Ministère de l'enseignement supérieur et de la recherche scientifique
Université Djillali BOUNAAMA - Khemis Miliana (UDBKM)
Faculté des Sciences et de la Technologie
Département de Mathématiques et d'Informatique

Chapitre 2

Algorithme séquentiel simple

MI-L1-UEF121 : Algorithmiques et Structures de Données I

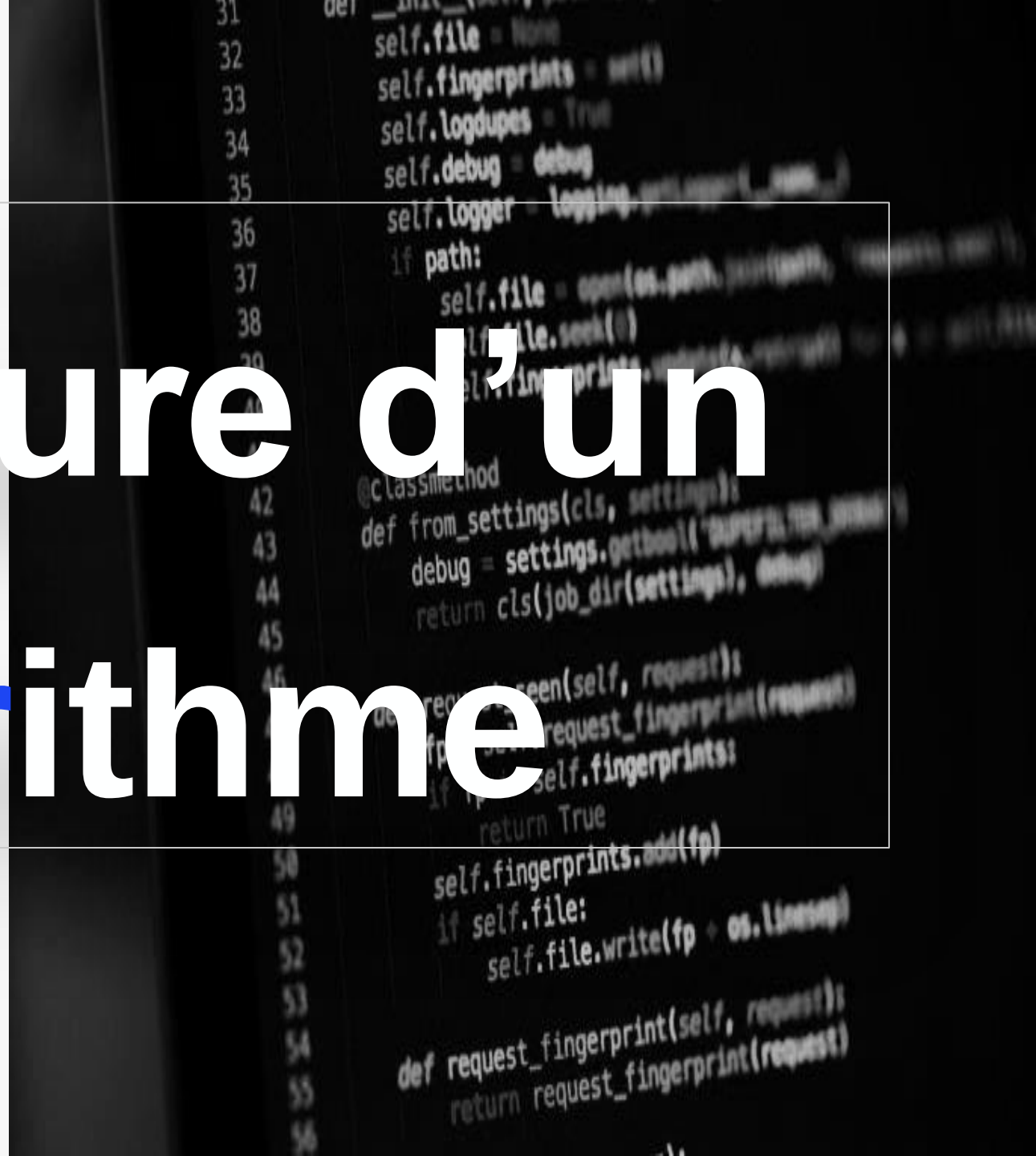
Noureddine AZZOUZA

n.azzouza@univ-dbkm.dz

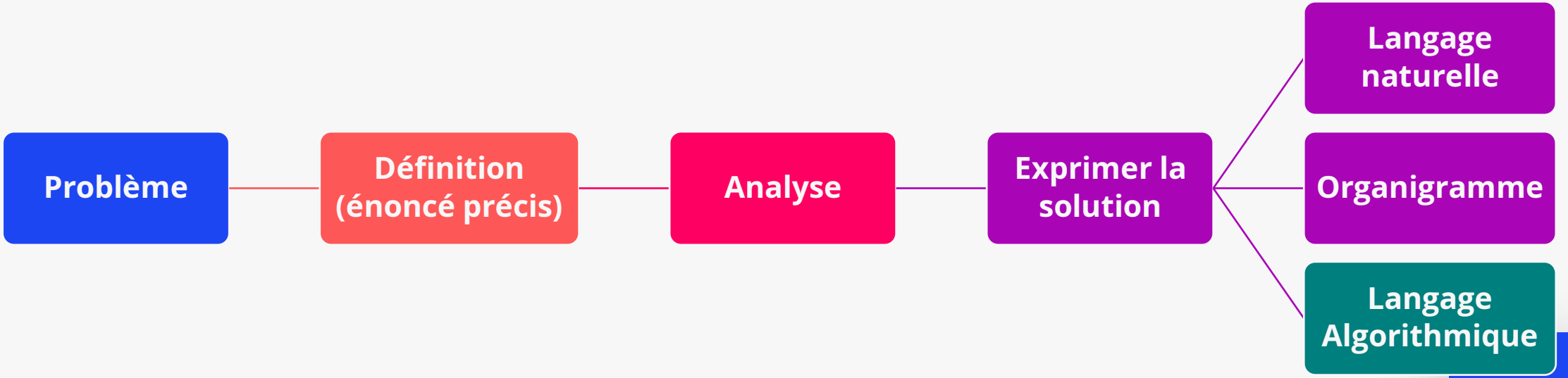
Plan du Cours

1. Structure d'un algorithme
2. Les objets : Constantes, Variables et Types
3. Les actions de base
    - 3.1 L'affectation
    - 3.2 Les expressions
    - 3.3 Les entrées / sorties
4. Représentation d'un algorithme par un organigramme
5. Traduction en langage de programmation

Structure d’un algorithme



Besoin d'un langage algorithmique



- ✓ **Langage naturelle** (description littéraires) : imprécision, ambiguïté, propres règles et conventions, explications différentes du même concept …
- ✓ **Organigramme :** fouillis, difficile à modifier et mettre à jour, …

Langage algorithmique

- ✓ Un **langage** (ou formalisme) **algorithmique** est un ensemble de ***conventions*** (ou de ***règles***) dans lequel on exprime toute solution d'un problème donnée.
- ✓ Appelé aussi : **Pseudo Code**,
- ✓ **Propriétés :**
  - ➢ Un langage ***commun***;
  - ➢ Principe de ***communication***;
  - ➢ ***Précision*** et clarté (non ambiguïté)

Structure d'un Algorithmique

Entête

Environnement

Corps

Algorithme $nom\_algorithme$

Type
Const
Var
} **Déclaration des données (objets) et des modules**

Début

- {**Initialisations**}
-
-
-
-
-
-
- {**Résultats et sorties**}

Fin.

} **Traitements et manipulation des objets**

Les Commentaires

- ✓ Donne une description humaine dans un code destiné à la machine.
- ✓ Maintenance du code simplifié et donc, débogage rapide
- ✓ Important lorsque écrire des fonctions pour d'autres utilisateurs

Syntaxe

```
//      Commentaire sur une seul ligne
/*      Commentaire sur
        plusieurs lignes
*/
```

Exemple : Carré d'un Nombre

Entête

Déclaration

Corps

```
Algorithme  Carre;

Var  N, carre : entier ;

Debut
    //les entrées
     Lire (N);

      //manipulation des données
      carre ← N*N

    //les sorties
      Ecrire ( ' le carré de ',N,' est ', carre);

Fin.
```

Exemple : Carré d'un Nombre

Programme en langage PASCAL

```
program Carre_Exemple;

uses    crt;

var     N, carree : Integer;

begin
    //lecture des entrées
    ReadLn(N);

    carree := N*N;
    {
        écriture et affichage
        des résultats
    }
    WriteLn('le carre de ',N,' est ', carree);

end.
```

Entête

Déclaration des bibliothèques

Déclaration des variables

Corps

Programme en langage C

```
#include <stdio.h>

int main (){

    int N, carre;

    //lecture des entrées
    scanf("%d", &N);

    carre = N*N;

    /* écriture et affichage
        des résultats */
    printf("le carre de %d est %d", N, carre);

    return 0;
}
```

Déclaration des bibliothèques

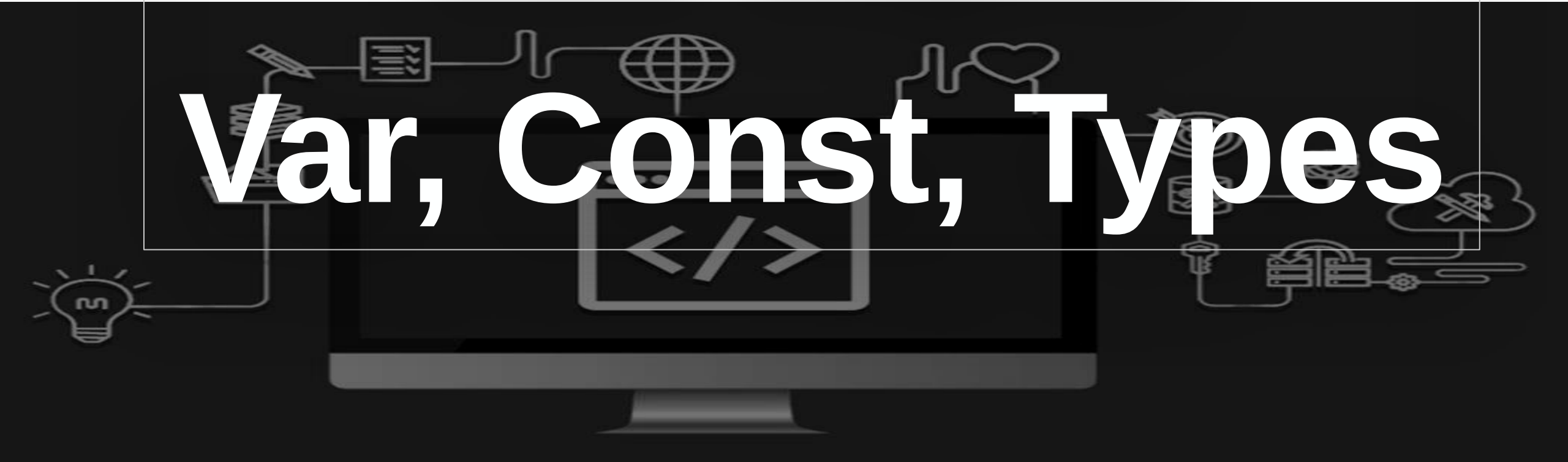
Déclaration des variables

Corps

Fonction principale

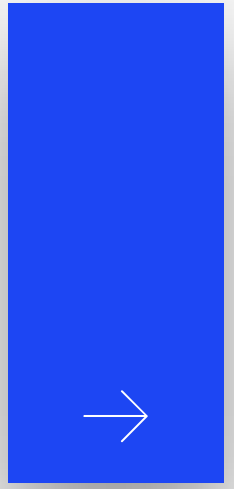
Les objets

Var, Const, Types



Les objets: Représentation mémoire

- ✓ Tout **objet** manipulé par un **algorithme** (ou un **programme**) est ***stocké*** dans la ***mémoire centrale*** (RAM).
- ✓ La ***mémoire centrale*** est constitué d'une ***suite de cases*** contiguës appelées « **cases** » ou « **cellules** » mémoire.
- ✓ Chaque **case mémoire** est caractérisée par:
  - ➢ Une ***adresse*** : ***unique*** qui référence la case
  - ➢ Un ***emplacement*** : pour stocker les ***valeurs*** des objets



Les objets: Représentation Mémoire

Adresse @	Mémoire (Valeur)	Objet (Var, Const, ..)
63999		
63998		
63997		
14792	10	i
14791	14.75	Note
14790	0	J
14789		
14788	Ali	Nom
2		
1		
0		

Les objets: Définition

- ✓ Tous les **objets** constituants d'un **algorithme** doivent être ***décrite*** ou ***déclaré*** dans l'***environnement*** (ou la partie « ***déclaration*** »).
- ✓ Chaque objet est caractérisé par :
  - ➢ **Un Nom** : « ***identificateur unique*** » : une suite de ***caractères alphanumérique*** qui permet de le désigner et le distinguer
  - ➢ **Un Type** : qui indique la ***nature*** de l'ensemble dans lequel il prend ses ***valeurs***
  - ➢ **Une valeur** : qui indique la ***grandeur*** prise par un objet à un moment donné

Les objets: Identificateur

- ✓ Un **nom** ou **identificateur** est une suite de ***caractères alphanumérique*** dont le premier caractère est ***alphabétique***.
  - ➢ L'identificateur peut etre : nom du programme, noms des variables et des constantes, noms des fonctions
  - ➢ Peux contenir des lettres et des chiffres
  - ➢ Pas de (la plupart) signes de ponctuation
- ✓ **Exemples :**
  - ➢ **Identificateurs valides** : produit, i, j , T1, L_21, surface, nom_etudiant,
  - ➢ **Identificateurs non valides** : 5etudiants, date de sortie, x+y, T_1, …

Programmation: Identificateurs

Identificateurs en langage PASCAL

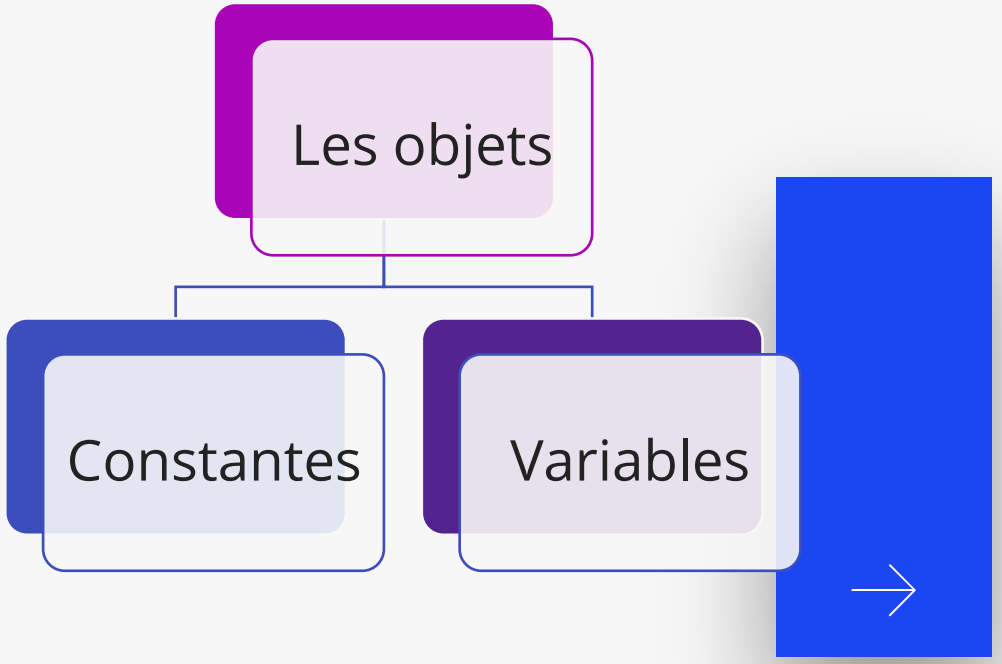
1. L'identificateur est "Unique"
2. Il ne doit pas contenir des:
   - Caractères accentués, ni d'espace, ni des caractères tels que %,?,*,.,-,
3. Il doit exclusivement être composé des:
   - 26 lettres de l'alphabet, des 10 chiffres et le caractère de soulignement
4. Un chiffre ne peut pas être placé au début d'un identificateur.
5. ne différencie pas entre majuscule et minuscule.

Identificateurs en langage C

1. Les meme conditions que le langage PASCAL
2. fait la distinction entre lettres minuscules et majuscules.
3. Par convention:
   - Les noms des variables et fonctions ne contiennent pas de majuscules
   - Les constantes ne contiennent pas de minuscules
   - utilise le souligné pour séparer les mots

Les objets: Catégories

- ✓ Les **objets** servent à ***stocker les données*** manipulées par l'***algorithme***
- ✓ Il existe deux catégories d'objets :
  - ➢ **Constante** : c'est un objet dont la valeur est ***invariable***
  - ➢ **Variable** : c'est un objet qui ***peut varier*** durant le déroulement d'un algorithme .



Les Constantes

Déclaration

Const *nom_constante* **=** *Valeur*

Exemples

```
Algorithme exemple_const;
   Const    pi = 3.14
            cent = 100
            Lettre = 'M'
            Titre = 'Résultat :'
Début
   ...
   ...
Fin.
```

Les Objets

Déclaration

Exemples

PASCAL

Const nom_constante **=** Valeur

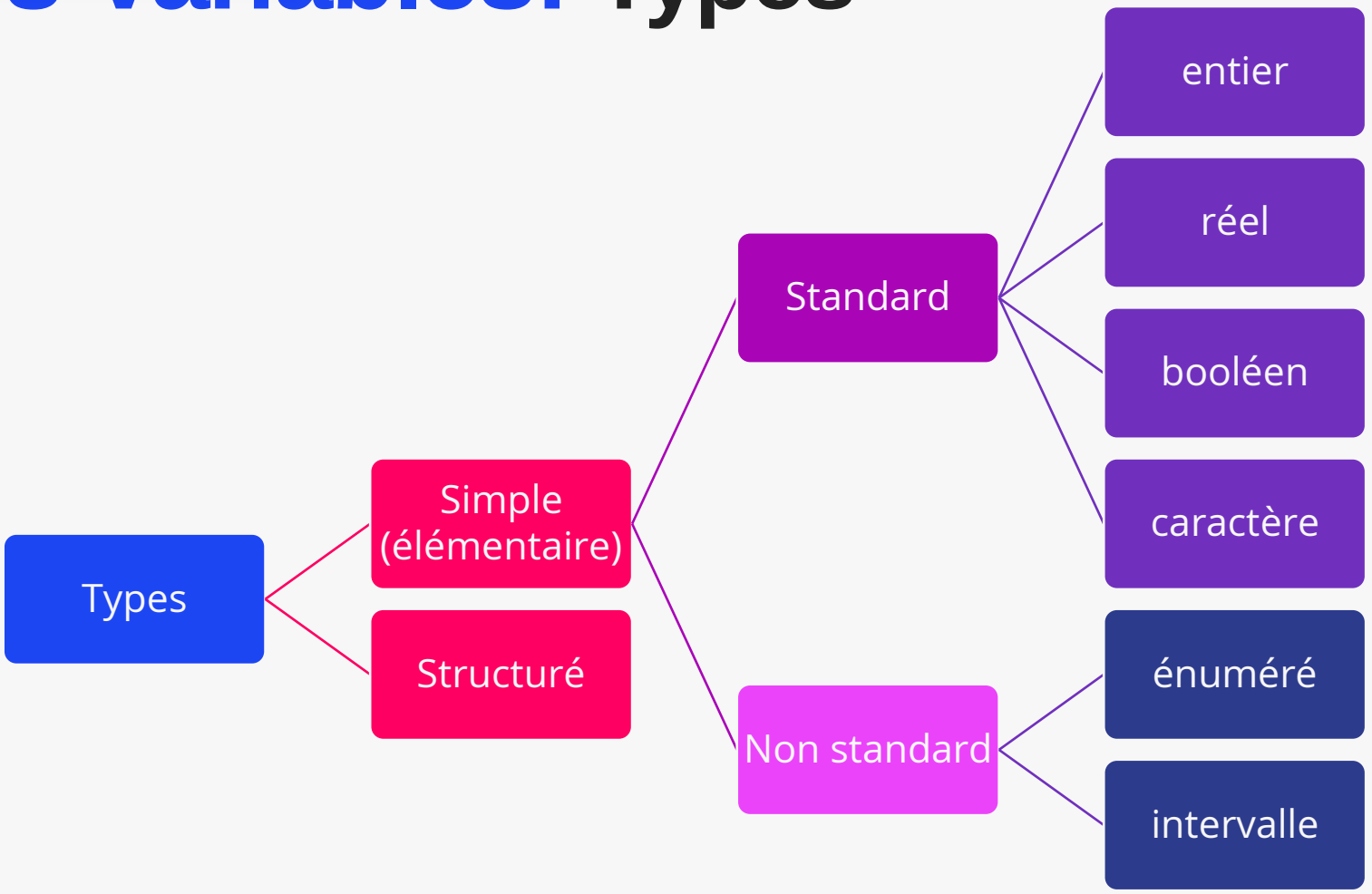
```pascal
program Exemple_Const;

const   pi = 3.14;
        cent = 100;
        Lettre = 'M';
        Titre = 'Résultat :';

var     ...

begin
    //...

end.
```

C

#define NOM_CONST valeur

```c
#include <stdio.h>

#define PI 3.14
#define CENT 100
#define LETTRE 'M'
#define TITRE 'Résultat :'

int main (){

    /*
          ...
    */
    return 0;
}
```

Les Variables: Types



Les Variables

Déclaration

Var *nom_variable* **: Type**

Exemples

```
Algorithme exemple_vars;
   Var      C : caractere
            N, i, j, jour, mois: entier
            x, y, racine : Réel
            trouver= booléen
Début
   ...
   ...
Fin.
```

Les Types Numériques

- **ENTIER**: c'est l'ensemble des entier relative.
- **RÉEL** : C'est l'ensemble des nombre ayant une partie fractionnelle.

Type	Octets	Plage
Byte (octet)	1	0 à 255
Entier simple	2	-32 768 à 32 767
Entier long	4	-2 147 483 648 à 2 147 483 647
Réel simple	4	$-3{,}40\times10^{38}$ à $-1{,}40\times10^{45}$ pour les valeurs négatives $1{,}40\times10^{-45}$ à $3{,}40\times10^{38}$ pour les valeurs positives
Réel double	8	$1{,}79\times10^{308}$ à $-4{,}94\times10^{-324}$ pour les valeurs négatives $4{,}94\times10^{-324}$ à $1{,}79\times10^{308}$ pour les valeurs positives

Les Types Numériques

Déclaration

```
Var     nom_variable1: Entier
        nom_variable2: Réel
```

Exemples

```
Algorithme  exemple_vars_num;
   Var      N, i, j, jour, mois: entier
            x, y, racine : Réel

Début
   ...
   ...
Fin.
```

Les Types Numériques

Déclaration

PASCAL

ENTIER : byte, shortint, integer, longint
REEL : real, double, extended

C

ENTIER : short, int, long, double long
REEL : float, double, long double

Exemples

```pascal
program Exemple_Const;

var     x, y, z : Integer;
        mois, annee : Integer;
        age : ShortInt;
        longueur, largeur : Integer;
        surface : Integer;
        totale : Double;

begin
    //...


end.
```

```c
#include <stdio.h>

int main (){
    int x, y, z;
    int mois, annee;
    short age;
    float longueur, largeur;
    float surface;
    double totale;

    /*
          ...
    */
    return 0;
}
```

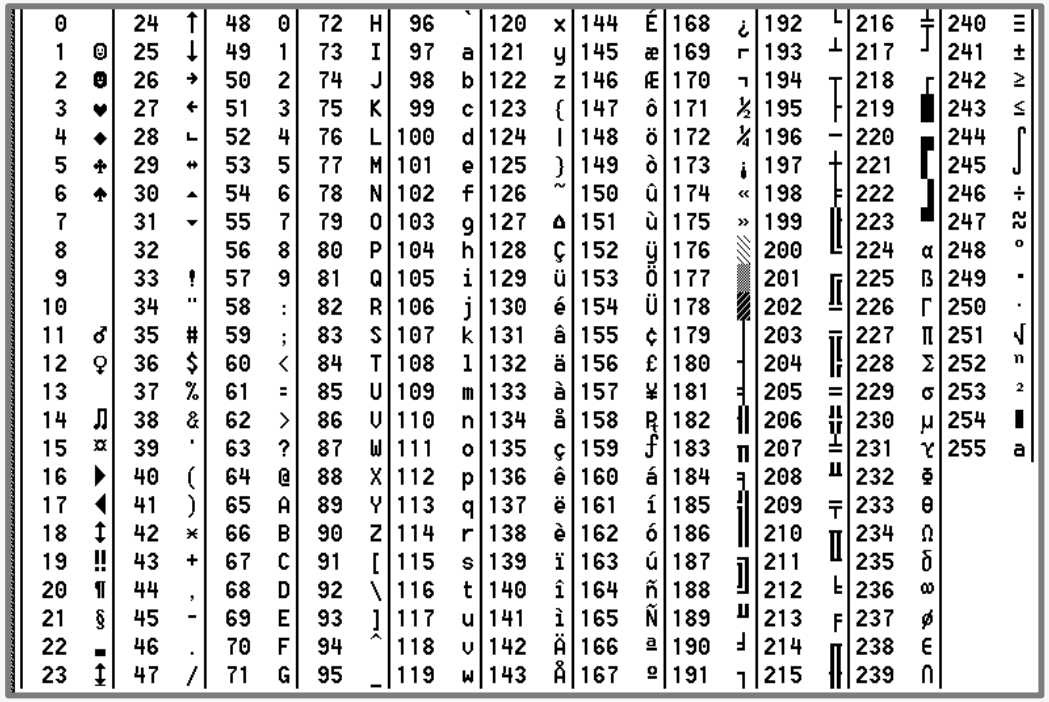
Les Types Alphanumérique

- **CARACTÈRE** : il correspond à un seul caractère (selon le système). Le jeux de caractères peut varier.
  - Se met entre deux guillemets simples.
  - Il comprend l'ensemble des caractères alphabétiques, numériques, signes de ponctuation, signes spéciaux, l'espace (blanc), caractère vide …

- **CHAINE DE CARACTÈRES** : C'est un ensemble (groupe) de caractères.

Les Types Alphanumérique

- Chaque caractère possède un code ASCII
- La code ASCII est un tableau de 0 à 255 caractères qui contient : des lettres minuscules et majuscules, des chiffres, des ponctuations, symboles spéciaux et graphiques …
- ***Fonctions manipulant des caractères :***
  - ***CHR(x):*** retourne le Caractère correspondant au code ASCII ***x***
  - ***ORD(c):*** retourne le code ASCII correspondant au Caractère ***c***



Code ASCII

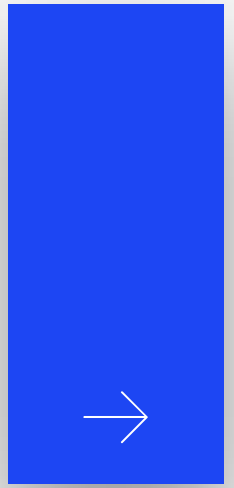
Les Types Alphanumérique

Déclaration

```
Var     nom_variable1: Caractère
        nom_variable2: Chaine
```

Exemples

```
Algorithme  exemple_vars_aplha;
   Var      c, aplha: caractère
            jour, mois, nom: Chaine

Début
   ...
   ...
Fin.
```



Les Types Alphanumérique

Les Objets

Déclaration

PASCAL

Caractère : Char
Chaines de Caractère : String

C

Caractère : char
Chaines de Caractère : tableau de caractères

Exemples

```pascal
program Exemple_Const;

var     c : Char;
        mot: String;
        nom, prenom : String[25];

        code : Integer;
begin
    //...
    c := Chr(82);
    WriteLn('c = ',c); // Affiche : c = R

    code := Ord('T');
    WriteLn('code = ',code); // code = 84

    //...

end.
```

```c
#include <stdio.h>

int main (){
    char c;
    char mot[];
    char nom[25], prenom[25];

    //
    c = code;
    printf("c = %c", c); // Affiche : c = R

    c = 'T';
    printf("c = %d", c);
    /* Affiche : c = 84 */


    //
    return 0;
}
```

Le Type Booléen

- **BOOLÉEN** : c'est l'ensemble des valeurs { VRAI, FAUX}.

Déclaration

```
Var      nom_variable1: Booléen
```

Exemples

```
Algorithme  exemple_vars_bool;
   Var      trouver, continue: entier

Début
   ...
   ...
Fin.
```

Les Types Alphanumérique

Les Objets

Déclaration

PASCAL

Booléen : Boolean

C

Booléen : nécessite un bibliothèque « **stdbool.h** » sinon, toutes valeur **non nulle** correspond à « **true** » et **0** correspond à « **false** »

Exemples

```pascal
program Exemple_Const;

var     trouver : Boolean;


begin
    //...
    trouver := true;
    WriteLn('trouver = ', trouver);
    // Affiche : trouver = TRUE

    //...

end.
```

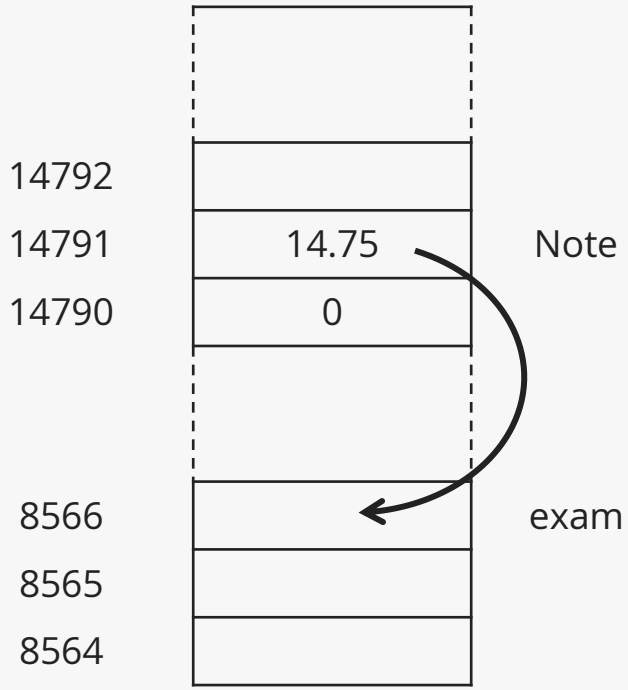
```c
#include <stdio.h>

int main (){

    int trouver = 0;

    if (trouver){
        printf("trouver est VRAI");
    }else{
        printf("trouver est FAUX");
    }

    return 0;
}
```

Les instructions

De base

L'affectation

- ✓ Son rôle consiste à ***affecter*** (***donner***, ***attribuer***) **une valeur** à un objet
- ✓ Cette valeur peut être :
  - ➢ Une constante
  - ➢ La valeur d'un autre objet
  - ➢ Le résultat d'une expression



Syntaxe

variable ← *expression*

exam ← Note

L'affectation

- ✓ Double **rôles** de l'affectation :
  1) Calculer et évaluer l'expression à ***droite*** de symbole : ←
  2) Affectation et rangement du résultat dans l'objet à ***gauche*** du symbole ←

Exemples

```
Algorithme  exemple_affect;
   Var      n, m, l: entier
Début
   ...
   n ← 10
   m ←  n
   l ← n*2 + m*3
   ...
Fin.
```

L'affectation

Déclaration

PASCAL

variable := expression

C

variable = expression

Exemples

```pascal
program Exemple_Const;

var     a, b, c : Integer;


begin
    //...
    a := 5;
    b := a + 3;
    c := a * b;

    WriteLn('a = ', a); // a = 5
    WriteLn('b = ', b); // b = 8
    WriteLn('c = ', c); // c = 40

    //...

end.
```

```c
#include <stdio.h>

int main (){

    int a, b, c;

    a = 5;
    b = a + 3;
    c = a * b;

    printf("a = %d", a); // Affiche : a = 5
    printf("b = %d", b); // Affiche : b = 8
    printf("c = %d", c); // Affiche : c = 40

    return 0;
}
```

Les expressions

- ✓ Une ***expression*** est un ensemble de valeurs (***opérandes***) , reliées par des ***opérateurs***, et équivalent à une seule valeur.
- ✓ Un ***opérateur*** est un signe qui relie deux valeurs, pour produire un résultat.

Syntaxe

> *expression = opérande* ***opérateur*** *opérande*

Les expressions : arithmétiques

- ✓ ***Opérandes :*** **entier**, **réel**.
- ✓ ***Opérateur*** :

+ : addition

- : soustraction

***** : multiplication

/ : division

DIV: division entière (quotient)

MOD: Reste de la division entière (modulo)

^ : puissance.

Les expressions : arithmétiques

Les Objets

Déclaration

PASCAL

Opérateurs : **+**, **-** , ***** , **/** , **DIV** (Division entière), **MOD** (Modulo)

C

Opérateurs : **+**, **-** , ***** , **/** (Division entière) , **%** (Modulo), **++** (Incrément),**--** (Décrément)

Exemples

```pascal
program Exemple_Const;

var     a, b, c : Integer;


begin
    //...
    a := 15;
    b := a + 1;
    c := b MOD 3;

    WriteLn('a = ', a); // a = 15
    WriteLn('b = ', b); // b = 16
    WriteLn('c = ', c); // c = 1

    //...

end.
```

```c
#include <stdio.h>

int main (){

    int a, b, c;

    a = 15;
    b = a++;
    c = b % 3;

    printf("a = %d", a); // Affiche : a = 15
    printf("b = %d", b); // Affiche : b = 16
    printf("c = %d", c); // Affiche : c = 1

    return 0;
}
```

Les expressions : alphanumérique

- ✓ ***Opérandes :*** **caractère, chaines de caractères.**
- ✓ ***Opérateur*** :

  + : concaténation

Exemples

```
Algorithme  exemple_alpha;
   Var      c1, c2: chaine
Début
   ...
   c1 ← "nom" + "prénom"  // résultat : "nomprénom"
   c2 ← "nom" + " " + "prénom"  // résultat : "nom prénom"
   ...
Fin.
```

Les expressions : alphanumériques

Les Objets

Déclaration

Exemples

PASCAL

Opérateurs : +

```pascal
program Exemple_Const;

var     nom, prenom : String[25];
        titre : String;

begin
    //...
    nom := 'BENALI';
    prenom := 'Karim';

    titre := nom + prenom;
    WriteLn('titre = ', titre);
         //titre = BENALIKarim
    titre := nom + " " + prenom;
    WriteLn('titre = ', titre);
         //titre = BENALI Karim

end.
```

C

Opérateurs :

```c
#include <stdio.h>

int main (){


    return 0;
}
```

Les expressions : logiques

- ✓ ***Opérandes :*** **booléen (vrai, faux)**.
- ✓ ***Opérateur*** :

  NON : La négation

  ET : le « ET » logique

  OU : le « OU » logique

Les expressions : logiques

Déclaration

PASCAL

Opérateurs : **AND**, **OR**, **NOT**

C

Opérateurs : **&&** (AND), **||** (OR), **!** (NOT)

Exemples

```pascal
program Exemple_Const;

var     nom, prenom : String[25];
        titre : String;

begin
    //...

end.
```

```c
#include <stdio.h>

int main (){


    return 0;
}
```

Les expressions : relationnelles

- ✓ ***Opérandes :*** **entier**, **réel, caractère**, **chaines de caractères**.
- ✓ ***Opérateur*** :

< : inférieur

> : supérieur

= : égale

<= : inférieur ou égale

>= : supérieur ou égale

<> : différent

Les expressions : logiques

Déclaration

PASCAL

Opérateurs : <, <=, >, >=, =, <>

C

Opérateurs : <, <=, >, >=, ==, !=

Exemples

```pascal
program Exemple_Const;

var     nom, prenom : String[25];
        titre : String;

begin
    //...

end.
```

```c
#include <stdio.h>

int main (){


    return 0;
}
```

Les entrées / sorties : Lecture

- ✓ Permet de ***fournir*** des valeurs venant de l'extérieur
  - ➢ Les valeurs sont entrées au ***clavier***
  - ➢ Les P1, P2, …Pn sont des variables

Syntaxe

Lire *(P1, P2, …, Pn)*

Exemples

```
Algorithme  exemple_affect;
   Var      n, m, l: entier
Début
   Lire (n, m)
   Lire (l)
   …
Fin.
```

La Lecture

Déclaration

PASCAL

Syntaxe: **Read**, **ReadLn**

C

Syntaxe: **scanf** (<stdio.h)

Exemples

```pascal
program Exemple_Const;

var     a1, a2 : Integer;
        b : Real;
        c : Char;

begin
    //...
    ReadLn(a1, a2);
    ReadLn(b);
    Read(c);

    //...

end.
```

```c
#include <stdio.h>

int main (){

    int a;
    float b;
    char c;

    scanf("%d", &a);
    scanf("%f", &b);
    scanf("%c", &c);

    //...

    return 0;
}
```

Les entrées / sorties : Ecriture

- ✓ Permet de ***restituer*** (afficher) les résultats d'un algorithme
  - ➢ Les valeurs sont affiché sur l'***écran***
  - ➢ Les *E1, E2, ...En* peuvent être: des variables, des chaines ou des expressions

Syntaxe

Ecrire *(E1, E2, ..., En)*

Exemples

```
Algorithme  exemple_affect;
   Var      n, m, l: entier
Début
   Ecrire(n, m)
   Ecrire ('la somme =', n+m)
   ...
Fin.
```

L'écriture

Les Objets

Déclaration

PASCAL

Syntaxe: **Write**, **WriteLn**

C

Syntaxe: **printf** (<stdio.h)

Exemples

```pascal
program Exemple_Const;

var     a1, a2 : Integer;
        b : Real;
        c : Char;

begin
    //...

    WriteLn('Hello, World');
    WriteLn(a1, a2);
    WriteLn(b);
    WriteLn(c);
    WriteLn('Résultat est ', a1+a2);

    //...

end.
```

```c
#include <stdio.h>

int main (){

    int a;
    float b;
    char c;

    //...
    printf("Hello, World");
    printf("%d", a);
    printf("%f", b);
    printf("%c", c);
    printf("Résultat est %d", a);
    //...

    return 0;
}
```

Ministère de l'enseignement supérieur et de la recherche scientifique
Université Djillali BOUNAAMA - Khemis Miliana (UDBKM)
Faculté des Sciences et de la Technologie
Département de Mathématiques et d'Informatique

Chapitre 2

Algorithme séquentiel simple

MI-L1-UEF121 : Algorithmiques et Structures de Données I

Noureddine AZZOUZA

n.azzouza@univ-dbkm.dz