CP2–Assignment–Instructions

Aidan O'Hara

2023-08-15

CP2 Assignment Instructions

- Now that you are here in your own GitHub Classroom repository use the `CP2-Assignment-Instructions.pdf` and `cp2_assignment.R` files to complete the assignment.
- Note that slides from lectures, cheat sheets, and other useful information can be found in the Resources folder.
-

Assignment Completion

- Within `CP2-Assignment-Instructions.pdf` you will find instructions about how to complete each exercise.
- Within `cp2_assignment.R` you'll find empty variable names for you to assign your answers to. It is VERY important that you use the provided variable names as Gradescope will not look for your answers under a different name.
- Once you've completed the exercises, or just some of them, you can commit your changes to the classroom repo, just like we've been practicing.
- Afterward, locate **CP-2 Assignment** in Gradescope, and submit the link for your repository to Gradescope. Gradescope will score you accordingly. Multiple submissions are encouraged and coding solutions to fit the specifics, pre-loaded, on Gradescope can be tricky. Pay close attention to the instructions and your answers to make things easiest.

Exercise 1

Create the vector (2,0,2,3) and assign it to `aVector`.

Exercise 2

a. Create the vector (20,19,18,17,16,. . . ,0). Call your vector `tweVector`.

b. Create the vector (2,1.9,1.8,1.7,1.6,. . . ,0). Call your vector `twe.Vector`.

c. Create the vector (2,2,2,2,2,2,2,2), and use the `rep` function. Call your vector `twoVector`.

Exercise 3

Call `pi` in the console. Create a vector with the first 7 elements of pi. Call your vector `piVector`.

Next, using `rev` and/or `sort`, create `pieVector` with the `piVector` elements in decreasing order and `eipVector` with the `piVector` elements in increasing order.

Exercise 4

Construct a vector called `megaVector` that contains the following elements:

- the first and last element of `piVector`
- the elements of `pieVector` equal to `3`
- the third and fourth elements of `eipVector`

Exercise 5

`rivers` is a vector of river lengths, in miles, included with base R. Try `rivers` in console, or better still `?rivers` for more information.

For this exercise logical operators in your vector selections to construct the following from the `rivers` vector:

a. `noMeanRivers` : Including rivers less than 200 miles and rivers more than 600 miles.

b. `river301` : With only rivers that are exactly 301 miles long.

c. `specRivers` : With special rivers between 100 and 200 miles, as well as rivers between 500 and 600 miles, do **not** include rivers with lengths equal to 135.

Exercise 6

Write a function that prints a message citing a relatively-sized river given a positive integer representing miles, use the following rivers as breakpoints:

- Feather River: 73 miles
- Des Moines River: 525 miles
- Rio Grande River: 1896 miles

Your function should return a string in the following format: A river X miles long is shorter than/ longer than/ exactly as long as the "most similar river from above".

ex:

- "A river 325 miles long is shorter than the Des Moines River."
- "A river 9000 miles long is longer than the Rio Grande River."
- "A river 525 miles long is exactly as long as the Des Moines River."

Break ties in favor of the longer river.