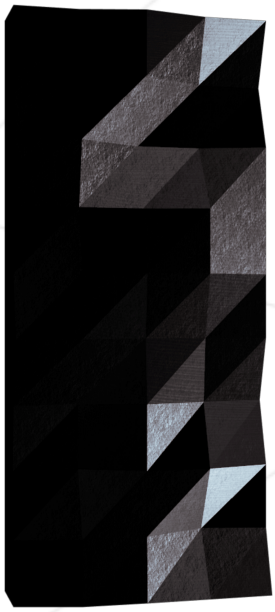

Go Piscine

Go 05

Summary: THIS document is the subject for the Go 05 module of the Go Piscine @ 42Tokyo.

Contents

Chapter I

Instructions

- Only this page will serve as reference; do not trust rumors.
- Watch out! This document could potentially change up to an hour before submission.
- These exercises are carefully laid out by order of difficulty - from easiest to hardest. We `will not` take into account a successfully completed harder exercise if an easier one is not perfectly functional.
- Make sure you have the appropriate permissions on your files and directories.
- You have to follow the submission procedures for every exercise.
- Your exercises will be checked and graded by your fellow classmates.
- You cannot leave any additional file in your directory than those specified in the subject.
- Got a question? Ask your peer on the right. Otherwise, try your peer on the left.
- Your reference guide is called `Google / man / the Internet / ...`.
- Examine the examples thoroughly. They could very well call for details that are not explicitly mentioned in the subject...
- If no other explicit information is displayed, you must use the latest versions of Go.
- Your turn-in directory for each exercise should look something like this:

```
ex[XX]
|-- main.go
|-- vendor
    |-- ft
        |-- printrune.go
    |-- piscine
        |-- [excercisename].go
```

Chapter II

Exercise 00 : appendrange

	Exercise 00
appendrange	
Turn-in directory : $ex00/$	
Files to turn in : `*`	
Allowed packages : `fmt`	
Allowed builtin functions : `None`	

Write a function that takes an int min and an int max as parameters. The function should return a slice of ints with all the values between the min and max.

- Min is included, and max is excluded.
- If min is greater than or equal to max, a nil slice is returned.
- `make` is not allowed for this exercise.
- Expected function

```
func AppendRange(min, max int) []int {

}
```

- Usage

```
package main

import (
        "fmt"
        "piscine"
)

func main() {
        fmt.Println(piscine.AppendRange(5, 10))
        fmt.Println(piscine.AppendRange(10, 5))
}
```

- Output of usage

```
$ go mod init ex00
$ go run .
[5 6 7 8 9]
[]
$
```

Chapter III

Exercise 01 : makerange

	Exercise 01
makerange	
Turn-in directory : $ex01/$	
Files to turn in : `*`	
Allowed packages : `fmt`	
Allowed builtin functions : `None`	

Write a function that takes an int min and an int max as parameters. The function must return a slice of ints with all the values between min and max.

- Min is included, and max is excluded.
- If min is greater than or equal to max, a nil slice is returned.
- `append` is not allowed for this exercise.
- Expected function

```
func MakeRange(min, max int) []int {

}
```

- Usage

```
package main

import (
        "fmt"
        "piscine"
)

func main() {
        fmt.Println(piscine.MakeRange(5, 10))
        fmt.Println(piscine.MakeRange(10, 5))
}
```

- Output of usage

```
$ go mod init ex01
$ go run .
[5 6 7 8 9]
[]
$
```

Chapter IV

Exercise 02 : concatparams

	Exercise 02
concatparams	
Turn-in directory : $ex02/$	
Files to turn in : `*`	
Allowed packages : `fmt`	
Allowed builtin functions : `None`	

Write a function that takes the arguments received in parameters and returns them as a string. The string is the result of all the arguments concatenated with a newline (n) between.

- Expected function

```
func ConcatParams(args []string) string {

}
```

- Usage

```
package main

import (
	"fmt"
	"piscine"
)

func main() {
	test := []string{"Hello", "how", "are", "you?"}
	fmt.Println(piscine.ConcatParams(test))
}
```

- Output of usage

```
$ go mod init ex02
$ go run .
Hello
how
are
you?
$
```

Chapter V

Exercise 03 : splitwhitespaces

	Exercise 03
splitwhitespaces	
Turn-in directory : *ex*03/	
Files to turn in : *	
Allowed packages : fmt	
Allowed builtin functions : None	

Write a function that separates the words of a string and puts them in a string slice.

- The separators are spaces, tabs and newlines.
- Expected function

```
func SplitWhiteSpaces(s string) []string {

}
```

- Usage

```
package main

import (
        "fmt"
        "piscine"
)

func main() {
        fmt.Printf("%#v\n", piscine.SplitWhiteSpaces("Hello how are you?"))
}
```

- Output of usage

```
$ go mod init ex03
$ go run .
[]string{"Hello", "how", "are", "you?"}
$
```

Chapter VI

Exercice 04 : printwordstables

	Exercice 04
printwordstables	
Turn-in directory : $ex04/$	
Files to turn in : `*`	
Allowed packages : `None`	
Allowed builtin functions : `None`	

Write a function that receives a string slice and prints each element of the slice in a seperate line.

- Expected function

```
func PrintWordsTables(a []string) {

}
```

- Usage

```
package main

import "piscine"

func main() {
	a := piscine.SplitWhiteSpaces("Hello how are you?")
	piscine.PrintWordsTables(a)
}
```

- Output of usage

```
$ go mod init ex04
$ go run .
Hello
how
are
you?
$
```

Chapter VII

Exercise 05 : convertbase

	Exercise 05
convertbase	
Turn-in directory : $ex05/$	
Files to turn in : `*`	
Allowed packages : `fmt`	
Allowed builtin functions : `None`	

Write a function that does the following.

- The function receives three arguements:
  - nbr: A string representing a numberic value in a base.
  - baseFrom: A string representing the base nbr it's using.
  - baseTo: A string representing the base nbr should be represented in the returned value.
- Only valid bases will be tested.
- Negative numbers will not be tested.
- Expected function

```
func ConvertBase(nbr, baseFrom, baseTo string) string {

}
```

- Usage

```
package main

import (
        "fmt"
        "piscine"
)

func main() {
        result := piscine.ConvertBase("101011", "01", "0123456789")
        fmt.Println(result)
}
```

- Output of usage

```
$ go mod init ex05
$ go run .
43
$
```

Chapter VIII

Exercise 06 : split

	Exercise 06
split	
Turn-in directory : *ex*06/	
Files to turn in : *	
Allowed packages : fmt	
Allowed builtin functions : None	

Write a function that capitalizes each letter in a string.

- Expected function

```
func Split(s, sep string) []string {

}
```

- Usage

```
package main

import (
        "fmt"
        "piscine"
)

func main() {
        s := "HelloHAhowHAareHAyou?"
        fmt.Printf("%#v\n", piscine.Split(s, "HA"))
}
```

- Output of usage

```
$ go mod init ex06
$ go run .
[]string{"Hello", "how", "are", "you?"}
$
```