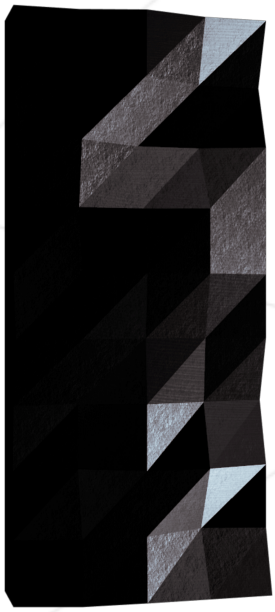

Input-Output

Learn to talk

Kai kai@42.us.org

Summary: Keep a friend at your side, and study in the path of inspired ones who have given their knowledge out for free.

Contents

Chapter I

Before you Start!

Create your project folder:

1. From your project page on intra, copy the git repository link. Now, in the terminal type "git clone " and paste the link. After the link and before pressing enter, write a name for the new folder. Cloning your git repository always creates a new folder.

2. cd into the folder you just created and from now on, save your work there. Use the command "mkdir <name>" to create new folders. Put each puzzle from this project in a folder with the same name.

Chapter II

Format your Code

Each 42 challenge you turn in must adhere to the following format:

```ruby
#!/usr/bin/env ruby

# This is what my program does
# By <userid>

def function_a
 #code
end

def function_b
 #code
end

def main(ARGV)
 #main method
 function_a
 function_b
end

main(ARGV)
```

- Always begin with the "#!/usr/bin/env ruby" statement. This tells your terminal to run the program using Ruby. In python, the first line is "#!/usr/bin/env python".
- Always add a comment stating what this program is for, some hints to help others use or understand it, and your name or intra ID.
- Do not write any code outside of functions except for one line, at the end of your program, which calls the main() function.
- The (ARGV) parameter is not always needed. In Python it is sys.argv.

Reference your chosen intro to coding class to learn about functions/methods (The keyword "def" means "define function...").

Chapter III

Exercise 0: What is your name

	Exercise
Turn-in directory : `ex00`	
Files to turn in : `ex00.rb or ex01.py`	
Notes : `Ruby` IO, ARGF `Python` Built-in Functions, System	

- Create a script `ex00.rb` which asks your name and greets you with it.

```
?> ruby ex00.rb
Hello hacker, what is your name?
?> O'Brian
Welcome, O'Brian.
```

Chapter IV

Exercise 1: Poetry

	Exercise
Turn-in directory : `ex01`	
Files to turn in : `ex01.rb or ex01.py`	
Notes : `Ruby IO, ARGF Python Built-in Functions, System`	

Create a short Mad-Libs puzzle. It will take a command-line argument which sets the title of the madlib. Then, your program prompts your corrector for the following:

- an adjective (for example, "fruity")
- a business (for example, "orchard")
- an animal (for example, "bat")
- a noise (for example, "click")

Then, print a version of "Old MacDonald Had a Farm" using the user input instead of some of the traditional words.

Example of the program running:

```
?> ruby 4200_io.rb "Ode to Joy"
Please input an adjective: fruity
Please input a business: orchard
Please input an animal: bat
Please input a noise: click

ODE TO JOY
fruity Macdonald had a orchard, E-I-E-I-O
and on that orchard he had a bat, E-I-E-I-O
with a click click here
and a click click there,
here a click, there a click,
everywhere a click click,
fruity Macdonald had a orchard, E-I-E-I-O!
```

`Bonuses!` Finish these bells and whistles to get extra credit.

- match the example exactly.

- change 'a to 'an' depending if the word starts with a vowel.
- put the title in call caps.
- capitalize the start of each line including the adjective.
- put other input words in lowercase if they were capitalized.
- print an error message if the input is empty, longer than one word, or otherwise nonsensical.

Chapter V

Grade your work!

Turn in your work by typing three commands in order:

- git add *
- git commit -m "<your comments here>"
- git push
- If you have an error during the git push, you may need to refresh your authentication ticket. Do this by typing "kinit <username>" and then typing your intra password.

Then, go to your project page and click "Set the project as finished". Next click "Subscribe to defense" and schedule two peer corrections. If you run out of correction points (check your number on your profile page!), it means you need to open correction slots and correct other people in return. :)