Structures

> *Dans cette série d'exercices, vous allez créer un magasin de jeux vidéo en ligne de commande.*

Exercice 1

1. Déclarez les structures suivantes :
   - **item** : **id** (`int`), **name** (`string`), **price** (`int`)
   - **game** : intégrez la structure **`item`** , **genre** (`string`)
2. Créez une tranche (`slice`) de jeux vidéo (`game`) en utilisant les données suivantes :

id	name	price	genre
1	god of war	50	action adventure
2	x-com 2	30	strategy
3	minecraft	20	sandbox

3. Affichez tous les jeux vidéo.

```
package main

func main() {
    // Votre code ici
}
```

Résultat attendu :

```
Le magasin propose 3 jeux.

#1   : "god of war"    (action adventure)   50€
#2   : "x-com 2"       (strategy)           40€
#3   : "minecraft"     (sandbox)            20€
```

Indice :
Regarder les codes de formatage d'espacement pour les chaînes de caractères (https://cheatography.com/fenistil/cheat-sheets/go-fmt-formattings/).

Exercice 2

1. Utilisez l'interface **`bufio.Scanner`** pour ajouter une fonctionnalité d'interaction avec l'utilisateur dans votre programme.
   - Les utilisateurs devraient pouvoir lister les jeux ou rechercher des jeux par ID.
2. Affichez les commandes disponibles.
3. Implémentez la commande **quit** : Permet de quitter la boucle.
4. Implémentez la commande **list** : Permet d'afficher la liste de tous les jeux.

```
package main

func main() {
    // Utilisez votre solution de l'exercice précédent
```

```
    // pour créer une interaction avec l'utilisateur ici.
}
```

Indice :
Regarder la documentation de l'interface ** `bufio.Scanner` ** (https://golang.org/pkg/bufio/#Scanner).

```go
package main

import (
    "bufio"
    "fmt"
    "os"
)

func main() {
    // bufio.Scanner example
    in := bufio.NewScanner(os.Stdin)
    fmt.Println("Please enter your name: ")
    in.Scan()
    fmt.Printf("Hello %q\n", in.Text())
}
```

Résultat attendu :

```
$ go run main.go
Le magasin propose 3 jeux.

Commandes :
> list   : liste tous les jeux
> quit   : quitte

Votre choix : list

#1   : "god of war"    (action adventure)   50€
#2   : "x-com 2"       (strategy)           40€
#3   : "minecraft"     (sandbox)            20€

Commandes :
> list   : liste tous les jeux
> quit   : quitte

Votre choix : dd

Commande inconnue.

...

Votre choix : quit

Au revoir !
```

Exercice 3

1. Implémentez la commande **id** : Permet de rechercher un jeu par ID.
    - Lorsqu'un utilisateur tape : `id 2` , il devrait afficher uniquement le jeu avec l'identifiant 2.

2. Avant la boucle, créez un dictionnaire (`map`) qui associe les ID des jeux au jeu correspondant.
3. Gérez les erreurs :

```
id
ID invalide.

id HEY
ID invalide.

id 10
Jeu introuvable.

id 1
#1   : "god of war"    (action adventure)   50€

id 2
#2   : "x-com 2"       (strategy)           40€
```

Indice :
Regardez ce que font les fonctions :

- **`strings.Fields`** (https://golang.org/pkg/strings/#Fields)
- **`strconv.Atoi`** (https://golang.org/pkg/strconv/#Atoi)

Résultat attendu :

```
$ go run main.go
Le magasin propose 3 jeux.

Commandes :
> list   : liste tous les jeux
> id N   : affiche le jeu d'identifiant N
> quit   : quitte

Votre choix : id

ID invalide.

...

Votre choix : id S

ID invalide.

...

Votre choix : id 1

#1   : "god of war"    (action adventure)   50€

...

Votre choix : id 10

Jeu introuvable.

...
```

```
Votre choix : quit

Au revoir !
```