QE Interview homework

Assignment

Foobar is a (imaginary) program which loads file **/tmp/foobar** file as its configuration. It supports multiple options and the values for the options are assigned using the '**=**' sign (whitespaces around the equal sign are allowed). List of supported options:

- **Foo** - integer value
- **Bar** - string value
- **FooBar** - integer value

Foobar will load only the last occurrence of an option in the configuration file, ignoring the previous occurrences. Comments are also supported and can be included by adding '**#**' character at the beginning of the line. Example of a configuration file **/tmp/foobar**:

```
# Set Foo to 10.
Foo = 10

Bar="hello world"

FooBar=0
```

Task 1

The task is to implement the **foobar_foo_gte_10** Bash function which scans the **/tmp/foobar** configuration file and reports the result pass (returns 0) if the '**Foo**' option is greater than or equal to value 10, otherwise reports fail (returns 1). Skeleton of the function can be found below, feel free to create other helper functions as needed:

```
function foobar_foo_gte_10() {
    file="/tmp/foobar"
    opt="Foo"
    expected_value="10"
    assert_msg="Option '$opt' is greater than or equal to
'$expected_value' in '$file'"

    # This part needs to be implemented. Function should
    # find the $opt option in the $file file and check if
    # a value of the option is greater than or equal 10.
    if true; then
        echo "PASS - $assert_msg"
        return 0
    else
        echo "FAIL - $assert_msg"
        return 1
    fi
}
```

Example usage:

```
$ echo "Foo=10" > /tmp/foobar
$ foobar_foo_gte_10
PASS - Option 'Foo' is greater than or equal to '10' in
'/tmp/foobar'
$ echo $?
0

$ echo "Foo=5" > /tmp/foobar
$ foobar_foo_gte_10
FAIL - Option 'Foo' is greater than or equal to '10' in
'/tmp/foobar'
$ echo $?
1
```

Task 2

Test the implemented **foobar_foo_gte_10** Bash function reporting capabilities by creating test scenarios for it. Each test scenario should be a separate Bash function and should test a single use case. Be creative in this part and try to come up with test scenarios which would exercise various configurations of the '**Foo**' option in the **/tmp/foobar** configuration file.
The test script should prepare the system for the test execution (install any required packages, etc.) and then also undo all the committed changes including changes performed by the tests, therefore bringing the test system to the original state (the state in which the system was before running the test script).

Implementation

- Homework must be implemented using Bash and standard command-line utilities found in Linux like grep, sed, etc.
- Feel free to use any Linux distribution in your implementation or use a Linux container.

Interview

Present your solution of the homework during the interview including a short demo.
Please submit your code on a public GitHub or public GitLab repository and send a link to this repository to mmarhefk@redhat.com.