Ruby - Strings - Indexing

When you index a string, you extract or alter the portions. This is perhaps the most important operation you want to perform on strings. The string class provides a convenient array-like square bracket `[]` operator, which allows you to extract portions of the string as well as altering the content when used on the left side of an assignment.

Consider the following examples:

```
> str = "Hello World!"
> str[str.size-1]
"!"
> str[-1] # first character from the end of the string
"!"
> str[-2] # second last character
"d"
> str[0] # first character
"H"
```

More often, you'd want to extract specific portions of the string rather than individual characters. To do this, use comma separated operands between the square brackets. The first operand specifies an index (which may be negative), and the second specifies a length (which must be non-negative).

Consider the example below:

```
> str[0,4] # first four characters
"Hell"
> str[6,3] # 3 characters starting from index 6 ( 0-indexing )
"Wor"
> str[-1, 1] # 1 character starting from the END of string
"!"
```

The same examples shown above can be used for assignment / deletion of characters. We can insert characters by giving a non-empty assignment or delete characters in the range by giving an empty assignment.

Consider the example below:

```
> str = "Hello"
> str[str.size, 0] = " World!" # append by assigning at the end of the string
> str
"Hello World!"
> str[5, 0] = "," # insert a comma after the 5th position
> str[5, 6] = ""  # delete 6 characters starting from index 6.
"Hello!"
> str[5,1] = " World" # replace the string starting from index 5 and of length 1 with the given string.
```

But wait, there's more! Ruby also allows you to index strings using a *Range* or a *Regexp* object as well. We will discuss these methods in the future.

In this challenge, your task is to code a `serial_average` method which is described below:

- It takes a fixed width string in format: `SSS-XX.XX-YY.YY`. `SSS` is a three digit serial number, `XX.XX` and `YY.YY` are two digit numbers including up to two decimal digits.
- It returns a string containing the answer in format `SSS-ZZ.ZZ` where `SSS` is the serial number of that

input string, and `ZZ.ZZ` is the average of `XX.XX` and `YY.YY`.

- All numbers are rounded off to two decimal places.

For example:

```
> serial_average('002-10.00-20.00')
"002-15.00"
```

You can use string interpolation to insert Ruby code within a string.

For example:

```
> tmp = 123
> "Hello #{tmp}"
Hello 123
```