



ゴー・ピスチーナ 02へ

概要：この文書は、Go Piscine @ 42Tokyo の Go 02 モジュールのための主題である。

内容

I	取扱説明書	2
II	エクササイ 反復算術式 ズ00:	3
さん じゅ うろ く	練習問題01: さいきょうはかいほう	4
点滴	練習問題02: 反復力	6
V	練習問題03: 再帰的電力	7
六価	練習問題04: フィボナッチ	9
VII	練習問題 05: 自乗	11
VIII	練習問題06: イズプライム	12
IX	練習問題 07: 次点	14
X	練習問題 08: 八十帖	16

第一章 説明

- 噂を鵜呑みにせず、このページだけを参考にしてください。
- ご注意ください。この文書は、提出の1時間前までに変更される可能性があります。
- これらのエクササイズは、簡単なものから難しいものへと、難易度順に慎重に並べられています。簡単なエクササイズが完璧に機能しない場合、より難しいエクササイズを成功裏に完了させることは考慮されません。
- ファイルやディレクトリに適切なパーミッションが設定されていることを確認してください。
- 演習のたびに提出手続きを行う必要があります。
- 演習は、クラスメートによってチェックされ、採点されます。
- お題で指定されたファイル以外のファイルをディレクトリに残すことはできません。
- 質問がありますか？右側の仲間に質問してください。そうでなければ、左側の仲間に尋ねてください。
- あなたのリファレンスガイドは、Google / man / the Internet / ... と呼ばれています。
- 例題を十分に検討すること。例題に明示されていない内容が要求されることも大いにあり得ます...
- 他に明示的な情報が表示されない場合は、最新バージョンのGoを使用する必要があります。
- 各エクササイズのターンインディレクトリはこうになっているはずです。

```
ex[XX]の場合
|-- main.go
|-- ベンダー
    |-- ft
        |-- printrune.go
    |-- 水槽
        |-- [exercisename].go (エクササイズネーム)。
```

第二章

練習問題00：反復算法

	練習問題 00
	iterativefactorial
提出先ディレクトリ:	
ex00/ 提出するファイル	
:* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数:なし	

パラメータとして渡された `int` の階乗を返す反復関数を作成しなさい。

- エラー（取り得ない値やオーバーフロー）は0を返します。
- 期待される機能

```
func IterativeFactorial(nb int) int {  
      
}  
  
パッケージメイン  
  
輸入  
    "fmt"  
    $ go mod init ex00  
    $ go run .24  
    $  
func n  
    arg := 4  
    fmt.Println(piscine.IterativeFactorial(arg))  
    です。  
}
```

第三章

練習問題01：再帰的要素法

	練習問題01
	recursivefactorial
提出先ディレクトリ:	
ex01/ 提出するファイル	
:* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数:なし	

パラメータとして渡された`int`の階乗を返す再帰的関数を作成しなさい。

- エラー（取り得ない値やオーバーフロー）は0を返します。
- は、この演習では禁止されています。
- 期待される機能

```
func RecursiveFactorial(nb int) int {  
      
}  
  
パッケージメイン  
輸入  
    "fmt"  
    "piscine"  
)  
  
func main() {  
    arg := 4  
    fmt.Println(piscine.RecursiveFactorial(arg))  
    です。  
}
```

```
$ go mod init ex01  
$ go run .24  
$
```

第四章

Exercice02: 反復力

	演習02
	反復力
提出先ディレクトリ:	
ex02/ 提出するファイル	
:* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数:なし	

nbのべき乗の値を返す反復関数を書け.

- 負のべき乗は0を返します。オーバーフローは処理する必要がありません。
- 期待される機能

```
func IterativePower(nb int, power int) int {  
    }  
  
パッケージメイン  
輸入  
    // Ex02  
    $ go mod init ex02  
    $ go run .64  
    $  
  
func main() {  
    fmt.Println(piscine.IterativePower(4,3)) を実行。  
}
```

第五章

Exercice03 : 再帰的パワー

	演習03
再帰的電力	
提出先ディレクトリ :	
ex03/ 提出するファイル	
:* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数 :なし	

nbのべき乗の値を返す反復関数を書け.

- 負の累乗は0を返します。オーバーフローは処理する必要がありません。
- は、この演習では禁止されています。
- 期待される機能


```
func RecursivePower(nb int, power int) int {  
}  
  
パッケージメイン  
輸入  
    "fmt"  
    "piscine"  
)  
  
func main() {  
    fmt.Println(piscine.RecursivePower(4,3))を実行。  
}
```



```
$ go mod init ex03  
$ go run .64  
$
```

第六章

Exercise04: フィボナッチ

	演習04
	フィボナ
提出先ディレクトリ:	ッチ
ex04/ 提出するファイル	
.* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数:なし	

フィボナッチ数列の位置インデックスにある値を返す再帰的関数を書け.

- 最初の値はインデックス0である。
- 0、1、1、2、3...といった具合に並びます。
- 負のインデックスを指定すると、-1が返される。
- は、この演習では禁止されています。
- 期待される機能

```
func フィボナッチ(index int) int {  
}
```

- 使用方法

```
パッケージメイン
  
輸入
    "fmt"
)
$ go mod init ex04
$ go run .3
$
func main() {
    fmt.Println(piscine.Fibonacci(arg1))です。
}
```

第VII章 練習問

題05 : sqrt

	演習05
自乗	
提出先ディレクトリ:	
ex05/ 提出するファイル	
:* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数:なし	


パラメータとして渡された`int`の平方根が整数の場合、その平方根を返す関数を作成しなさい。それ以外の場合は0を返します。

- 期待される機能

```
func Sqrt(nb int) int {  
      
}  
  
パッケージメイン  
  
輸入  
-----  
$ go mod init ex05  
$ go run .2  
0  
func n$  
-----  
fmt.Println(piscine.Sqrt(3))である。  
}
```

第VIII章 Exercise06

: isprimeについて

	エクササイ ズ06
提出先ディレクトリ:	イズブ
ex06/ 提出するファイル	ライム
* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数:なし	

パラメータとして渡されたintが素数であればtrueを返す関数を作成しなさい。
それ以外の場合はfalseを返す。

- (正の数のみが素数になり得ると考える)。
- (1が素数でないことも考慮する)。
- 期待される機能

```
func IsPrime(nb int) bool {  
}
```

パッケージメイン

輸入


```
    "fmt"  
    "piscine"  
)
```

```
func main() {  
    fmt.Println(piscine.IsPrime(5))  
    fmt.Println(piscine.IsPrime(4))です。  
}
```

```
$ go mod init ex06
$ go run
true
擬似
$
```

第IX章

Exercise07: 次の素数を見つける

	エクササイズ 07
提出先ディレクトリ:	次点
ex07/ 提出するファイル	
:* 許可されたパッケージ:	
fmt	
使用可能な組み込み関数:	なし

パラメータとして渡されたintと等しいかそれより大きい最初の素数を返す関数を書きなさい。

- (正の数のみが素数になり得ると考える)。
- 期待される機能

```
func FindNextPrime(nb int) int {  
}
```

パッケージメイン

輸入

```
"fmt"  
$ go mod init ex07  
$ go run .
```

```
func main() {  
    fmt.Println(piscine.FindNextPrime(5))  
    fmt.Println(piscine.FindNextPrime(4))  
    です。  
}
```

Go


PiscineGo

02

02 01 01

第X章

Exercise08 : エイトクイーン

	エクササイズ 08
提出先ディレクトリ:	八十帖
ex08/ 提出するファイル	
:* 許可されたパッケージなし	
使用可能な組み込み関数:なし	

8人の女王のパズルの解答を表示する関数を作成しなさい.

- この問題を解決するためには、再帰性を利用する必要があります。
- 期待される機能

```
func EightQueens() {  
}
```

```
$ go mod init ex08  
$ go run  
.15863724  
16837425  
17468253  
...  
$
```