Java Test

Expert : Aycan Gülmez

09.01.2020

Sabri Bayrakdar

In this test, I had already working code.But I was supposed to add some new features with following flags:

-c -> create a csv file for report

—top <number> -> sort according to file size and cut from beginning to number'th file.

Firstly, I created a report directory and a file called report.csv in that directory.I created initial two boolean and one count integer.Booleans are for c and top flags and count integer is for top number.I initially check the String args list and set that initial created variable accordingly.

If top flag is given and its number is valid, then I sorted all files according to their sizes and I write it's first given top number element down in the report.If c flag is given too, then I write the calculated report into report.csv file, otherwise I just print the report to console.

If top flag is not given, then the code is running as given to me except for c flag.The code is working as the same the code given to me except writing down the report.csv file.If c flag is true then it writes report.csv file.

I created sortFiles and findUser functions and modified main and run functions.findUser function returns a user with given userId number.sortFiles function sorts the files list according to file sizes.

The algorithm of the sortFiles is the following:

- Start with the first element, search all elements and find maximum file size and its index.
- Compare the maximum size file value with the first element of files list.If maximum value is greater than first element size, then swap that two file order.
- Reset the maximum size value to zero.
- The code doesn't have to start from the first element to find the second maximum file size anymore.Because the code is sure that first element has greatest file size.So, it didn't include it into comparing pool.
- Therefore, the code find and swap first max value, second, third and so on.
- It is efficient that it didn't start from the first element every time.

Also, I created JunitTest.java file to define a test code for Runner.java file.I initially prepare users.csv and files.csv files.I run the main function of the Runner with an example args list.I check whether the returning value is correct or not with assertEquals function.

I also, changed the some functions from private to public, because I need to access these functions from the JunitTest.java file.

If you want to see what I add the code, I add upload the finished code to GitHub on https://github.com/yusufbayrakdar/reengen.You can check the commit and see the difference from the initial code easily.