Tableaux

Tableaux à taille fixe

Définition

> Simplement Un tableau à taille fixe est une séquence d'éléments d'une taille définie

- Tout est alloué d'un seul bloc ➡ les cases sont contiguës en mémoire
- Le premier index démarre à 0.
- La taille est définitive, pour agrandir, il faut allouer un autre tableau ou utiliser une autre technique.
- Le contenu sera toujours initialisé à `0, "", ...`.

Syntaxe

```
var nom[taille]type
```

```
var tab[5]int
t[3] = 12
```

Déclaration rapide

```
odds := [4]int{1, 3, 5, 7}
pair := [4]int{2, 4} // [2, 4, 0, 0]
```

Affichage

```
var names [3]string
names[0] = "Bob"
names[2] = "Alice"

fmt.Printf("name[2]=%v\n", names[2])
```

```
fmt.Printf("names=%v (len=%d)\n", names, len(names))
```

Tableaux dynamiques (Slice)

Définition

Tableau de taille dynamique

- Slice ⇒ Tranche ◫
- Un Slice représente une tranche d'un tableau.
- Un Slice est une "vue" sur le tableau sous-jacent
- Modifier le slice ➡ modifier le tableau

Syntaxe

```
s := make([]type, taille, capacité)
```

- **Taille** : nombre d'éléments du slice
- **Capacité** (facultatif) : nombre d'éléments du tableau

```
s := make([]int, 3)
s[0] = -3
len(s) // 3
cap(s) // 3
```

Réallocation

```
s := make([]int, 3)
s = append(s, 12)
len(s) // 4
cap(s) // 6
```

Explication :

- Si on dépasse la taille du tableau
- Un nouveau tableau est alloué, de capacité doublée

Sous-tableaux

```
letters := []string{"g", "o", "l", "a", "n", "g"}
fmt.Printf("%v \n", letters)

// subslicing
sub1 := letters[:2]
sub2 := letters[2:]
fmt.Printf("%v\n", sub1) // ?
fmt.Printf("%v\n", sub2) // ?
```

Référence des sous tableaux

Que se passe-t-il si on fait ça ?

```
letters := []string{"g", "o", "l", "a", "n", "g"}
sub1 := letters[:2]
sub2 := letters[2:]

sub2[0] = "UP"
fmt.Printf("%v\n", sub2) // ?
fmt.Printf("%v\n", letters) // ?
```

Et là ?

```
letters := []string{"g", "o", "l", "a", "n", "g"}
sub2 := letters[2:]

subCopy := make([]string, len(sub2))
copy(subCopy, sub1)
subCopy [0] = "UP"
fmt.Printf("%v\n", subCopy) // ?
fmt.Printf("%v\n", letters) // ?
```

Revision #1
Created 2022-05-09 19:06:55 UTC by Noé Larrieu-Lacoste
Updated 2022-05-09 19:09:30 UTC by Noé Larrieu-Lacoste