A2 Preliminary harness instructions

Instructions

Get the correct version:
- If you are using a computer with an Intel-based CPU, download the file `test2preIntel.zip`
- If you are using a computer with an ARM-based CPU - e.g. a Mac with an M1 - M4 chip - download the file `test2preARM.zip`

If you are not sure, check the specifications of your computer online.

Use your Makefile to create a shared library file `libvcparser.so`, as per Assignment 2 instructions, and place it in the same directory as the `test2pre` executable. Remember to use the original, unmodified versions of `LinkedListAPI.h` and `VCParser.h` - they have been provided for your reference in the assignment description.

To test your code, run the executable `test2pre`. You should see a long list of test cases, with the max displayed score of 55. The executables were compiled using the CIS*2750 Docker container on Intel- and ARM-based computers for you, so you must run this executable in the CIS*2750 Docker container. Needless to say, your library file must be compiled there as well.

The output for the code that passes all tests will look like this:

```
ASSIGNMENT 2 TESTING - PRELIMINARY TESTS

Test 1 (2%): writeCard (NULL arguments): PASSED 2/2 tests
    SUCCESS: Subtest 1.1: Successfully handled NULL file name.
    SUCCESS: Subtest 1.2: Successfully handled NULL Card object.

Test 2 (4%): Testing writeCard. Creating a .vcf file from a valid reference (Minimum valid Card object): PASSED 1/1 tests
    SUCCESS: Subtest 2.1: correctly saved valid Card object

Test 3 (4%): Testing writeCard. Creating a .vcf file from a valid reference (Valid Card object with multiple properties and multiple property values): PASSED 1/
1 tests
    SUCCESS: Subtest 3.1: correctly saved valid Card object

Test 4 (4%): Testing writeCard. Creating a .vcf file from a valid reference (Minimum valid Card object with a text birthday): PASSED 1/1 tests
    SUCCESS: Subtest 4.1: correctly saved valid Card object

Test 5 (4%): Testing writeCard. Creating a .vcf file from a valid reference (Minimum valid Card object with a date-only birthday): PASSED 1/1 tests
    SUCCESS: Subtest 5.1: correctly saved valid Card object

Test 6 (4%): Testing writeCard. Creating a .vcf file from a valid reference (Minimum valid Card object with a time-only birthday): PASSED 1/1 tests
    SUCCESS: Subtest 6.1: correctly saved valid Card object

Test 7 (4%): Testing validateCard (expected INV_CARD): NULL card: PASSED 1/1 tests
    SUCCESS: Subtest 7.1: successfuly validated

Test 8 (5%): Testing validateCard (expected INV_CARD): 1 - NULL fn, 2 - NULL otherProperties, 3 - VERSION property in otherProperties: PASSED 3/3 tests
    SUCCESS: Subtest 8.1: successfuly validated
    SUCCESS: Subtest 8.2: successfuly validated
    SUCCESS: Subtest 8.3: successfuly validated

Test 9 (4%): Testing validateCard (expected INV_PROP) on object with a missing property value: PASSED 1/1 tests
    SUCCESS: Subtest 9.1: successfuly validated

Test 10 (4%): Testing validateCard (expected INV_PROP) on object where KIND property appears twice (cardinality violation): PASSED 1/1 tests
    SUCCESS: Subtest 10.1: successfuly validated

Test 11 (5%): Testing validateCard (expected INV_DT): 1 - invalid text DateTime (UTC flag true), 2 - invalid text DateTime (non-empty date string), 3 - invalid
text DateTime (non-empty time string), 4 - invalid non-text DateTime (non-empty text string): PASSED 4/4 tests
    SUCCESS: Subtest 11.1: successfuly validated
    SUCCESS: Subtest 11.2: successfuly validated
    SUCCESS: Subtest 11.3: successfuly validated
    SUCCESS: Subtest 11.4: successfuly validated

Test 12 (4%): Testing validateCard (expected INV_DT): 1 - BDAY and 2 - ANNIVERSARY in otherProperties: PASSED 2/2 tests
    SUCCESS: Subtest 12.1: successfuly validated
    SUCCESS: Subtest 12.2: successfuly validated

Test 13 (5%): Testing validateCard on valid objects containing DateTime properties (expected OK): PASSED 4/4 tests
    SUCCESS: Subtest 13.1: object successfuly validated
    SUCCESS: Subtest 13.2: object successfuly validated
    SUCCESS: Subtest 13.3: object successfuly validated
    SUCCESS: Subtest 13.4: object successfuly validated
Score: 55
```

If your code fails some test cases, the test harness score will be below 55 and the failed tests will get the appropriate feedback (and will be highlighted in red).

If you are having trouble running the harness, check the troubleshooting instructions for the A1 harness.

Harness details

The test harness validates the functionality of your assignment:
- For Module 1, each test case manually crates various valid Card structs, saves them to the disk using your `writeCard`, then uses this saved file to create a new Card using your `createCard`. The new Card is expected to match the reference Card.
- For Module 2, each test case passes various manually created invalid (and valid) Card structs to `validateCard` and verifies that the returned error codes match the expected ones.

Please note that the test harness is a testing tool, and not a debugging tool. It will not tell why your code fails a test - it simply tells that the output of your function does not match the expected output, and therefore does not match the assignment specification.

You are responsible for finding and fixing errors in your code. Note that the difference that would cause the code to fail would include mismatched strings, mismatched list lengths, etc..

Some suggestions:
- Start with the simplest files.
- Make sure that `writeCard` creates a correct file with correct line endings
- Make sure your own `createCard` can correctly open files you created with `writeCard`
- Make sure `validateCard` works correctly on your Card structs, both valid and invalid

As before, the test harness never tries to deallocate memory - apart from the tests for the `deleteCard` function - to minimize the chances of your code crashing during the tests. As a result, it will definitely leak a lot of memory, so do not try running it through `valgrind` in an effort to find memory bugs in your code. You will need to create your own tests for memory leaks and errors.

Because of this, the test harness does not include any tests for memory leaks or memory errors. Memory testing will be done by different software.