

---

# Algorithmique & programmation

---

## Chapitre 2 : Vecteurs en algorithmique

### Introduction

---

## La notion de vecteur

---

### □ Définition intuitive

- Collection de données de même type numérotées
  - On peut accéder à la première, à la troisième, à la dixième donnée

### □ Définition formelle

- **E** un ensemble de valeurs
- **I** un intervalle sur  $\mathbb{N}$  ;  $I = [m..n]$ ,  $n \geq m$  (ensemble des indices)
- le vecteur comporte  $n-m+1$  éléments
  
- Un vecteur est une **application V** de **I** dans **E**

# Ensemble des indices

- Notation : **[inf..sup]**
  - **inf** : borne inférieure (indice le plus petit)
  - **sup** : borne supérieure (indice le plus grand)
- En algorithmique, on utilisera souvent **[1..n]**
- L'ensemble des indices peut être vide
  - si  $n=0 \rightarrow [1..0]$  est vide
  - si  $\text{inf} > \text{sup} \rightarrow [\text{inf}..\text{sup}]$  est vide
  - Dans ce cas, le vecteur est vide
- Le **nombre d'élément** dans l'ensemble des indice donne la **taille du vecteur**
  - si  $I = [1..n]$  la taille du vecteur est  $n$

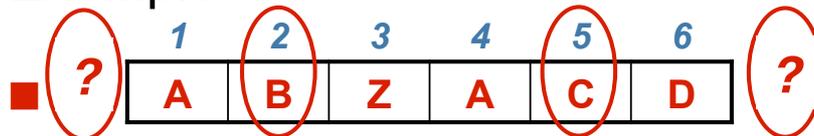
# Notations

- Élément d'un vecteur  $V$ 
  - $V[i]$  avec  $i \in [1..n]$  ( $i^{\text{ème}}$  élément du vecteur)
- Vecteur
  - $V[1..n]$  ou  $V$
- Liste des éléments d'un vecteur
  - $(V[1], V[2], V[3], \dots, V[i], \dots, V[n])$
- Vue graphique
  - $V$ 

1	2	3	4	...	$i$	...	$n$
$V[1]$	$V[2]$	$V[3]$	$V[4]$	...	$V[i]$	...	$V[n]$
  - $V[i]$  = valeur de l'élément d'indice  $i$  de  $V$

# Manipulation

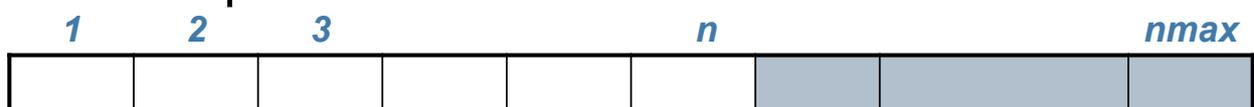
- Lecture d'un élément de vecteur →  $A := V[i]$  ;
- Écriture dans un vecteur →  $V[i] := A$  ;
- Attention :  $V[i]$  est indéfini si  $i \notin [1..n]$
- Exemple



- $V[1..6] = ('A', 'B', 'Z', 'A', 'C', 'D')$
- $V[2] = 'B'$
- $V[0]$  est indéfini
- $V[5] = 'C'$
- $V[7]$  est indéfini

# Déclaration d'un vecteur

- **type vecteur** = tableau  $[1..nmax]$  de **t**
  - **t** : n'importe quel type simple, structuré ou vecteur
  - **nmax** : nombre maximum de données pour le type vecteur
    - Il faut réserver la place nécessaire au pire !
    - Gestion statique de la mémoire (place réservée)
- Déclaration
  - $V[1..n]$  : **vecteur**
- On peut utiliser n'importe quel vecteur  $V[1..n]$  tel que  $0 < n \leq nmax$



# Notion de sous-vecteur

---

- Restriction de  $V$  à un intervalle **consécutif** de  $[1..n]$
- Noté  $V[i..j]$ 
  - si  $j \geq i$ 
    - Sous-vecteur composé de  $V[i], V[i+1], \dots, V[j]$
  - si  $j < i$ 
    - Sous-vecteur vide
- Exemple
  - si  $V = (2, 8, 6, 4, 10, 0, 5, 3)$ , alors :
    - $V[1..8] = (2, 8, 6, 4, 10, 0, 5, 3)$
    - $V[3..7] = (6, 4, 10, 0, 5)$
    - $V[5..5] = (10)$
    - $V[6..4] = ()$

# Relation d'ordre

---

- La relation d'ordre sur l'ensemble  $E$  des valeurs de type simple d'un vecteur
  - notée par le symbole  $<$  quel que soit  $E$
- Dans les assertions
  - $a \in V[1..n] \Rightarrow \exists j \in [1..n] / a = V[j]$
  - $a \notin V[1..n] \Rightarrow \forall j \in [1..n] / a \neq V[j]$
  - $a < V[1..n] \Rightarrow \forall j \in [1..n] / a < V[j]$
  - $a > V[1..n] \Rightarrow \forall j \in [1..n] / a > V[j]$
  - ...

# Paramètre de type vecteur (somme)

**type** vecteur = tableau[1..nmax] de entier;

**fonction** somme (**d V[1..n] : vecteur**) : entier ;

**spécification**  $\{n \leq 0\} \rightarrow \{\text{résultat} = \Sigma V[1..n]\}$

s, i : entier ;

**debfunc**

...

**finfunc** ;

## Paramètre formel

- **V** : nom du vecteur manipulé dans **somme**
- **[1..n]** : intervalle des indices du vecteur  
on peut traiter V[1] ... V[n]
- **1** : borne inférieure des indices
- **n** : borne supérieure des indices

# Paramètre de type vecteur (somme)

**type** vecteur = tableau[1..nmax] de entier;

## Paramètre effectif

- **Mon\_Vect** : nom du vecteur pour lequel faire la **somme**
- **[1..m]** : intervalle des indices du vecteur  
on va traiter Mon\_Vect[1] ... Mon\_Vect[m]
- **1** : borne inférieure des indices
- **m** : borne supérieure des indices

**procédure** principale;

**Mon\_Vect** : vecteur ;

**début**

{initialisation de **Mon\_Vect**}

écrivertexte("la somme des éléments du vecteur est : " &

**somme**(**Mon\_Vect[1..m]**)) ;

**finproc** ;

# Paramètre de type vecteur (somme)

type vecteur = tableau[1..nmax] de entier;

fonction somme ( **d V[1..n] : vecteur** ) : entier ;

spécification  $\{n \leq 0\} \rightarrow \{\text{résultat} = \Sigma V[1..n]\}$

s, i : entier ;

debfunc

...

finfunc ;

procédure principale;

**Mon\_Vect** : vecteur ;

début

{initialisation de **Mon\_Vect**}

écrivertexte("la somme des éléments du vecteur est : " &

**somme**(**Mon\_Vect**[1..m]));

finproc ;

## Appel

– La fonction somme est exécutée sur **Mon\_Vect**

– **V** prend la valeur de **Mon\_Vect**

– **[1..n]** prend la valeur de **[1..m]**

• **1** ← 1

• **n** ← m



# Paramètre de type vecteur (somme)

Si **Mon\_Vect** est défini sur l'intervalle **[1..4]**

■ tel que **Mon\_Vect**[1..4]=(2, 4, 6, 8)

alors **somme**( **Mon\_Vect**[1..4] ) retourne 20

Explication

■ lors de l'appel de

■ fonction **somme** ( **d V[1..n] : vecteur** ) : entier ;

**V** prend la valeur de **Mon\_Vect**

**1** ← 1

**n** ← 4

■ **Somme** produit comme résultat

**Mon\_Vect**[1] + **Mon\_Vect**[2] + **Mon\_Vect**[3] + **Mon\_Vect**[4]

# Parcours complet d'un vecteur

---

- de gauche à droite
  - indices croissants
  - *parcours complet*

$i := 1$  ;

**tantque**  $i \leq n$  **faire**

traiter (V[i]) ;

$i := i + 1$  ;

**finfaire** ;

ou

**pour**  $i := 1$  **haut**  $n$  **faire**

traiter (V[i]) ;

**finfaire** ;

- de droite à gauche
  - indices décroissants
  - *parcours complet*

$i := n$  ;

**tantque**  $i \geq 1$  **faire**

traiter (V[i]) ;

$i := i - 1$  ;

**finfaire** ;

ou

**pour**  $i := n$  **bas**  $1$  **faire**

traiter (V[i]) ;

**finfaire** ;