# Report for Assignment 1

## Project chosen

Name: Anki

URL: https://github.com/drfeelgood13/anki

Number of lines of code and the tool used to count it: 113.385 LOC, Lizard

Programming language: Python

## Coverage measurement

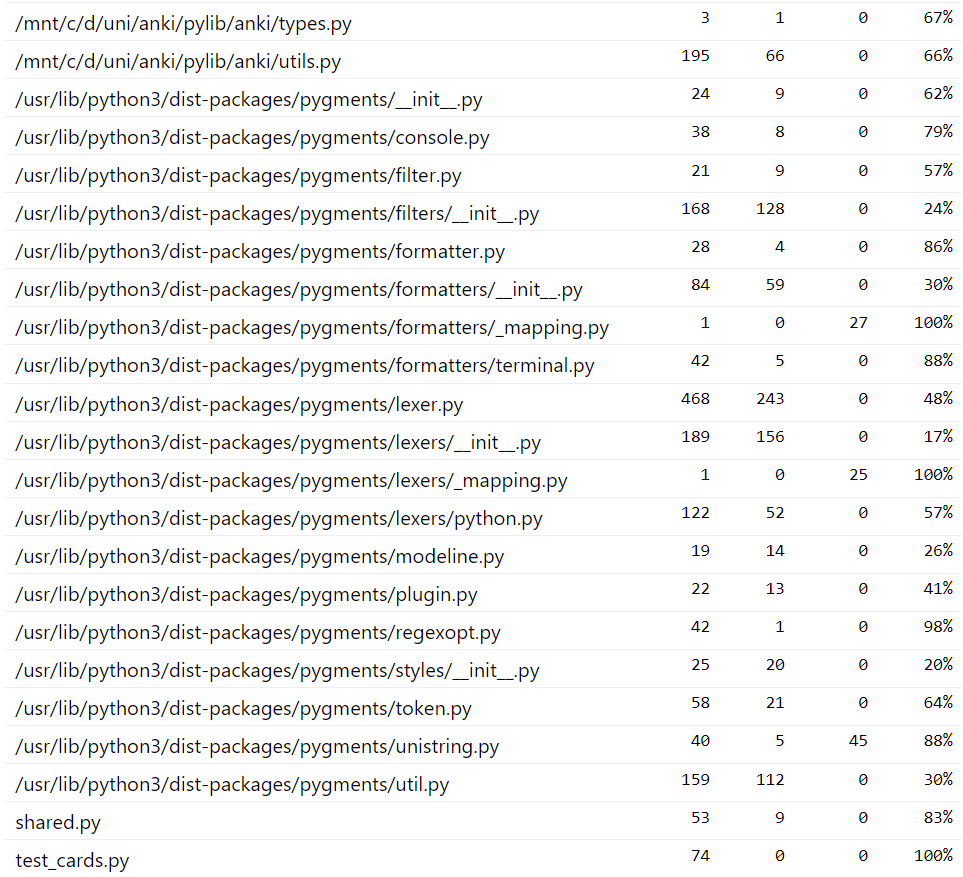
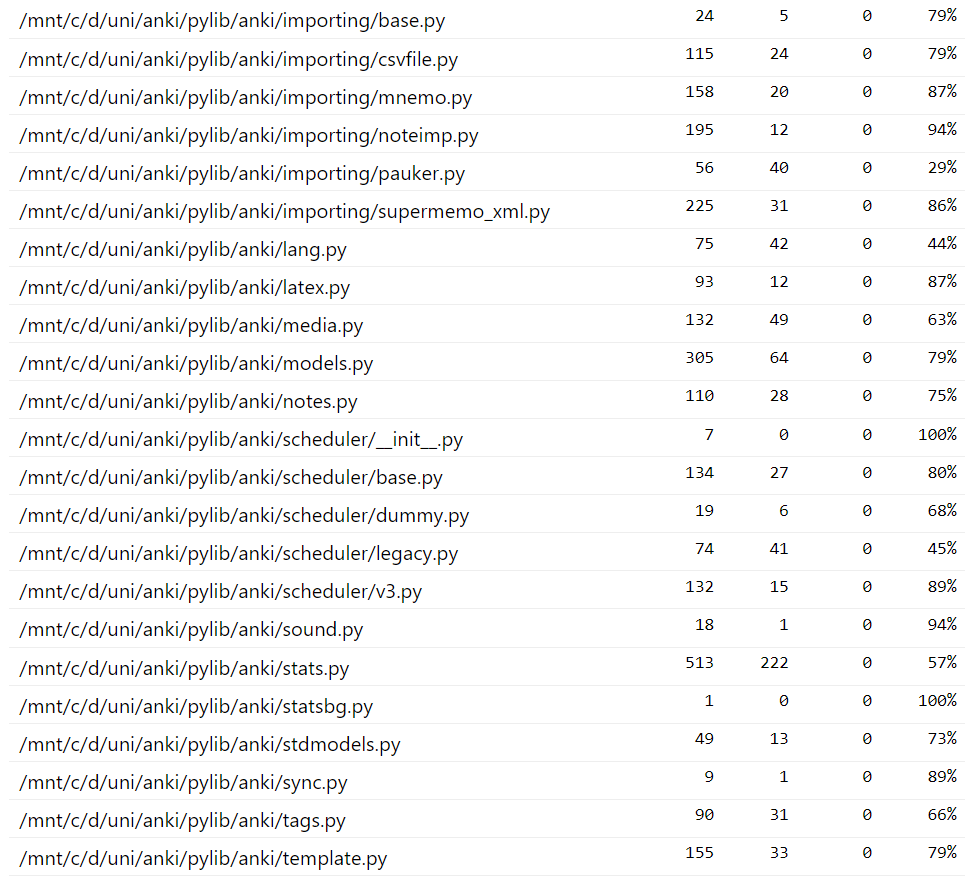
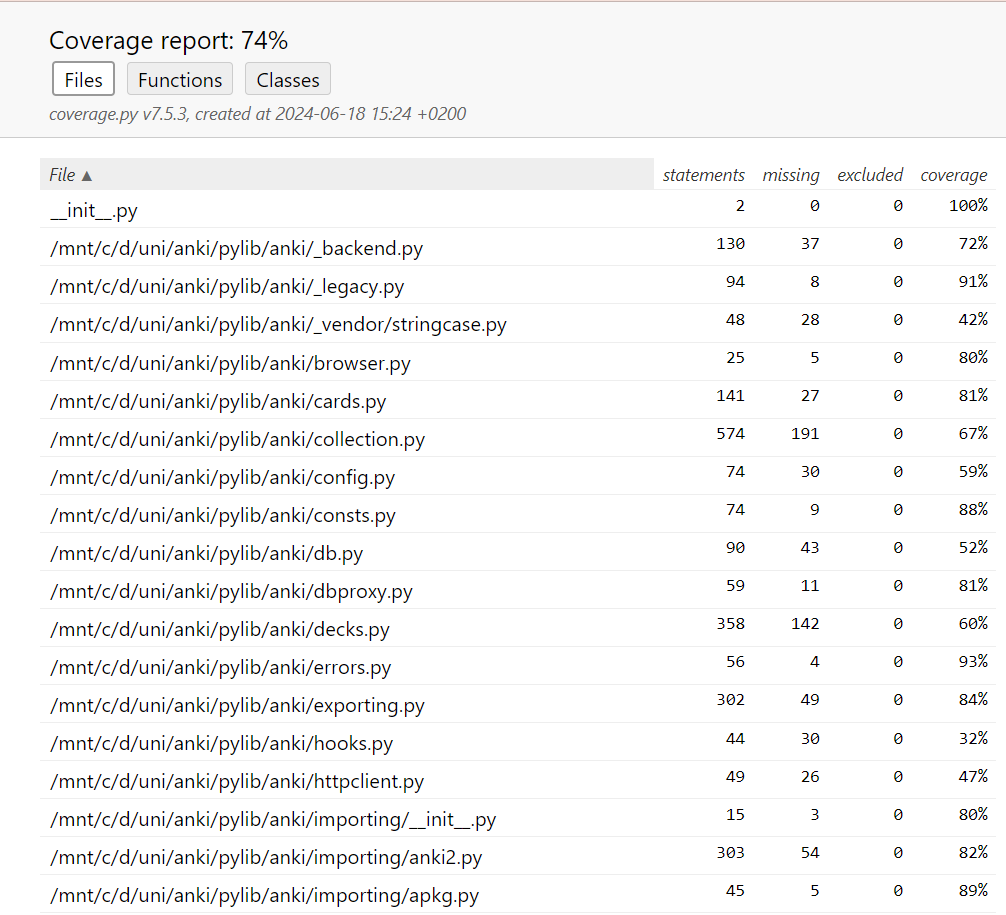
### Existing tool

<Inform the name of the existing tool that was executed and how it was executed>

The name of the existing tool we used is “Coverage.py”. We opened the project and checked if we had Python 3 installed. Some of us didn’t, so we installed the latest version of Python. Then we installed the tool using the following command: $ python3 -m pip install coverage. We then went to the wanted directory to check the coverage for, which is /anki/pylib/tests, and then ran the coverage test using the following command: $ coverage run -m pytest. Afterward, we used the command: $ coverage html to get the coverage report formatted nicely. In the end, we opened the report page from “htmlcov/index.html”.

After we executed the tool, we used the results to identify the poorly tested methods that need to be improved by us.

<Show the coverage results provided by the existing tool with a screenshot>



### Your own coverage tool

Rafael Haenel Musa

<Function 1 name>

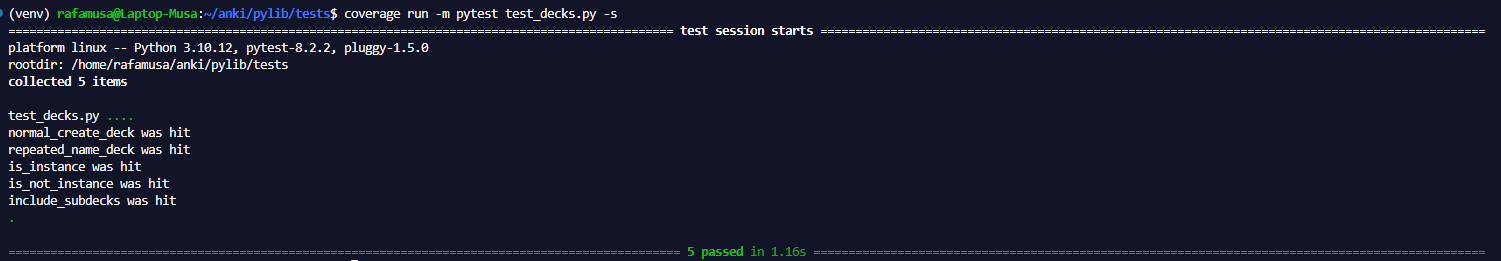
DeckManager.card\_count

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/ankitects/anki/compare/main...drfeelgood13:anki:rafael

<Provide a screenshot of the coverage results output by the instrumentation>





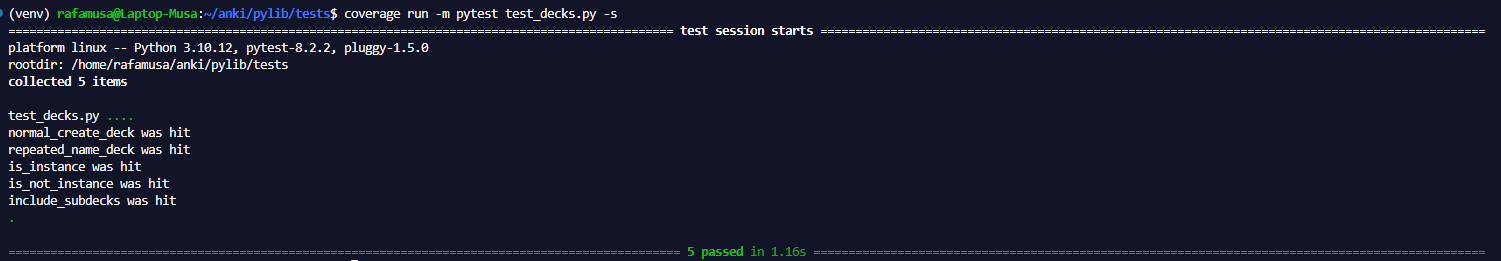
<Function 2 name>

DeckManager.add\_normal\_deck\_with\_name

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/ankitects/anki/compare/main...drfeelgood13:anki:rafael

<Provide a screenshot of the coverage results output by the instrumentation>



Vlad Parau

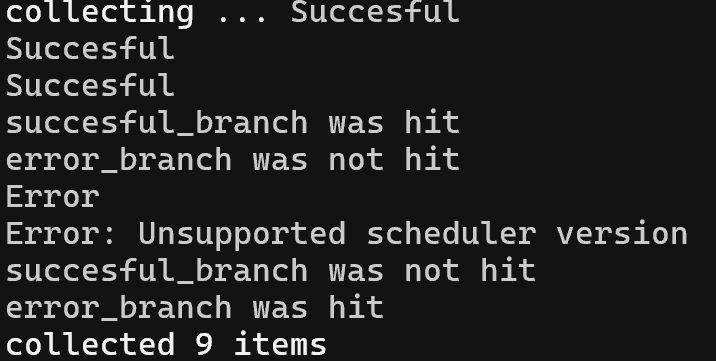
<Function 1 name>

Collection.sched\_ver()

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/ankitects/anki/commit/16be6169707fe02b991cef26d73e3b64c0f5616e

<Provide a screenshot of the coverage results output by the instrumentation>



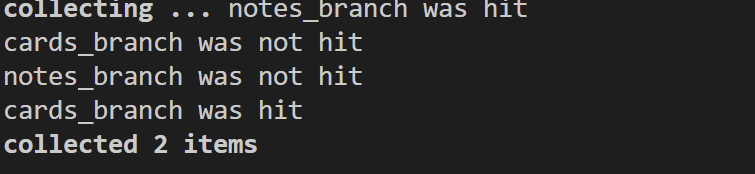
<Function 2 name>

BrowserConfig.sort\_column\_key()

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/ankitects/anki/commit/a7813812749d55efe6dcdfac6106c64816af8824

<Provide a screenshot of the coverage results output by the instrumentation>



Isabela Aterman

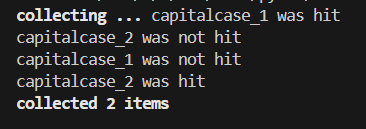
<Function 1 name>

capitalcase()

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/drfeelgood13/anki/commit/dd2dc9836b58607cc5d0ba575f772eb337a2050b

<Provide a screenshot of the coverage results output by the instrumentation>



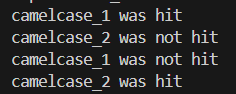
<Function 2 name>

camelcase()

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/drfeelgood13/anki/commit/247fbe0d32928b7280e53d7c9eff536d0a39c1b2

<Provide a screenshot of the coverage results output by the instrumentation>



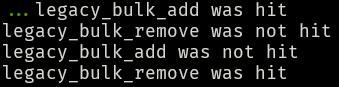
Lukas Jokubauskas

<Function 1 name>  
\_legacy\_bulk\_add

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/ankitects/anki/commit/ac9c6ac103117792340892757378ed0a1717c62c

<Provide a screenshot of the coverage results output by the instrumentation>



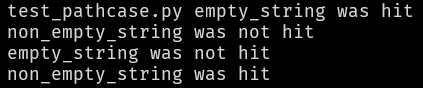
<Function 2 name>

pathcase

<Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements>

https://github.com/ankitects/anki/commit/4432cc229aa721831c310e3e4ef19acb300fbbf2

<Provide a screenshot of the coverage results output by the instrumentation>



## Coverage improvement

### Individual tests

Rafael Hanael Musa

<Test 1 and 2> (they were both made in the same test file for making it easier)

test\_decks.py

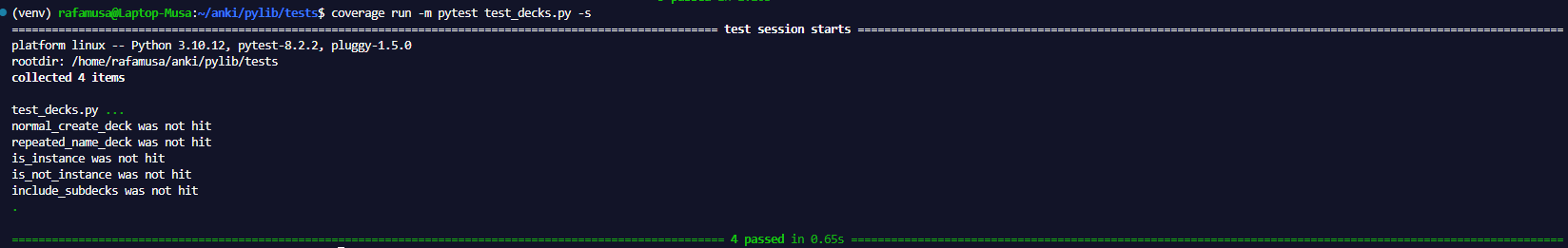
<Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test>

<https://github.com/ankitects/anki/compare/main...drfeelgood13:anki:rafael>

<Provide a screenshot of the old coverage results (the same as you already showed above)>



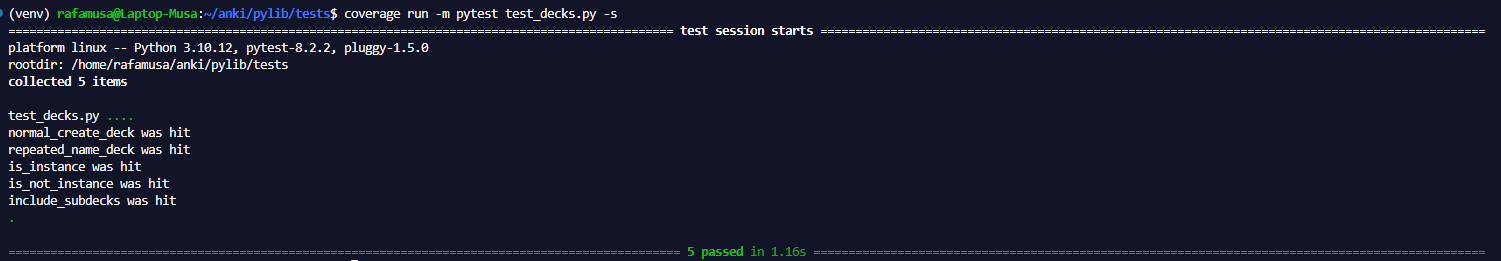




<Provide a screenshot of the new coverage results>







<State the coverage improvement with a number and elaborate on why the coverage is improved>

Coverage went from 0% to 100% in both the functions because none of them were accessed on any test before. Just by accessing them I could make big progress on improving the coverage, however some conditions had to be met so I could get the full coverage of that function. The function DeckManager.add\_normal\_deck\_with\_name had a condition of creating a deck with a repeated name. The other function DeckManager.card\_count had some conditions regarding how the deck parameter was called and if it should include its subdecks. I simply called those functions multiple times each time going through one of those conditions.

Vlad Parau

<Test 1>

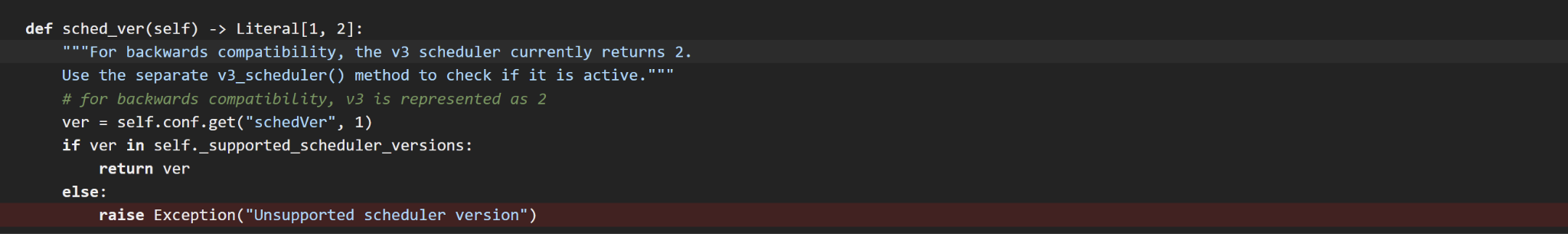
test\_collection.py

<Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test>

https://github.com/ankitects/anki/commit/16be6169707fe02b991cef26d73e3b64c0f5616e

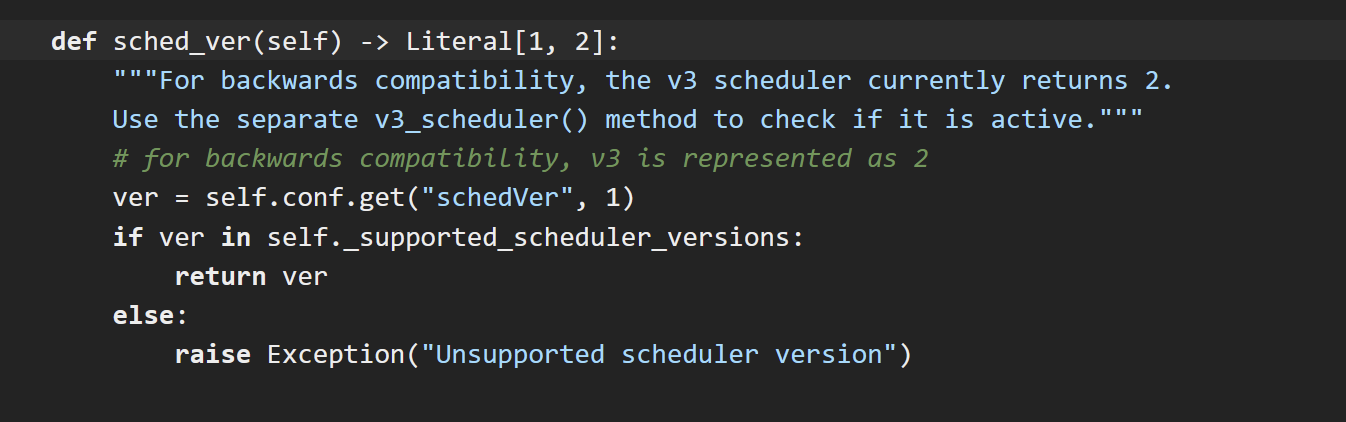
<Provide a screenshot of the old coverage results (the same as you already showed above)>





<Provide a screenshot of the new coverage results>





<State the coverage improvement with a number and elaborate on why the coverage is improved>

As illustrated by the screenshots above, the coverage improvement increased by 25%. The initial uncovered part was related to the branch reached only when sched\_ver() received an incompatible version of the app’s scheduler as input. In this case, the only compatible versions are 1 and 2. Since the first branch was already covered, I wrote a test for the branch that raised an exception. The test simply creates a configuration that has the third version(the only existing one that is incompatible) of the scheduler and calls sched\_ver(). More than that, it is built to expect an assertion. If the assertion isn’t raised, then there might be a problem with the code.

<Test 2>

test\_browser.py

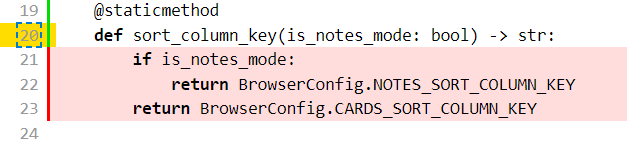
<Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test>

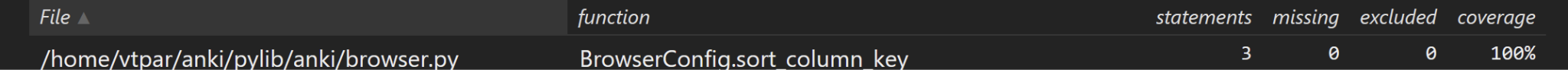
<Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test>

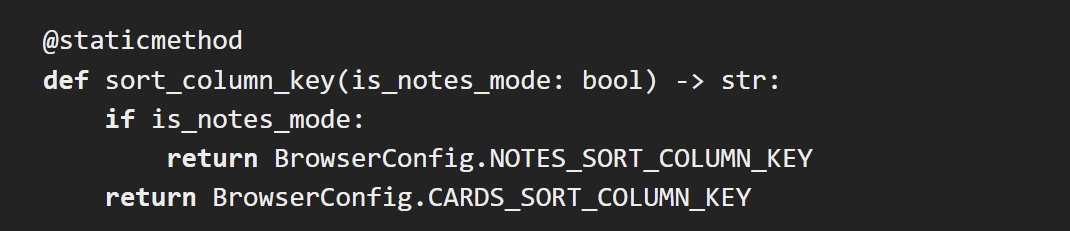
https://github.com/ankitects/anki/commit/a7813812749d55efe6dcdfac6106c64816af8824

<Provide a screenshot of the old coverage results (the same as you already showed above)>





<Provide a screenshot of the new coverage results> 



<State the coverage improvement with a number and elaborate on why the coverage is improved>

The coverage improvement increased by 100%. The reason why this happened is because none of the method’s branches were covered. The test behind this method is simple, I have just used the only two possible inputs that can occur, namely True and False. When the input is True, it means that the user is in “notes mode” and the test needs to check that this is indeed true, a functionality that was easily implemented using an assert. The other case occurs when the user is in “cards mode”, which will influence the sorting part differently, an aspect checked by another assert.

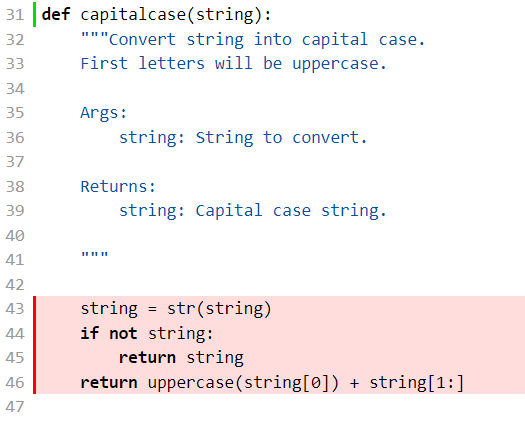
Isabela Aterman

<Test 1>

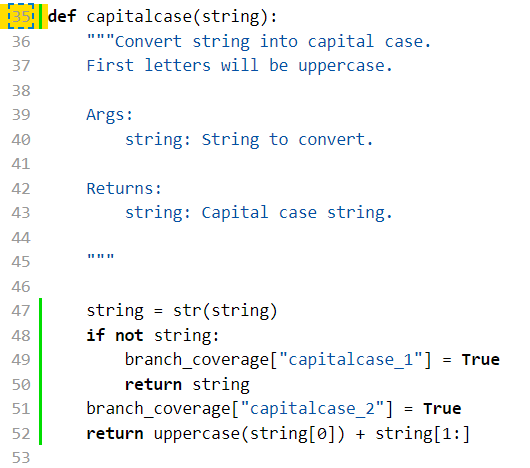
<Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test>

https://github.com/drfeelgood13/anki/commit/dd2dc9836b58607cc5d0ba575f772eb337a2050b

<Provide a screenshot of the old coverage results (the same as you already showed above)>



<Provide a screenshot of the new coverage results>



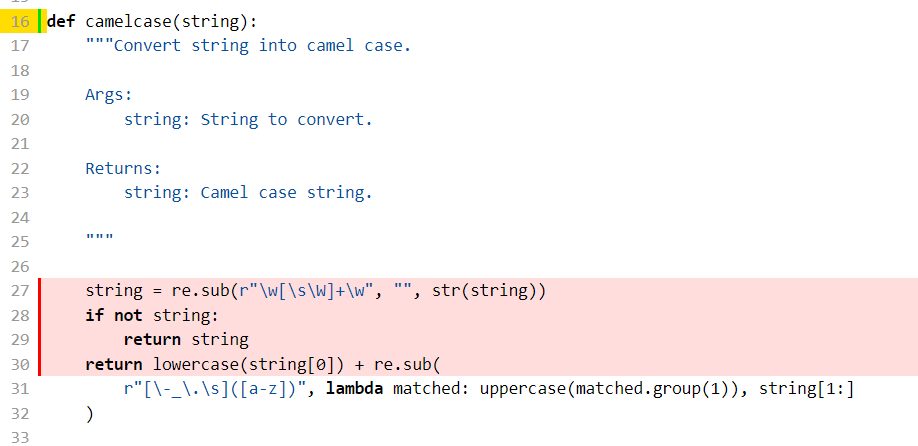


<State the coverage improvement with a number and elaborate on why the coverage is improved>

The coverage of the function went from 0% to 100%. It is improved because now every branch of the method is covered by tests. The chosen function transforms the first character of a string to uppercase, or returns immediately if the input string is empty. As mentioned before, the input that can occur is either an empty string, or a non-empty string. Thus, I created two asserts in which I check if the output of the function is as expected or not.

<Test 2>

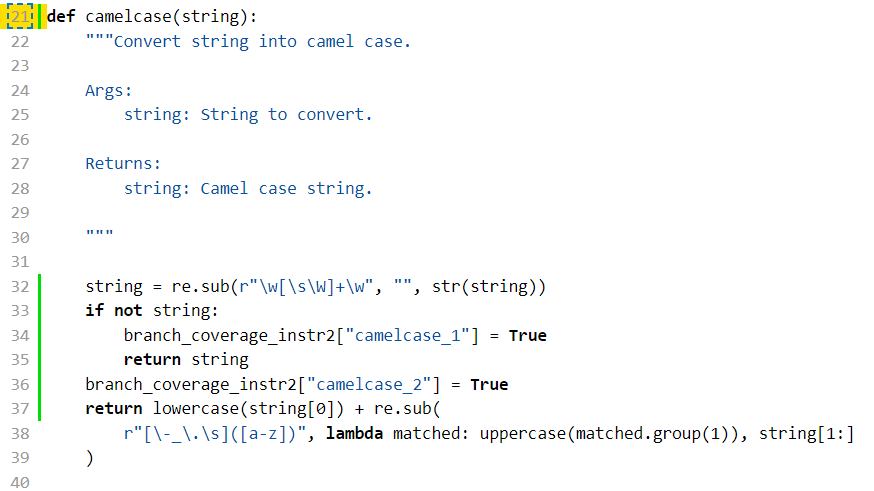
https://github.com/drfeelgood13/anki/commit/247fbe0d32928b7280e53d7c9eff536d0a39c1b2





New results:





The function had 0% coverage before. After I added tests for each branch, the coverage improved to 100%. The chosen method transforms strings to camel case, unless the input string is empty and it returns immediately. In order to test the branches, I created two asserts which check if the output of the function is as expected.

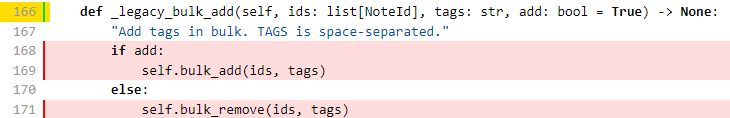
Lukas Jokubauskas

<Test 1>

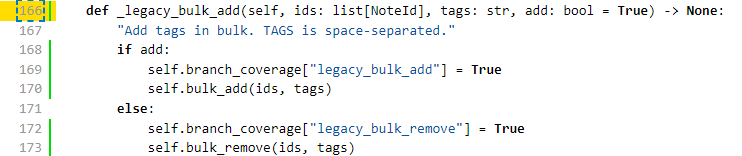
<Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test>

https://github.com/ankitects/anki/commit/ac9c6ac103117792340892757378ed0a1717c62c

<Provide a screenshot of the old coverage results (the same as you already showed above)>



<Provide a screenshot of the new coverage results>

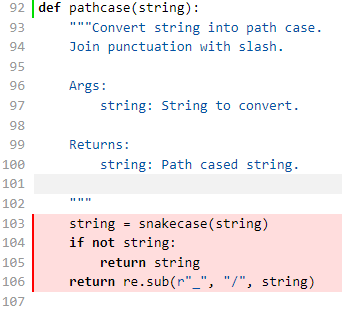


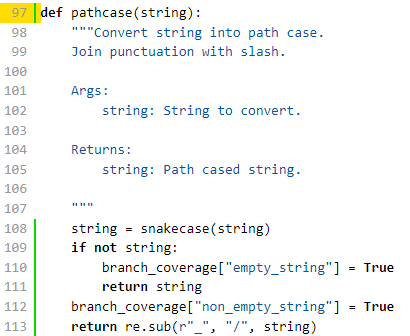
<State the coverage improvement with a number and elaborate on why the coverage is improved>

Coverage improved from 0% to 100%. It has improved because \_legacy\_bulk\_add function takes an array ids and tags and depending on boolean value it either adds multiple tags or removes them and I have tested the function with mock ids and tags both when boolean is true and when it is false and captured whether correct branches are entered in both cases. So all branches are tested now and none were before.

<Test 2>

https://github.com/ankitects/anki/commit/4432cc229aa721831c310e3e4ef19acb300fbbf2





Coverage improved from 0% to 100%. Coverage improved because both statements were tested, when the given string is empty and not empty. When given string is not empty it was successfully converted first to snakecase and the to pathcase which assertions confirm. Previously none of the statements were tested and now all statements are tested.

<Provide the same kind of information provided for Test 1>

### Overall

<Provide a screenshot of the old coverage results by running an existing tool (the same as you already showed above)>

<Provide a screenshot of the new coverage results by running the existing tool using all test modifications made by the group>

## Statement of individual contributions

<Write what each group member did>

Vlad improved the coverage of Collection.sched\_ver() and BrowserConfig.sort\_column\_key() and instrumented them at the same time. Besides this, he wrote his parts in the report and configured the environment for testing the methods.

Isabela improved the coverage of camelcase() and capitalcase() and instrumented them at the same time. Besides this, she wrote her parts in the report and the paragraph about the existing coverage tool that we used initially.

Rafael improved the coverage of DeckManager.add\_normal\_deck\_with\_name() and DeckManager.card\_count() and instrumented them at the same time.

Lukas instrumented and tested \_legacy\_bulk\_add() and pathcase() functions which improved their coverage to 100% and wrote his part of the report.