaque** case porte ainsi un **numéro** (**adresse**) pour s'y retrouver ;
- ✓ On peut **consulter/modifier** le contenu d'une case en question dès qu'on en **connait** l'adresse.





| |
|-------|
| année |
| âge |
| x |
| |
| |



Les variables sont **caractérisées** par :

- ✓ un nom (**identifiant**) ;
- ✓ un **type** ;
- ✓ une **valeur courante** (qui **peut** être **modifiée**).

Les variables sont **caractérisées** par :

- ✓ un nom (**identifiant**) ;
- ✓ un **type** ;
- ✓ une **valeur courante** (qui **peut** être **modifiée**).

Les types de base

Les numériques :

➤ **entiers** ;

1, 2 , 3 etc.

➤ **Réels** (flottants) ;

1,3 ; 2,5 ; 3,6 etc.

Les caractères :

- Une des finalités de l'informatique, c'est de **communiquer**...

Les caractères en informatique = ce que l'on **souhaite** écrire...

Les caractères (**suite**) :

```
Ecrire(Ascii_vers_Caractere(36));  
Ecrire(Caractere_vers_Ascii('Z'));
```

\$
90

| | 0 | 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 | 72 | 80 | 88 | 96 | 104 | 112 | 120 |
|---|---|---|----|----|----|----|----|----|----|----|----|----|----|-----|-----|-----|
| 0 | | | | | | (| 0 | 8 | @ | H | P | X | ` | h | p | x |
| 1 | | | | | ! |) | 1 | 9 | A | I | Q | Y | a | i | q | y |
| 2 | | | | | " | * | 2 | : | B | J | R | Z | b | j | r | z |
| 3 | | | | | # | + | 3 | ; | C | K | S | [| c | k | s | { |
| 4 | | | | | \$ | , | 4 | < | D | L | T | \ | d | l | t | |
| 5 | | | | | % | - | 5 | = | E | M | U |] | e | m | u | } |
| 6 | | | | | & | . | 6 | > | F | N | V | ^ | f | n | v | ~ |
| 7 | | | | | ' | / | 7 | ? | G | O | W | _ | g | o | w | |

Les caractères (**suite**) :

- Ce sont des suites de caractères (0,1, 1000 caractères)...
 - ✓ "Bonjour", "Au revoir", "première ligne\ndeuxième ligne",
 - ✓ "X"
 - ✓ ""
- la chaîne de caractères « **vide** » joue un rôle très **important**

Les booléens :

- Ce sont les valeurs **vrai** et **faux** (ou **true** et **false**)...

Les tableaux (**array** ou **TAB**) :

- Omniprésents en programmation !!

```
VAR
    tableau1 : TAB[10] ENTIER
```

Déclaration d'un tableau à une dimension (assimilable à un vecteur) pouvant contenir 10 entiers.

[illegible]

```
VAR
    tableau2 : TAB[3,10] CAR
```


Déclaration d'un tableau à deux dimensions (une matrice) pouvant contenir 30 caractères sur 3 lignes et 10 colonnes.

[illegible]

Les tableaux (**array** ou **TAB**) - exemple :

- `tableau1[5]` est le **cinquieme** élément d'un tableau **1D**.

~~Déclaration d'un tableau à une dimension (assimilable à un vecteur) pouvant contenir 10 entiers.~~

| | | | | | | | | | |
|--|--|--|--|---|--|--|--|--|--|
| | | | |  | | | | | |
|--|--|--|--|---|--|--|--|--|--|

- tableau2[2,3] est le **3eme** élément de la **2eme** ligne d'un tableau **2D**.

Déclaration d'un tableau à deux dimensions (une matrice) pouvant contenir 30 caractères sur 3 lignes et 10 colonnes.

[illegible]

Les noms des variables

Lorsque vous définissez une variable :

- choisissez un nom **approprié**, qui donne une idée de ce qui sera stocké dans la variable ;
- Choisissez un nom assez **court** mais **aussi long** que **nécessaire** (vous devrez l'écrire)
- évitez les noms réservés ;
- évitez également les **lettres accentuées** et les **caractères** de **ponctuation**.

Par contre :

- Vous pouvez utiliser une combinaison **quelconque** de lettres, chiffres et caractère de soulignement (_).

Parmi ces noms de variables, quels sont ceux qui sont valides et ceux qui ne le sont pas ?

| Nom de variable correct | Nom de variable incorrect | Raison |
|-------------------------|---------------------------|------------------------------|
| nomDeVariable | Nom de Variable | comporte des espaces |
| nomDeVariable123 | 123NomDeVariable | commence par un chiffre |
| totoAtMailCityDotCom | toto@mail-city.com | caractères spéciaux @ - et . |
| continuer | continue | nom réservé |

**Pourquoi porter autant d'attention aux
types
alors que tout peut se coder par
des bits ?**

Pourquoi le typage ?

- ✓ Parce qu'il est plus **pratique** d'écrire 'Bonjour' que
01000010011011110110111001101010011011110111010101110010
- ✓ Parce qu'on a **besoin** de savoir **combien de place** (en nombre de bits) il faut **réserver** en mémoire pour stocker une variable....
- ✓ Parce que les calculs qu'on peut faire changent d'un type à l'autre....

Questions ??

Prochain chapitre: Les expressions.