Assignment #2
278, Fall 2020

Question 1.

Define a string using either literal or “here document”, for example, the following variable “str” is defined with a “here document”

str = <<EOS

Facebook and its founder must release documents and electronic correspondence to a defense lawyer whose client has fled from criminal charges that he falsely claimed a majority ownership in the social media giant, a federal judge said Friday.

EOS

(you can also use %{ } to define string)

then write a method to calculate number of occurrence of any given word. if you pass 2 parameters, str and a string, then it will return number of occurrences of the string in str, if you just pass str, then it will return number of occurrences for every word in a hash

for example:

if using above string:

count_word(str, “and”). #. will return: 2

if the string is:

str = %{three, three, three}

count_word(str). #will return: {“three”=>3}

Question 2.

Define an array of student record, for example,

```
students = [
    {:firstname => "John", :lastname => "LastnameJohn",  :phonenumber => 123456789},
    {:firstname => "Ken", :lastname => "Lastnameken",  :phonenumber => 456734244},
    {:firstname => "Marisa", :lastname => "lastnamemarisa",  :phonenumber => 443234567},
    {:firstname => "Ken", :lastname => "Kenlastname",  :phonenumber => 456734244}
]
```

write a method to be able to query student,

for example find all the record with firstname is ken:

search_students(students, firstname: “ken”), it will print:

First Name Last Name   Phone#

Ken   Lastnameken 456734244
Ken   Kenlastname   456734244

you can also define a class to hold students record data and search method, then pass the object to search_students()

Question 3. Write a class for compressing a string (remove duplicate word and be able to recover the original string). when you create an object of this class, you provide a string, then the object save the compressed result as the attribute of the object. The compressed result will have two arrays: an array for strings and an array for index (for recovering purpose).

For example: assuming the name of your class is Compress

to create an object, you can call like this:

obj = Compress.new(“i love you but do you love me”)

then there will be two instance variables created inside the object to hold two values:

[“i”, “love”, “you”, “but”, “do”, “me”] # no duplicate word (compressed)

[0, 1, 2, 3, 4, 2, 1, 5] # index to the original array to represent original string

Define a method to return the original string.

Question 4. one of the method in Hash class is Hash#merge, it will merge two hashes.
it's format is like this:

merge(second_hash) → new hash
merge(second_hash){ | key, val1, val2 | newvalue} → new hash

if no block is given, it merge two hashes, if there is duplicate key, the value of the "other_hash" will be used. if a block is given, the value for the duplicate key is determined by calling the block with the duplicate key, the value in first hash (val1), and the value in second hash (val2)

for example:

```
h1 = { "a" => 100, "b" => 200 }
h2 = { "b" => 254, "c" => 300 }
h1.merge(h2)            #=> {"a"=>100, "b"=>254, "c"=>300}
h1.merge(h2){|key, val1, val2| val2 - val1}
                        #=> {"a"=>100, "b"=>54,  "c"=>300}
h1                      #=> {"a"=>100, "b"=>200}
```

merge!() is the "dangerous" version of merge(), which will change the h1 to be the merged hash.

re-open the class Hash, re-implement these two methods.

Question 5

A template is a HTML file with Ruby code inside. The Ruby code is marked by <% %>.
or if a line start with %, then the whole line is ruby code

For example, the following is an template file:

```
<%= simple_form_for @project do |f| %>
  <%= f.input :name %>
  <%= f.input :description %>
  <h3>Tasks</h3>
  <div id='tasks'>
    <%= f.simple_fields_for :tasks do |task| %>
      <%= render 'task_fields', :f => task %>
    <% end %>
    <div class='links'>
      <%= link_to_add_association 'add task', f, :tasks %>
    </div>
  </div>
  <%= f.submit 'Save' %>
<% end %>
```

Write a class, the object of this class has a template attribute and an instance method to "filter" its template, so that all ruby code are removed and filtered string is returned.

You can define your string using either here document or normal string quotation %{ }

then create the object, then filter the template and return the filtered string.

[assuming there is no nested case, for example, <% a <% b %> c %>]

Requirement:

- Define your classes in each .rb file

- Write a main .rb file to require these .rb files, and write code to create object and call method and output results.

- Write comment

- Your program should do some data check, for example, in question 1, if str is empty. in question 2, if no match, print out "no match is found".

- Use #! to make your main .rb program be able to run directly.