INFO-H-100
Séances d'exercices 7

Les tableaux (1) :

Les vecteurs

Les tableaux

- Collection indexée d'éléments de même type.

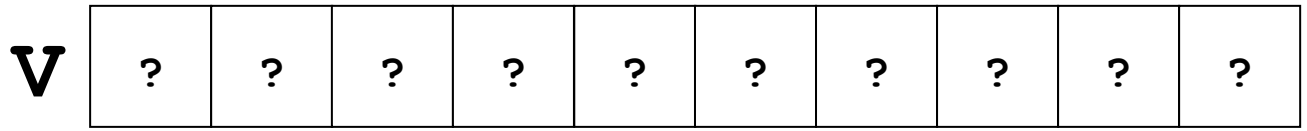
- Vocabulaire :
  - Un **vecteur** est un tableau à une dimension.
  - Une **matrice** est un tableau à deux dimensions.

- Par exemple, un vecteur de 10 entiers (`int`).

5	9	-7	6	6	-4	98	8	3	5

Déclaration d'un vecteur

- Le type des éléments
- Le nom du tableau
- Le [nombre] d'éléments (fixé par le programmeur)
- Exemples :
  - `int V[10];` déclare un vecteur de 10 entiers
  - `bool bools[5];` déclare un vecteur de 5 booléens

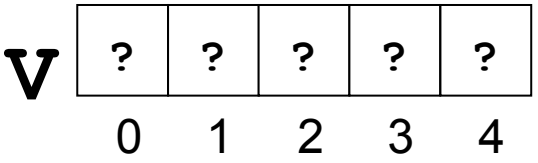


- Comme pour une variable, un tableau doit être initialisé après avoir été déclaré.

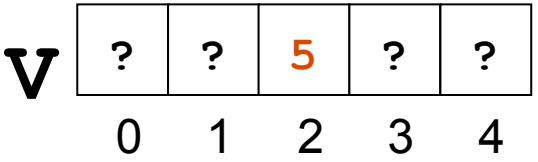
Accès aux éléments

- Les cases d'un vecteur de taille *n* sont numérotées de *0* à *n-1*.
- On accède à une case d'un vecteur via le nom du tableau et l'indice de la case.

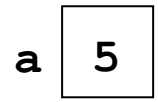
```
int V[5];
```




```
V[2] = 5;
```




```
int a = V[2];
```



Remarques importantes

- On ne peut pas accéder à une case qui n'existe pas :

```
bool B[5];
…
cout << B[7]; //INTERDIT !
cout << B[5]; //INTERDIT AUSSI
```

- Un vecteur a une taille fixe et constante décidée par le programmeur :

```
int i;
cin >> i;
int V[i]; //INTERDIT !
```

Remarques importantes

- On ne peut pas manipuler un vecteur en lui-même, on doit manipuler ses cases :

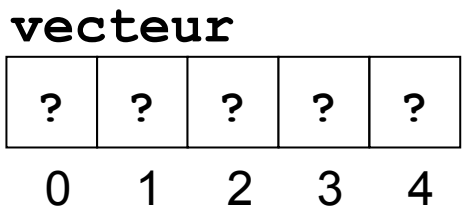
```
int V[10];
int W[10];
(…)
V = W;          //INTERDIT !
W = 3;          //INTERDIT !
if(V==W)        //INTERDIT !
{
        …
}
```

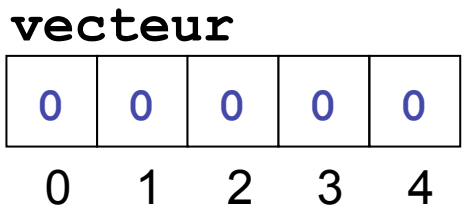
Exemple : initialisation d'un vecteur

```
#include <iostream>
using namespace std;

int main()
{
    int vecteur[5];

    for(int i=0; i<5; i++)
    {
        vecteur[i] = 0;
    }
    return 0;
}
```





Vecteurs et fonctions

- Un tableau est toujours passé par référence à une fonction.
- Il ne faut pas mettre de & dans la déclaration de la fonction.
- Quand on passe un vecteur à une fonction, on passe deux paramètres : le vecteur et sa taille

```
void print(int V[], int size);
```

Vecteurs et fonctions

```
#include <iostream>
using namespace std;

void init(int V[], int size);
void print(int V[], int size);

int main()
{
   int vecteur[5];
   init(vecteur, 5);
   print(vecteur, 5);
   return 0;
}
```

```
void init(int V[], int size)
{
  for(int i=0; i<size; i++)
  {
    V[i] = 0;
  }
}

void print(int V[], int size)
{
  for(int i=0; i<size; i++)
  {
    cout << V[i];
  }
}
```

Fonction qui produit un vecteur

- Pour produire un vecteur, il faut le passer en paramètre à la fonction qui va le remplir
- Ne jamais faire un **`return`** d'un vecteur : ~~**`return V;`**~~
- Exemple :

```
#include <iostream>
using namespace std;

void copie(int V[], int W[], int size);

int main()
{
    int V1[5];
    int V2[5];
    (…)
    copie(V1, V2, 5);
    /* ici le contenu de V1 est identique
       à celui de V2 */
    return 0;
}
```

```
/* copie les size premiers éléments de
   V dans W

   précondition : la taille de W est
   plus grande ou égale à size
*/
void copie(int V[], int W[], int size)
{
  for(int i=0; i<size; i++)
  {
    W[i] = V[i];
  }
}
```

Exercices

- 69, 70, 71, 72, 78, 75