```python
#THE FOLLOWING CODE WAS USED TO TEST. COMMENTS DESCRIBING THE PROCESS
#ARE INCLUDED

import pytest
import tempfile
from importlib import reload
import os
import io
import sys
from p3 import SourceCode

# Overall description of changes to the test code

# Pytest was used as the primary testing tool.
# The format is similar to the one provided at https://github.com/CISC-CMPE-327/CI-Python
# The main change is that the package structure is gone for simplicity and less folders and instead there is a main directory with
# the source code that implements the front end along with the test code that runs the tests
# There are subdirectories for each test
# All of the subdirectories contain an input command file (with a stream of commands),
# an expected terminal output file, and a valid accounts file.
# For tests that need it, there is also an expected tsf file
# The test code looks inside these subdirectories and uses the files for a specific test to run the test
# There is no _main_ file as it is contained in SourceCode already
# Blank _init_ file was moved to the main p3 directory
# Test method 1 was adapted as from assignment 1 the format of our tests was in files

# The test cases are tested in a similar manner. Each has a test method that calls one of two helper functions
# and specifies the folder to find all of the files needed for it
# There are two helper methods now, one for tests that do not care about TSF output and ones that do
# init_file is present
# The vaf and tsf files in the main directory are for testing the program though manual commands without pytest
# by running python SourceCode.py vaf.txt tsf.txt in the command line

# Run the code by typing pytest in the command line

# get path
path = os.path.dirname(os.path.abspath(__file__))


# The following are all test methods that get run when pytest is called
# Some use helperNoTSF, some use helper. All pass their id to the helper
# Naming and general structure preserved from the template

def test_r1a(capsys):
    """Testing 1a. All required information stored in folder 1a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='1a'
    )

def test_r1b(capsys):
    """Testing 1b. All required information stored in folder 1b.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
```

```
        capsys=capsys,
        test_id='1b'
    )


def test_r2a(capsys):
    """Testing 2a. All required information stored in folder 2a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='2a'
    )


def test_r3a(capsys):
    """Testing 3a. All required information stored in folder 3a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='3a'
    )


def test_r3b(capsys):
    """Testing 3b. All required information stored in folder 3b.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='3b'
    )


def test_r3c(capsys):
    """Testing 3c. All required information stored in folder 3c.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='3c'
    )


def test_r4a(capsys):
    """Testing 4a. All required information stored in folder 4a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='4a'
    )


def test_r4b(capsys):
    """Testing 4b. All required information stored in folder 4b.
```

```
129
130         Arguments:
131             capsys -- object created by pytest to capture stdout and stderr
132         """
133         helperNoTSF(
134             capsys=capsys,
135             test_id='4b'
136         )
137
138
139     def test_r4c(capsys):
140         """Testing 4c. All required information stored in folder 4c.
141
142         Arguments:
143             capsys -- object created by pytest to capture stdout and stderr
144         """
145         helperNoTSF(
146             capsys=capsys,
147             test_id='4c'
148         )
149
150
151     def test_r5a(capsys):
152         """Testing 5a. All required information stored in folder 5a.
153
154         Arguments:
155             capsys -- object created by pytest to capture stdout and stderr
156         """
157         helperNoTSF(
158             capsys=capsys,
159             test_id='5a'
160         )
161
162
163     def test_r5b(capsys):
164         """Testing 5b. All required information stored in folder 5b.
165
166         Arguments:
167             capsys -- object created by pytest to capture stdout and stderr
168         """
169         helperNoTSF(
170             capsys=capsys,
171             test_id='5b'
172         )
173
174
175     def test_r6a(capsys):
176         """Testing 6a. All required information stored in folder 6a.
177
178         Arguments:
179             capsys -- object created by pytest to capture stdout and stderr
180         """
181         helper(
182             capsys=capsys,
183             test_id='6a'
184         )
185
186
187     def test_r7a(capsys):
188         """Testing 7a. All required information stored in folder 7a.
189
190         Arguments:
191             capsys -- object created by pytest to capture stdout and stderr
192         """
193         helperNoTSF(
194             capsys=capsys,
195             test_id='7a'
```

```
    )


def test_r7b(capsys):
    """Testing 7b. All required information stored in folder 7b.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='7b'
    )



def test_r8a(capsys):
    """Testing 8a. All required information stored in folder 8a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='8a'
    )


def test_r9a(capsys):
    """Testing 9a. All required information stored in folder 9a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='9a'
    )


def test_r9b(capsys):
    """Testing 9b. All required information stored in folder 9b.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='9b'
    )


def test_r9c(capsys):
    """Testing 9c. All required information stored in folder 9c.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='9c'
    )

def test_r9d(capsys):
    """Testing 9d. All required information stored in folder 9d.

    Arguments:
```

```
263             capsys -- object created by pytest to capture stdout and stderr
264         """
265         helper(
266             capsys=capsys,
267             test_id='9d'
268         )
269 
270     def test_r9e(capsys):
271         """Testing 9e. All required information stored in folder 9e.
272 
273         Arguments:
274             capsys -- object created by pytest to capture stdout and stderr
275         """
276         helper(
277             capsys=capsys,
278             test_id='9e'
279         )
280 
281     def test_r9f(capsys):
282         """Testing 9f. All required information stored in folder 9f.
283 
284         Arguments:
285             capsys -- object created by pytest to capture stdout and stderr
286         """
287         helper(
288             capsys=capsys,
289             test_id='9f'
290         )
291 
292     def test_r10a(capsys):
293         """Testing 10a. All required information stored in folder 10a.
294 
295         Arguments:
296             capsys -- object created by pytest to capture stdout and stderr
297         """
298         helper(
299             capsys=capsys,
300             test_id='10a'
301         )
302 
303     def test_r10b(capsys):
304         """Testing 10b. All required information stored in folder 10b.
305 
306         Arguments:
307             capsys -- object created by pytest to capture stdout and stderr
308         """
309         helper(
310             capsys=capsys,
311             test_id='10b'
312         )
313 
314     def test_r10c(capsys):
315         """Testing 10c. All required information stored in folder 10c.
316 
317         Arguments:
318             capsys -- object created by pytest to capture stdout and stderr
319         """
320         helper(
321             capsys=capsys,
322             test_id='10c'
323         )
324 
325     def test_r10e(capsys):
326         """Testing 10e. All required information stored in folder 10e.
327 
328         Arguments:
329             capsys -- object created by pytest to capture stdout and stderr
```

```
330         """
331         helper(
332             capsys=capsys,
333             test_id='10e'
334         )
335 
336     def test_r11a(capsys):
337         """Testing 11a. All required information stored in folder 11a.
338 
339         Arguments:
340             capsys -- object created by pytest to capture stdout and stderr
341         """
342         helper(
343             capsys=capsys,
344             test_id='11a'
345         )
346 
347     def test_r11b(capsys):
348         """Testing 11b. All required information stored in folder 11b.
349 
350         Arguments:
351             capsys -- object created by pytest to capture stdout and stderr
352         """
353         helper(
354             capsys=capsys,
355             test_id='11b'
356         )
357 
358     def test_r12a(capsys):
359         """Testing 12a. All required information stored in folder 12a.
360 
361         Arguments:
362             capsys -- object created by pytest to capture stdout and stderr
363         """
364         helper(
365             capsys=capsys,
366             test_id='12a'
367         )
368 
369     def test_r13a(capsys):
370         """Testing 13a. All required information stored in folder 13a.
371 
372         Arguments:
373             capsys -- object created by pytest to capture stdout and stderr
374         """
375         helper(
376             capsys=capsys,
377             test_id='13a'
378         )
379 
380     def test_r13b(capsys):
381         """Testing 13b. All required information stored in folder 13b.
382 
383         Arguments:
384             capsys -- object created by pytest to capture stdout and stderr
385         """
386         helperNoTSF(
387             capsys=capsys,
388             test_id='13b'
389         )
390 
391     def test_r14a(capsys):
392         """Testing 14a. All required information stored in folder 14a.
393 
394         Arguments:
395             capsys -- object created by pytest to capture stdout and stderr
396         """
```

```
397         helper(
398             capsys=capsys,
399             test_id='14a'
400         )
401 
402     def test_r14b(capsys):
403         """Testing 14b. All required information stored in folder 14b.
404 
405         Arguments:
406             capsys -- object created by pytest to capture stdout and stderr
407         """
408         helperNoTSF(
409             capsys=capsys,
410             test_id='14b'
411         )
412 
413     def test_r15a(capsys):
414         """Testing 15a. All required information stored in folder 15a.
415 
416         Arguments:
417             capsys -- object created by pytest to capture stdout and stderr
418         """
419         helper(
420             capsys=capsys,
421             test_id='15a'
422         )
423 
424     def test_r15b(capsys):
425         """Testing 15b. All required information stored in folder 15b.
426 
427         Arguments:
428             capsys -- object created by pytest to capture stdout and stderr
429         """
430         helper(
431             capsys=capsys,
432             test_id='15b'
433         )
434 
435     def test_r16a(capsys):
436         """Testing 16a. All required information stored in folder 16a.
437 
438         Arguments:
439             capsys -- object created by pytest to capture stdout and stderr
440         """
441         helper(
442             capsys=capsys,
443             test_id='16a'
444         )
445 
446     def test_r16b(capsys):
447         """Testing 16b. All required information stored in folder 16b.
448 
449         Arguments:
450             capsys -- object created by pytest to capture stdout and stderr
451         """
452         helperNoTSF(
453             capsys=capsys,
454             test_id='16b'
455         )
456 
457     def test_r17a(capsys):
458         """Testing 17a. All required information stored in folder 17a.
459 
460         Arguments:
461             capsys -- object created by pytest to capture stdout and stderr
462         """
463         helper(
```

```
464             capsys=capsys,
465             test_id='17a'
466         )
467
468     def test_r17b(capsys):
469         """Testing 17b. All required information stored in folder 17b.
470
471         Arguments:
472             capsys -- object created by pytest to capture stdout and stderr
473         """
474         helperNoTSF(
475             capsys=capsys,
476             test_id='17b'
477         )
478
479     def test_r17c(capsys):
480         """Testing 17c. All required information stored in folder 17c.
481
482         Arguments:
483             capsys -- object created by pytest to capture stdout and stderr
484         """
485         helperNoTSF(
486             capsys=capsys,
487             test_id='17c'
488         )
489
490     def test_r18a(capsys):
491         """Testing 18a. All required information stored in folder 18a.
492
493         Arguments:
494             capsys -- object created by pytest to capture stdout and stderr
495         """
496         helper(
497             capsys=capsys,
498             test_id='18a'
499         )
500
501     def test_r18b(capsys):
502         """Testing 18b. All required information stored in folder 18b.
503
504         Arguments:
505             capsys -- object created by pytest to capture stdout and stderr
506         """
507         helperNoTSF(
508             capsys=capsys,
509             test_id='18b'
510         )
511
512     def test_r18c(capsys):
513         """Testing 18c. All required information stored in folder 18c.
514
515         Arguments:
516             capsys -- object created by pytest to capture stdout and stderr
517         """
518         helperNoTSF(
519             capsys=capsys,
520             test_id='18c'
521         )
522
523     def test_r19a(capsys):
524         """Testing 19a. All required information stored in folder 19a.
525
526         Arguments:
527             capsys -- object created by pytest to capture stdout and stderr
528         """
529         helper(
530             capsys=capsys,
```

```
            test_id='19a'
        )

    def test_r19b(capsys):
        """Testing 19b. All required information stored in folder 19b.

        Arguments:
            capsys -- object created by pytest to capture stdout and stderr
        """
        helper(
            capsys=capsys,
            test_id='19b'
        )

    def test_r20a(capsys):
        """Testing 20a. All required information stored in folder 20a.

        Arguments:
            capsys -- object created by pytest to capture stdout and stderr
        """
        helper(
            capsys=capsys,
            test_id='20a'
        )

    def test_r20b(capsys):
        """Testing 20b. All required information stored in folder 20b.

        Arguments:
            capsys -- object created by pytest to capture stdout and stderr
        """
        helperNoTSF(
            capsys=capsys,
            test_id='20b'
        )

    def test_r21a(capsys):
        """Testing 21a. All required information stored in folder 21a.

        Arguments:
            capsys -- object created by pytest to capture stdout and stderr
        """
        helper(
            capsys=capsys,
            test_id='21a'
        )

    def test_r21b(capsys):
        """Testing 21b. All required information stored in folder 21b.

        Arguments:
            capsys -- object created by pytest to capture stdout and stderr
        """
        helperNoTSF(
            capsys=capsys,
            test_id='21b'
        )

    def test_r21c(capsys):
        """Testing 21c. All required information stored in folder 21c.

        Arguments:
            capsys -- object created by pytest to capture stdout and stderr
        """
        helperNoTSF(
            capsys=capsys,
            test_id='21c'
```

```
    )

def test_r22a(capsys):
    """Testing 22a. All required information stored in folder 22a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='22a'
    )

def test_r22b(capsys):
    """Testing 22b. All required information stored in folder 22b.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='22b'
    )

def test_r22c(capsys):
    """Testing 22c. All required information stored in folder 22c.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='22c'
    )

def test_r23a(capsys):
    """Testing 23a. All required information stored in folder 23a.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='23a'
    )

def test_r23b(capsys):
    """Testing 23b. All required information stored in folder 23b.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helper(
        capsys=capsys,
        test_id='23b'
    )

def test_r24a(capsys):
    """Testing r2. All required information stored in folder r2.

    Arguments:
        capsys -- object created by pytest to capture stdout and stderr
    """
    helperNoTSF(
        capsys=capsys,
        test_id='24a'
    )
```

```
665
666     def test_r24b(capsys):
667         """Testing r2. All required information stored in folder r2.
668
669         Arguments:
670             capsys -- object created by pytest to capture stdout and stderr
671         """
672         helperNoTSF(
673             capsys=capsys,
674             test_id='24b'
675         )
676
677     def test_r25a(capsys):
678         """Testing r2. All required information stored in folder r2.
679
680         Arguments:
681             capsys -- object created by pytest to capture stdout and stderr
682         """
683         helperNoTSF(
684             capsys=capsys,
685             test_id='25a'
686         )
687
688     def test_r25b(capsys):
689         """Testing r2. All required information stored in folder r2.
690
691         Arguments:
692             capsys -- object created by pytest to capture stdout and stderr
693         """
694         helperNoTSF(
695             capsys=capsys,
696             test_id='25b'
697         )
698
699     def test_r26a(capsys):
700         """Testing r2. All required information stored in folder r2.
701
702         Arguments:
703             capsys -- object created by pytest to capture stdout and stderr
704         """
705         helperNoTSF(
706             capsys=capsys,
707             test_id='26a'
708         )
709
710     def test_r26b(capsys):
711         """Testing r2. All required information stored in folder r2.
712
713         Arguments:
714             capsys -- object created by pytest to capture stdout and stderr
715         """
716         helperNoTSF(
717             capsys=capsys,
718             test_id='26b'
719         )
720
721     def test_r26c(capsys):
722         """Testing r2. All required information stored in folder r2.
723
724         Arguments:
725             capsys -- object created by pytest to capture stdout and stderr
726         """
727         helperNoTSF(
728             capsys=capsys,
729             test_id='26c'
730         )
731
```

```
732   def test_r26d(capsys):
733       """Testing r2. All required information stored in folder r2.
734
735       Arguments:
736           capsys -- object created by pytest to capture stdout and stderr
737       """
738       helperNoTSF(
739           capsys=capsys,
740           test_id='26d'
741       )
742
743   def test_r27a(capsys):
744       """Testing r2. All required information stored in folder r2.
745
746       Arguments:
747           capsys -- object created by pytest to capture stdout and stderr
748       """
749       helperNoTSF(
750           capsys=capsys,
751           test_id='27a'
752       )
753
754   def test_r27b(capsys):
755       """Testing r2. All required information stored in folder r2.
756
757       Arguments:
758           capsys -- object created by pytest to capture stdout and stderr
759       """
760       helperNoTSF(
761           capsys=capsys,
762           test_id='27b'
763       )
764
765   def test_r28a(capsys):
766       """Testing r2. All required information stored in folder r2.
767
768       Arguments:
769           capsys -- object created by pytest to capture stdout and stderr
770       """
771       helper(
772           capsys=capsys,
773           test_id='28a'
774       )
775
776   def test_r28b(capsys):
777       """Testing r2. All required information stored in folder r2.
778
779       Arguments:
780           capsys -- object created by pytest to capture stdout and stderr
781       """
782       helper(
783           capsys=capsys,
784           test_id='28b'
785       )
786
787
788
789   def r32a(capsys):
790       """Testing 32a. All required information stored in folder 32a.
791
792       Arguments:
793           capsys -- object created by pytest to capture stdout and stderr
794       """
795       helperNoTSF(
796           capsys=capsys,
797           test_id='32a'
798       )
```

```
799
800     def test_r33a(capsys):
801         """Testing 33a. All required information stored in folder 33a.
802
803         Arguments:
804             capsys -- object created by pytest to capture stdout and stderr
805         """
806         helperNoTSF(
807             capsys=capsys,
808             test_id='33a'
809         )
810     def test_r33b(capsys):
811         """Testing 33b. All required information stored in folder 33b.
812
813         Arguments:
814             capsys -- object created by pytest to capture stdout and stderr
815         """
816         helperNoTSF(
817             capsys=capsys,
818             test_id='33b'
819         )
820     def test_r34a(capsys):
821         """Testing 34a. All required information stored in folder 34a.
822
823         Arguments:
824             capsys -- object created by pytest to capture stdout and stderr
825         """
826         helperNoTSF(
827             capsys=capsys,
828             test_id='34a'
829         )
830     def r34b(capsys):
831         """Testing 34b. All required information stored in folder 34b.
832
833         Arguments:
834             capsys -- object created by pytest to capture stdout and stderr
835         """
836         helperNoTSF(
837             capsys=capsys,
838             test_id='34b'
839         )
840
841     def test_r35a(capsys):
842         """Testing 35a. All required information stored in folder 35a.
843
844         Arguments:
845             capsys -- object created by pytest to capture stdout and stderr
846         """
847         helperNoTSF(
848             capsys=capsys,
849             test_id='35a'
850         )
851     def test_r35b(capsys):
852        """Testing 35b. All required information stored in folder 35b.
853
854        Arguments:
855            capsys -- object created by pytest to capture stdout and stderr
856        """
857        helperNoTSF(
858            capsys=capsys,
859            test_id='35b'
860        )
861
862     # Helper function for tests with tsf
863     # Changes from the template are noted at specific points
864
865     def helper(
```

```
866             capsys,
867             test_id):
868         """ a helper function that test requirements for the example app
869 
870         Arguments:
871             capsys -- object created by pytest to capture stdout and stderr
872         """
873 
874         # Change from template: There's no need to clean the package since there is no
            package
875 
876         # locate test case folder:
877         case_folder = os.path.join(path, test_id)
878         # Change from template:
879         # The following was changed from template since test files follow the format of ex.
            1a for input,
880         # o1a for expected output, o1a_TSF for tsf
881         # The code generates the file names based on the test name rather than using the
            same file names for all tests
882         # or manually having to enter the name for each test
883 
884         # concatenate test_id with .txt
885         test_id_txt = test_id + ".txt"
886 
887         # concatenate test_id_txt with o
888         out_test_id_txt = "o" + test_id_txt
889 
890         # concatenate test_id with o, _TSF and .txt
891         out_tsf_test_id_txt = "o" + test_id + "_TSF.txt"
892 
893         # read terminal input:
894         with open(
895             os.path.join(
896                 case_folder, test_id_txt)) as rf:
897             terminal_input = rf.read().splitlines()
898 
899         # read expected tail portion of the terminal output:
900         with open(
901             os.path.join(
902                 case_folder, out_test_id_txt)) as rf:
903             terminal_output_tail = rf.read().splitlines()
904 
905         # create a temporary file in the system to store output transactions
906         temp_fd, temp_file = tempfile.mkstemp()
907         transaction_summary_file = temp_file
908 
909 
910         # prepare program parameters
911         sys.argv = ['SourceCode.py',
912             os.path.join(case_folder, 'vaf.txt'),
913             transaction_summary_file]
914 
915         # set terminal input
916         sys.stdin = io.StringIO(
917             '\n'.join(terminal_input))
918 
919         # Runs the main function of our program (thats run in _main_). Not called app.
920 
921         SourceCode.main()
922 
923         # capture terminal output / errors
924         # assuming that in this case we don't use stderr
925         out, err = capsys.readouterr()
926 
927         # split terminal output in lines
928         out_lines = out.splitlines()
929 
```

```
930         # Changes from template:
931         # For easier understanding of results, we print the line numbers, and expected and
            current outputs
932         # upon an assertion error to see the results more clearly (gives the complete
            lines) for easier debugging
933         # We also print a label for the expected output so that we know which one is which
            when doing comparisons
934
935         # compare terminal outputs at the end.`
936         for i in range(1, len(terminal_output_tail)+1):
937             index = i * -1
938             print(index)
939             print("What we expect")
940             print(terminal_output_tail[index])
941             print(out_lines[index])
942             assert terminal_output_tail[index] == out_lines[index]
943
944         # compare transactions:
945         with open(transaction_summary_file, 'r') as of:
946             content = of.read()
947             with open(os.path.join(case_folder, out_tsf_test_id_txt), 'r') as exp_file_of:
948                 expected_content = exp_file_of.read()
949                 print("Content")
950                 print(content)
951                 print("expected")
952                 print(expected_content)
953                 assert content == expected_content
954
955         # clean up
956         os.close(temp_fd)
957         os.remove(temp_file)
958
959     # Functionally identically to the above but lacking the part under compare transactions
        since there is no tsf for these cases to compare
960     # Created since not all of our tests use it.
961
962     def helperNoTSF(
963             capsys,
964             test_id):
965         """ a helper function that test requirements for the example app
966
967         Arguments:
968             capsys -- object created by pytest to capture stdout and stderr
969         """
970
971         # locate test case folder:
972         case_folder = os.path.join(path, test_id)
973
974         # concatenate test_id with .txt
975         test_id_txt = test_id + ".txt"
976
977         # concatenate test_id_txt with o
978         out_test_id_txt = "o" + test_id_txt
979
980         # read terminal input:
981         with open(
982                 os.path.join(
983                     case_folder, test_id_txt)) as rf:  # REPLACE NAME HERE
984             terminal_input = rf.read().splitlines()
985
986         # read expected tail portion of the terminal output:
987         with open(
988                 os.path.join(
989                     case_folder, out_test_id_txt)) as rf:  # REPLACE NAME HERE
990             terminal_output_tail = rf.read().splitlines()
991
992         # create a temporary file in the system to store output transactions
```

```
        temp_fd, temp_file = tempfile.mkstemp()
        transaction_summary_file = temp_file

        # prepare program parameters
        sys.argv = ['SourceCode.py',
                    os.path.join(case_folder, 'vaf.txt'),
                    transaction_summary_file]

        # set terminal input
        sys.stdin = io.StringIO(
            '\n'.join(terminal_input))

        # run the program
        # app.main()

        SourceCode.main()

        # capture terminal output / errors
        # assuming that in this case we don't use stderr
        out, err = capsys.readouterr()

        # split terminal output in lines
        out_lines = out.splitlines()

        # compare terminal outputs at the end.`
        for i in range(1, len(terminal_output_tail) + 1):
            index = i * -1
            print(index)
            print("What we expect")
            print(terminal_output_tail[index])
            print(out_lines[index])
            assert terminal_output_tail[index] == out_lines[index]

        # clean up
        os.close(temp_fd)
        os.remove(temp_file)
```

Test Case	What It's Testing	Success (Y/N)	Explanation	Fixed By
1a	Test to see that login is accepted	Y	Behaved as expected	
1b	Test to see that no input other than login is accepted	Y	Behaved as expected	
2a	Test to see that no subsequent login is accepted after the system is already logged in	Y	Behaved as expected	
3a	Test to see that privileged transactions are not accepted in ATM.	Y	Behaved as expected	
3b	Test to see that unprivileged actions are accepted in ATM.	N	Expected EOS output was incorrect although the code was correct - EOS format was thought to be XFR AccOut Amount AccIn instead of XFR AccIn Amount AccOut	The expected EOS output file was changed so that the transfer EOS code followed the correct format
3b	Test to see that unprivileged actions are accepted in ATM.	N	Code produced EOS of WDR 0000000 Amount Account rather than WDR Account Amount 0000000	The expected EOS output file was changed so that the withdraw EOS code followed the correct format
3b	Test to see that unprivileged actions are accepted in ATM.	Y	Behaved as expected	
3c	Test to see that invalid input is denied in ATM.	Y	Behaved as expected	
4a	Test to see that all valid transactions are accepted in Agent.	Y	Behaved as expected	
4b	Test to see that all invalid transactions are not accepted in Agent.	Y	Behaved as expected	
4c	Test to see that invalid input is denied in Agent.	Y	Behaved as expected	
5a	Test to see that the only session input accepted is ATM or agent	N	Input command file and console output file had blank new lines separating separate blocks of inputs/outputs	The new lines were removed
5a	Test to see that the only session input accepted is ATM or agent	Y	Behaved as expected	
5b	Test to see that the invalid input is not accepted.	N	Expected console output file had the wrong information (login instead of Status: login) so the expected output did not match the actual output.	Replaced all instances of the faulty lines in the expected console output file, o5b.txt, from "login" to "Status: login"
5b	Test to see that the invalid input is not accepted.	Y	Behaved as expected	
6a	Test to see if accounts file is read and and transactions are possible after type is accepted	Y	Behaved as expected	
7a	Test to see that login is accepted after a logout	Y	Behaved as expected	
7b	Test to see that no input is accepted after a logout, other than login	Y	Behaved as expected	
8a	Test to see if a TSF is produced after log out	Y	Behaved as expected	
9a	Test to see that valid account number is accepted.	Y	Behaved as expected	
9b	Test to see that account number beginning with zero is rejected.	Y	Behaved as expected	
9c	Test to see that account number without 7 digits is rejected.	Y	Behaved as expected	
9d	Test to see that account number with alphabetical characters is rejected.	Y	Behaved as expected	
9e	Test to see that account number with spaces is rejected.	Y	Behaved as expected	
9f	Test to see if account number is the same as an existing account number	Y	Behaved as expected	
10a	Test to see if the account name is between 3 and 30 alphanumeric characters.	Y	Behaved as expected	
10b	Test to see if the account name has less than 3 characters.	Y	Behaved as expected	
10c	Test to see if the account name has more than 30 characters.	Y	Behaved as expected	
10d	Test to see if name is the same as an existing name.	Y	Behaved as expected	
10e	Test to see if name begins or ends with a space.	N	Account name is allowed to start/end with a space/blank character. The code allows the name if it does not start with a blank character OR it does not end with a blank character.	Changing the if statement so that it only allows the name if it does not start AND does not end with a blank character
10e	Test to see if name begins or ends with a space.	Y	Behaved as expected	
11a	Test to see if the account can only be made when session is set to Agent	Y	Behaved as expected	
11b	Test to see if the account cannot be made when session is not set to Agent	Y	Behaved as expected	
12a	Test to see if a transaction is not accepted in the new account during the same session it is created in	Y	Behaved as expected	
13a	Test to see if the account can only be deleted when session is set to Agent	Y	Behaved as expected	
13b	Test to see if the account cannot be deleted when session is not set to Agent	N	A line in the input command file, 13b.txt, was set incorrectly - it had "deleteacct" instead of "deleteAccount"	Changed "deleteacct" to "deleteAccount"
13b	Test to see if the account cannot be deleted when session is not set to Agent	Y	Behaved as expected	
14a	Test to see if delete occurs when both account number and account name exist	Y	Behaved as expected	
14b	Test to see if delete does not occur when either account number or name do not exist	Y	Behaved as expected	
15a	Test to see if createacct is accepted on a deleted account	N	A line in the input command file, 15a.txt, was set incorrectly - it had "createacct" instead of "createAccount"	Change "createacct" to "createAccount"
15a	Test to see if createacct is accepted on a deleted account	Y	Behaved as expected	
15b	Test to see if a transaction (other than createacct) is accepted on a deleted account	Y	Behaved as expected	
16a	Test to see if account number exists	Y	Behaved as expected	
16b	Test to see if account number does not exist	Y	Behaved as expected	
17a	Test to see if the amount is valid	Y	Behaved as expected	
17b	Test to see if there are spaces, characters, symbols in the desired amount and deny transaction	N	Expected input was set incorrectly - had accNum instead of a hardcoded account number from the existing account files in the vaf.txt	Changed "accNum" to "1234567", which is an existing account number in the vaf.txt
17b	Test to see if there are spaces, characters, symbols in the desired amount and deny transaction	Y	Behaved as expected	
17c	Test to see if the amount is not within the limit (between 0-200000)	N	Expected input was set incorrectly - had accNum instead of a hardcoded account number from the existing account files in the vaf.txt	Changed "accNum" to "1234567", which is an existing account number in the vaf.txt
17c	Test to see if the amount is not within the limit (between 0-200000)	Y	Behaved as expected	
18a	Test to see if the amount is valid	Y	Behaved as expected	
18b	Test to see if there are spaces, characters, symbols in the desired amount and deny transaction	Y	Behaved as expected	
18c	Test to see if the amount is not within the limit (between 0-99999999)	Y	Behaved as expected	
19a	Test to see if the combined daily deposit is within $5000 per account in ATM	Y	Behaved as expected (backend currently non-existent)	
19b	Test to see if the combined daily deposit is beyond $5000 per account in ATM	Y	Behaved as expected (backend currently non-existent)	
20a	Test to see if account number exists	Y	Correct results but assertion failed due to newline character in tsf output	
20b	Test to see if account number does not exist	Y	Behaved as expected	
21a	Test to see if the amount is valid	Y	Behaved as expected	
21b	Test to see if there are spaces, characters, symbols in the desired amount and deny transaction	Y	Behaved as expected	
21c	Test to see if the amount is not within the limit (between 0-100000)	Y	Behaved as expected	
22a	Test to see if the amount is valid	Y	Behaved as expected	
22b	Test to see if there are spaces, characters, symbols in the desired amount and deny transaction	Y	Behaved as expected	
22c	Test to see if the amount is not within the limit (between 0-99999999)	Y	Behaved as expected	
23a	Test to see if the combined daily withdraw is within $5000 per account in ATM	Y	Behaved as expected (backend currently non-existent)	
23b	Test to see if the combined daily withdraw is beyond $5000 per account in ATM	Y	Behaved as expected (backend currently non-existent)	
24a	Test to see if account number exists	Y	Behaved as expected	
24b	Test to see if account number does not exist	N	The transfer function had swapped label for accountIn and accountFrom	fixing the swapped mistake between accountFrom and accountTo
24b	Test to see if account number does not exist	Y	Behaved as expected	
25a	Test to see if the amount is valid	Y	Behaved as expected	
25b	Test to see if the amount is not valid	Y	Behaved as expected	
26a	Test to see if the transfer amount is within the limit for ATM (between 0 - 1000000)	N	The limits set for the transfer amount were not correct in the source code	The transfer amount limits were corrected
26a	Test to see if the transfer amount is within the limit for ATM (between 0 - 1000000)	Y	Behaved as expected	

26b	Test to see if the transfer amount is beyond the limit for ATM (between 0 - 1000000)	Y	Behaved as expected	
26c	Test to see if the transfer amount is within the limit for Agent (between 0 - 9999999)	N	The limits set for the transfer amount were not correct in the source code	The transfer amount limits were corrected
26c	Test to see if the transfer amount is within the limit for Agent (between 0 - 9999999)	Y	Behaved as expected	
26d	Test to see if the transfer amount is beyond the limit for Agent (between 0 - 9999999)	Y	Behaved as expected	
27a	Test to see if the total transfer amount is within the daily limit (1000000 in ATM)	N	Mistake in test case input	Removed extra space from test input
27a	Test to see if the total transfer amount if within the daily limit (1000000 in ATM)	Y	Behaved as expected	
27b	Test to see if the total transfer amount is beyond the daily limit (1000000 in ATM)	N	Mistake in test case input	Removed extra space from test input
27b	Test to see if the total transfer amount is beyond the daily limit (1000000 in ATM)	Y	Behaved as expected	
28a	Test to see if the accounts during a transfer are different accounts	Y	Behaved as expected	
28b	Test to see that accounts that are the same during a transfer transaction is rejected	Y	Behaved as expected	
29a	Test to see if the transaction message follows the format as stated in the constraints	Y	Behaved as expected	
30a	day	Y	Behaved as expected	
31a	Test to see that the file ends with the end of session transaction code	Y	Behaved as expected	
32a	Test to see if the system still works after 20 consecutive sessions.	Y	No crashes after 20 consecutive changes and all results are as expected	
33a	Test to see if time complexity of account list is constant.	Y	Accounts file consisting of 100000 names was used. Test accessing first account took 0.45s. Test accessing last account took 0.44s, therefore accessing accounts is constant time.	
34a	Test to see if there is exactly one space between inputs on a line.	Y	Behaved as expected	
34b	Test to see if there is less or more than one space between inputs on a line.	N	Input is still invalid but the error message is different than expected. Functionally equivalent: -invalid input vs +account number must be 7 decimals long	Changing error message in test case
34b	Test to see if there is less or more than one space between inputs on a line.	Y	Behaved as expected	
35a	Test to make sure entering AGENT is not allowed when ATM is active.	Y	Behaved as expected	
35b	Test to make sure entering ATM is not allowed when AGENT is active.	Y	Behaved as expected	