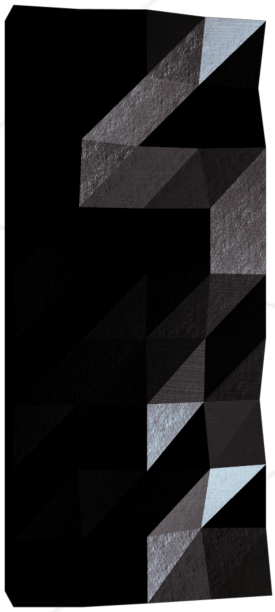

Introduction to Programming 42

Day 02

Kai kai@42.us.org
Gaetan gaetan@42.us.org

Summary: This document is the subject of the day 02 of the introduction to programming piscine.

Contents

Chapter I

Guidelines

- Corrections will take place in the last hour of the day. Each person will correct another person according to the peer-corrections model.
- Questions? Ask the neighbor on your right. Next, ask the neighbor on your left.
- Read the examples carefully. The exercises might require things that are not specified in the subject...
- Your reference manual is called Google / "Read the Manual!" / the Internet / ...

Chapter II

Preamble

`Intro text will go here.`

Text is done.

Chapter III

Exercise 00 : puts each

- Create a script `puts_each.rb` which declares a number array [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] and uses each method to iterate over the array and display each value.

```
?> ./puts_each.rb | cat -e
1$
2$
3$
4$
5$
6$
7$
8$
9$
10$
?>
```

Google array, each

Chapter IV

Exercise 01 : display rev params

- Create a script `display_rev_params.rb`

```
?> ./display_rev_params.rb | cat -e
none$
?> ./display_rev_params.rb "yolo" | cat -e
none$
?> ./display_rev_params.rb "Garkbit" "God" "Genghis Khan"
Genghis Khan$
God$
Garkbit$
?>
```

`Google ARGV, array, reverse`

Chapter V

Exercise 02 : Did you get that

- Create a script `i_got_that.rb`. This script must contain a while loop that accepts a user input, write a return phrase, and stops only when the user has entered "STOP!". Each round of loops must accept input from the user.

```
?> ./i_got_that.rb | cat -e
What you gotta say?: Hello$
I got that! Anything else?: I like ponies$
I got that! Anything else?: stop...$
I got that! Anything else?: STOP!
?>
```

Google while, break

Chapter VI

Exercise 03 : Strings are Arrays

- Create a script `strings_arr.rb` that takes a character string as a parameter. When executed, the script displays "z" for each character "z" in the string passed as a parameter, followed by a newline. If the number of parameters is different that 1, or there is no "z" character in the string, display "none" followed by a newline.

```
?> ./strings_arr.rb "The target character is not in this string" | cat -e
none$
?> ./strings_arr.rb "z" | cat -e
z$
?> ./strings_arr.rb "Zealously zany zapping zebras zigzag zested Zoology zone" | cat -e
zzzzzz$
?>
```

Strings are also composed of boxes. You can do it!

Chapter VII

Exercise 04 : Array Play

- Create a script `file.rb` which takes an array of numbers (that you define) and builds a new array that is the result of adding the value 2 to each value of the original array. You must have two arrays at the end of the program, the original one and the new one you created. Display the two arrays on the screen using the p method rather than puts. For example, if your original array is [2, 8, 9, 48, 8, 22, -12, 2] you will get the following output:

```
?> ./play_with_arrays.rb | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
[4, 10, 11, 50, 10, 24, -10, 4]$
?>
```

Google p method in ruby, array each

Chapter VIII

Exercise 05 : Array++

- Create a script `play_with_arrays.rb`

```
?> ./play_with_arrays.rb | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
[10, 11, 50, 10, 24]$
?>
```

Google p method in ruby, array each

Chapter IX

Exercise 06 : Array+=2

- Create a script `play_with_arrays.rb`

```
?> ./play_with_arrays.rb | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
[10, 11, 50, 24]$
?>
```

`Google array, uniq`

Chapter X

Exercise 07 : Count It

- Create a script `count_it.rb`, which, when executed, has "parameters:" and then the number of parameters, followed by a newline, then each parameter and its size followed by a newline. If there is no parameter, output `none` followed by a newline.

```
?> ./count_it.rb "Life" "Universe" "Everything" | cat -e
parameters: 3$
Life: 4$
Universe: 8$
Everything: 10$
?>
```

Google size

Strictly adhere to the format indicated in the example.

Chapter XI

Exercise 08 : Pig Latin

- Create a script `atinlay.rb` that translates its arguments into pig latin.
- If the word starts with one or more consonents, remove the group of consonants and add them to the end of the string with a hypen. Then, add the letters "ay".
- If the first letter is a vowel, just add a "way" t the end.
- If there is no parameter, display `none` followed by a newline.
- You should probably downcase them, too

```
?> ./append_it.rb "paranoid" "android" "Marvin" "so" "stoic" | cat -e
aranoidpay$
androidway$
arvinmay$
osay$
oicstay$
?>
```

`oink, oink`