Portfolio element – Ruby

Unit	Programming languages: principles and design (6G6Z1110) Programming languages – SE frameworks (6G6Z1115)
	Frogramming languages – SE frameworks (000Z1113)
Lecturer	Rob Frampton
Week	5
Portfolio element	Ruby (15% of coursework)

Assignment

You must implement a new class in Ruby called LetterHistogram. LetterHistogram has one attribute with a getter and setter, text, which is a *String*. LetterHistogram also has a private method called CalculateFrequencies that computes the frequency of each letter appearing in text and returns the output as a Hash. The method should count lower and uppercase as the same letter. Punctuation as well as any other special character must be ignored. *LetterHistogram* should also have a method called display that displays for each letter A - Z (including letters not in text) a row of asterisks representing how many times the character appeared. The user should be able to initialise "text" when an object type "LetterHistogram" is created. If initial value is not supplied by the user, then it should be assigned "Hello, World!" as default value. See Figure 1 for an example of using "LetterHistogram".

```
irb(main):001:0> load 'LetterHistogram.rb'
=> true
irb(main):002:0> sentence = "Far out in the uncharted backwaters of
the unfashionable end of the Western Spiral arm of the Galaxy lies a
small unregarded yellow sun.">
=> "Far out in the uncharted backwaters of the unfashionable end of
the Western Spiral arm of the Galaxy lies a small unregarded yellow
sun."
irb(main):003:0> h = LetterHistogram.new sentence
    #<LetterHistogram:0x000000003422428
                                          @text="Far out
uncharted backwaters of the unfashionable end of the Western Spiral
arm of the Galaxy lies a small unregarded yellow sun.">
irb(main):004:0> h.display
B:**
C:**
D:***
E:*********
F:****
G:**
H:*****
I:***
J:
L:*****
M:**
N:*****
0:*****
P:*
Q:
```

```
W:***
X:*
Y:**
Z:
```

Figure 1. Example of using "LetterHistogram"

Table 1 summarises the tests that your implementation should pass in order to be considered it is complete and fully functional. Figure 2 shows some examples of those tests.

Table 1: Tests on "LetterHistogram" class

Test	Example	Comments
Create new LetterHistogram object	See Figure 2, line 002.	Initialise "text" with default
		string "Hello, World!"
Create new LetterHistogram object	See Figure 1, line 003.	Initialise "text" with the passed
passing in a String		value.
Get "text" value	See Figure 2, line 003	
Set "text" value	See Figure 2, line 004	
Call "calculateFrequencies" method	See Figure 2, line 005	Not allowed,
		"calculateFrequencies" method
		is private.
Call "display" method	See Figure 1, line 004; and	Displays histogram as indicated
	Figure 2, line 006.	on the assignment.

```
irb(main):001:0> load 'LetterHistogram.rb'
=> true
irb(main):002:0> h = LetterHistogram.new
=> #<LetterHistogram:0x0000000034fb020 @text="Hello World!">
irb(main):003:0> h.text
=> "Hello World!"
irb(main):004:0> h.text = "What's it going to be then, eh?"
=> "What's it going to be then, eh?"
irb(main):005:0> h.calculateFrequencies
NoMethodError: private method `calculateFrequencies'
                                                           called for
#<LetterHistogram:0x0000000034fb020>
        from (irb):5
        from C:/Program Files/Ruby24-x64/bin/irb.cmd:19:in `<main>'
irb(main):006:0> h.display
A:*
B:*
C:
D:
E:***
F:
G:**
H:***
I:**
J:
K:
L:
Μ:
N:**
0:**
P:
Q:
R:
```

Portfolio element – Ruby

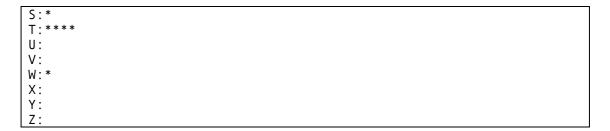


Figure 2. Examples of testing "LetterHistogram"

Submission

You must submit a file named "LetterHistogram.rb" through Moodle, which contains only the LetterHistogram class definition. Submission link is available in Week 5 section.