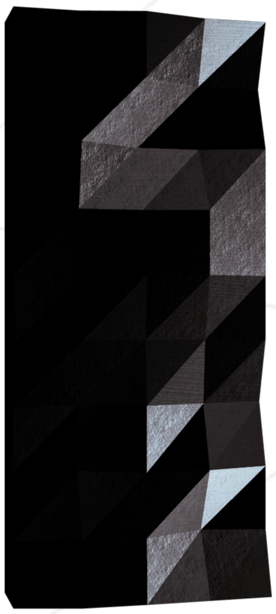

String

Learn to Write

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:

```
#!/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: Strings

	Exercise
Turn-in directory : `ex00`	
Files to turn in : `ex00.rb or ex00.py`	
Notes : `Ruby String, Regular Expressions Python String, Regular Expressions`	

Write a progam which performs three tasks in a row on the phrase given as a command line argument.

First, it prints out the phrase with eVeRy OtHeR lEtTeR cApItAlIzEd.
Then, print out the same string with every capitalized vowel replaced by an asterisk.
Next, run a check_parentheses function which determines whether every open paren "(" in the phrase is balanced with a close paren ")". Print out "Balanced? True" or "Balanced? False" accordingly.

```
?> ruby ex00.rb "(whats going on()?)"
(WhAtS gOiNg On()?)
(Wh*tS g*iNg *n()?)
Balanced? True
```

```
?> ruby ex00.rb "()()()()(((a)b)b)b"
()()()()(((A)b)B)b
()()()()(((*)b)B)b
Balanced? True
```

```
?> ruby ex00.rb "((this doesn't match)"
((ThIs DoEsN't MaTcH)
((Th*s Do*sN't MaTcH)
Balanced? False
```

`Use gsub or re.sub. Study a little about regex. Look up what string interpolation means & use it all the time.`

Chapter IV

Exercise 1: Do You Even 101010

	Exercise
Turn-in directory : `ex01`	
Files to turn in : `ex01.rb or ex01.py`	
Notes : `n/a`	

- Create a script `ex06.rb` which takes in a number in base 10 and prints out its equivalent in base 2 (binary).

```
?> ruby ex01.rb 94555
10111000101011011
?>
```

`There are two types of people in the world: those who understand binary, and those who donut.`

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. :)