



Go Piscine

Go 00

*Summary: THIS document is the subject for the Go 00 module of the Go Piscine @ 42Tokyo.*

# Contents

<b>I</b>	<b>Instructions</b>	<b>2</b>
<b>II</b>	<b>Exercise 00 : printalphabet</b>	<b>3</b>
<b>III</b>	<b>Exercise 01 : printreversealphabet</b>	<b>4</b>
<b>IV</b>	<b>Exercise 02 : printdigits</b>	<b>5</b>
<b>V</b>	<b>Exercise 03 : isnegative</b>	<b>6</b>
<b>VI</b>	<b>Exercise 04 : printcomb</b>	<b>8</b>
<b>VII</b>	<b>Exercise 05 : printcomb2</b>	<b>10</b>
<b>VIII</b>	<b>Exercise 06 : printcombn</b>	<b>11</b>


# 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.

# Chapter II

## Exercise 00 : printalphabet

	Exercise 00
printalphabet	
Turn-in directory : <i>ex00/</i>	
Files to turn in : *	
Allowed packages : <a href="https://github.com/42tokyo/ft">github.com/42tokyo/ft</a>	
Allowed builtin functions : None	


Write a program that prints the Latin alphabet in lowercase on a single line.

- A line is a sequence of characters preceding the end of line character ('`\n`').
- Usage

```
$ go run .  
abcdefghijklmnopqrstuvwxyz  
$
```

# Chapter III

## Exercise 01 : printreversealphabet

	Exercise 01
printreversealphabet	
Turn-in directory : <i>ex01/</i>	
Files to turn in : *	
Allowed packages : <a href="https://github.com/42tokyo/ft">github.com/42tokyo/ft</a>	
Allowed builtin functions : None	


Write a program that prints the Latin alphabet in lowercase in reverse order (from 'z' to 'a') on a single line.

- A line is a sequence of characters preceding the end of line character ('\n').
- Usage

```
$ go run .  
zyxwvutsrqponmlkjihgfedcba  
$
```

# Chapter IV

## Exercise 02 : printdigits

	Exercise 02
printdigits	
Turn-in directory : <i>ex02/</i>	
Files to turn in : *	
Allowed packages : <a href="https://github.com/42tokyo/ft">github.com/42tokyo/ft</a>	
Allowed builtin functions : None	


Write a program that prints the decimal digits in ascending order (from 0 to 9) on a single line.

- A line is a sequence of characters preceding the end of line character (' \n').
- Usage

```
$ go run .  
0123456789  
$
```

# Chapter V

## Exercise 03 : isnegative

	Exercise 03
isnegative	
Turn-in directory : <i>ex03/</i>	
Files to turn in : *	
Allowed packages : <a href="https://github.com/42tokyo/ft">github.com/42tokyo/ft</a>	
Allowed builtin functions : None	

Write a function that prints 'T' (true) on a single line if the int passed as parameter is negative, otherwise it prints 'F' (false).

- Expected function

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

- Usage

```
package main  
  
import "piscine"  
  
func main() {  
    piscine.IsNegative(1)  
    piscine.IsNegative(0)  
    piscine.IsNegative(-1)  
}
```


- Output of usage

```
$ go run .  
F  
F  
T  
$
```



# Chapter VI

## Exercise 04 : printcomb

	Exercise 04
printcomb	
Turn-in directory : <i>ex04/</i>	
Files to turn in : *	
Allowed packages : <a href="https://github.com/42tokyo/ft">github.com/42tokyo/ft</a>	
Allowed builtin functions : None	

Write a function that prints, in ascending order and on a single line: all unique combinations of three different digits so that, the first digit is lower than the second, and the second is lower than the third.

- These combinations are separated by a comma and a space.
- Expected function

```
func PrintComb() {  
}
```

- Usage

```
package main  
  
import "piscine"  
  
func main() {  
    piscine.PrintComb()  
}
```


- Incomplete output of usage

```
$ go run . | cat -e
012, 013, 014, 015, 016, 017, 018, 019, 023, ..., 689, 789$
$
```

- 000 or 999 are not valid combinations because the digits are not different.
- 987 should not be shown because the first digit is not less than the second.

# Chapter VII

## Exercise 05 : printcomb2

	Exercise 05
	printcomb2
	Turn-in directory : <i>ex05/</i>
	Files to turn in : *
	Allowed packages : <a href="https://github.com/42tokyo/ft">github.com/42tokyo/ft</a>
	Allowed builtin functions : None

Write a function that prints in ascending order and on a single line: all possible combinations of two different two-digit numbers.

- These combinations are separated by a comma and a space.
- Expected function

```
func PrintComb2() {  
}
```

- Usage


```
package main  
  
import "piscine"  
  
func main() {  
    piscine.PrintComb2()  
}
```

- Incomplete output of usage

```
$ go run . | cat -e  
00 01, 00 02, 00 03, ..., 00 98, 00 99, 01 02, 01 03, ..., 97 98, 97 99, 98 99$  
$
```

# Chapter VIII

## Exercise 06 : printcombn

	Exercise 06
printcombn	
Turn-in directory : <i>ex06/</i>	
Files to turn in : *	
Allowed packages : <a href="https://github.com/42tokyo/ft">github.com/42tokyo/ft</a>	
Allowed builtin functions : None	

Write a function that prints all possible combinations of n different digits in ascending order.

- n will be defined as :  $0 < n < 10$
- Below are the references for the printing format expected.
- (for n = 1) '0, 1, 2, 3, ..., 8, 9'
- (for n = 3) '012, 013, 014, 015, 016, 017, 018, 019, 023,...689, 789'
- Expected function

```
func PrintCombN(n int) {  
}
```

- Usage

```
package main  
  
import "piscine"  
  
func main() {  
    piscine.PrintCombN(1)  
    piscine.PrintCombN(3)  
    piscine.PrintCombN(9)  
}
```

- Incomplete output of usage

```
$ go run .  
0, 1, 2, 3, 4, 5, 6, 7, 8, 9  
012, 013, 014, 015, 016, 017, 018, ... 679, 689, 789  
012345678, 012345679, ..., 123456789  
$
```