Structures de contrôle

Exercice 1

```
package main

import "fmt"

func main() {
    age := 10

    if age > 60 {
        fmt.Println("Getting older")
    } else if age > 30 {
        fmt.Println("Getting wiser")
    } else if age > 20 {
        fmt.Println("Adulthood")
    } else if age > 10 {
        fmt.Println("Young blood")
    } else {
        fmt.Println("Booting up")
    }
}
```

Exercice 2

```
package main

import "fmt"

func main() {
    isSphere, radius := true, 200

    if isSphere && radius >= 200 {
        fmt.Println("C'est une grosse sphère.")
    } else {
        fmt.Println("Je ne sais pas.")
    }
}
```

Exercice 3

```
package main

import (
    "fmt"
    "os"
)

func main() {
    // Récupérez les arguments depuis la ligne de commande
```

```
    args := os.Args[1:]

    // Utilisez len(args) pour déterminer le nombre d'arguments et affichez le résultat attendu
en conséquence.
    switch len(args) {
    case 0:
        fmt.Println("Donnez-moi des arguments")
    case 1:
        fmt.Printf("Il y en a un : %q\n", args[0])
    case 2:
        fmt.Printf("Il y en a deux : %q\n", args[0]+" "+args[1])
    default:
        fmt.Printf("Il y a %d arguments\n", len(args))
    }
}
```

Exercice 4

```
func main() {
    args := os.Args
    if len(args) != 2 {
        fmt.Println("Donnez-moi la magnitude du tremblement de terre.")
        return
    }

    richter, err := strconv.ParseFloat(args[1], 64)
    if err != nil {
        fmt.Println("Je n'ai pas pu obtenir ça, désolé.")
        return
    }

    var desc string

    switch r := richter; {
    case r < 2:
        desc = "micro"
    case r >= 2 && r < 3:
        desc = "très mineur"
    case r >= 3 && r < 4:
        desc = "mineur"
    case r >= 4 && r < 5:
        desc = "léger"
    case r >= 5 && r < 6:
        desc = "modéré"
    case r >= 6 && r < 7:
        desc = "fort"
    case r >= 7 && r < 8:
        desc = "majeur"
    case r >= 8 && r < 10:
        desc = "énorme"
    default:
        desc = "massif"
    }
```

```
    fmt.Printf("%.2f est %s\n", richter, desc)
}
```

Exercice 5

```
package main

import (
    "fmt"
    "os"
    "strings"
    "time"
)

func main() {
    if len(os.Args) != 2 {
        fmt.Println("Donnez moi un nom de mois")
        return
    }

    annee := time.Now().Year()
    bissextile := annee%4 == 0 && (annee%100 != 0 || annee%400 == 0)

    jours, mois := 28, os.Args[1]

    switch strings.ToLower(mois) {
    case "avril", "juin", "septembre", "novembre":
        jours = 30
    case "janvier", "mars", "mai", "juillet",
        "aout", "octobre", "decembre":
        jours = 31
    case "fevrier":
        if bissextile {
            jours = 29
        }
    default:
        fmt.Printf("%q n'est pas un mois.\n", mois)
        return
    }

    fmt.Printf("%q a %d jours.\n", mois, jours)
}
```

Exercice 6

```
package main

import (
    "fmt"
)

func main() {
    var sum int
```

```
    for i := 1; i <= 10; i++ {
        sum += i
    }
    fmt.Println("Sum:", sum)
}
```

Exercice 7

```
func main() {
    i := 1
    sum := 0

    for ; i <= 10; i++ {
        if i%2 != 0 {
            continue
        }
        fmt.Print(i)
        if i != 10 {
            fmt.Print(" + ")
        }
        sum += i
    }
    fmt.Printf(" = %d\n", sum)
}
```

Exercice 8

```
package main

import (
    "fmt"
    "os"
    "strconv"
)

func main() {
    args := os.Args

    if len(args) != 2 {
        fmt.Println("Donnez-moi la taille de la table")
        return
    }

    taille, err := strconv.Atoi(args[1])
    if err != nil || taille <= 0 {
        fmt.Println("Mauvaise taille")
        return
    }

    // imprimer l'en-tête
    fmt.Printf("%5s", "X")
    for i := 0; i <= taille; i++ {
        fmt.Printf("%5d", i)
```

```
    }
    fmt.Println()

    for i := 0; i <= taille; i++ {
        // imprimer l'en-tête vertical
        fmt.Printf("%5d", i)

        // imprimer les cellules
        for j := 0; j <= taille; j++ {
            fmt.Printf("%5d", i*j)
        }
        fmt.Println()
    }
}
```