CS 279 - Homework 10

Deadline:

Due by 11:59 pm on **SUNDAY**, November 16.

How to submit:

Submit your files using `~ah270/279submit` on nrs-labs, with a homework number of `10`, by the deadline shown above.

Purpose

To practice with bash shell script functions (writing them and calling them).

Important notes:

* **Each** bash shell script that you write is expected to include a descriptive opening comment block including your name and the last modified date.

    – each bash shell script FUNCTION you write should also have an descriptive opening comment block including at least descriptions of what it expects and what it produces and/or its side-effects when run

* **AHA** -- remember I said that I would get back to you with what `return` means in a function?

    – ...you use `return` to get a *function* to produce an exit status!

    – Why not use `exit` for that? Because it would cause the entire shell script calling the function to be exited, which may be TOO strong an action sometimes!

    – So, you use `return` for this, and the calling function CAN indeed use `$?` to see the function call's exit status.

* HINT: beware! if you REUSE variable names in different functions, you may REGRET it here...! If one function changes a variable name used by another, the other might "see" that change after a call to the other!

* It is possible that your answers may be collected and posted to the course Moodle site.

The Problems:

Problem 1

Create a shell script `hw10-functions.sh`.

Remember the little shell script `get-cs-num.sh` from Homework 9 - Probem 3 part b? Rewrite it as a little function `get_cs_num` that expects one argument, a string, and if that string contains an HSU

CS course prefix (as described in Homework 9, Problem 3), it prints just that course's number to the screen.

After you define the function:

- echo that you are about to call `get_cs_num`, and then

- include a call to that function, calling it with a string that indeed contains a proper HSU CS course prefix.

Problem 2

Now consider `get-length.sh`, from Homework 7 - Problem 2.

Modify this into a function `get_length` within `hw10-functions.sh`, noting the following:

- don't have it complain to the screen if it gets too many arguments -- just have it return its error status in that case

- INSTEAD of having it echo a *message* to the screen with the length of its argument, have it JUST echo the length of the single input

After its definition:

- echo that you are about to call `get_length` with no arguments, and then do so (it will ask for an argument, then, and output the length of what is entered to the screen)

- echo that you are about to call `get_length` with one argument, and then do so -- BUT for this call, set a VARIABLE to the backquoted result of calling this function with one argument, and then echo the value of that variable afterwards to the screen in a message of your choice.

- echo that you are about to call `get_length` with more than one argument, do so, and then echo the exit status of that call to the screen in a message of your choice.

Problem 3

Now consider `is-quant.sh` from Homework 7 - Problem 3.

Modify this into a function `is_quant` within `hw10-functions.sh`, noting the following:

- remember that it needs to `return` the desired exit status, instead of using `exit` to do so!

After its definition:

- echo that you are about to call `is_quant` with no arguments, and then do so -- then echo that call's exit status to the screen in a message of your choice.

- echo that you are about to call `is_quant` with an unsigned integer, and then do so -- then echo that call's exit status to the screen in a message of your choice.

- echo that you are about to call `is_quant` an argument that ISN'T an unsigned integer, do so, then echo that call's exit status to the screen in a message of your choice.

Problem 4

Now consider `make-line.sh` from Homework 7 - Problem 4.

Modify this into a function `make_line` within `hw10-functions.sh`, noting the following:

- remember that it needs to `return` any desired exit status, instead of using `exit` to do so!

- this should call your function `is_quant` from Problem 3 (not the shell script `is-quant.sh`)

- don't have it complain to the screen if it gets bad arguments -- just have it return its error status in those cases

After its definition:

- echo that you are about to call `make_line` with the wrong number arguments, and then do so, and then echo the exit status of that call to the screen in a message of your choice.

- echo that you are about to call `make_line` with two arguments, but with a NON-number as the second argument, and then do so, and then echo the exit status of that call to the screen in a message of your choice.

- echo that you are about to call `make_line` with two GOOD arguments, and then do so -- BUT for this call, set a VARIABLE to the backquoted result of calling this function with those good arguments, and then echo the value of that variable afterwards to the screen in a message of your choice.

Problem 5

Now consider `mantra2.sh` from Homework 7 - Problem 5.

Modify this into a function `mantra2` within `hw10-functions.sh`, noting the following:

- remember that it needs to `return` any desired exit status, instead of using `exit` to do so!

- this should call your functions `is_quant` from Problem 3 and `make_line` from Problem 4 (not the shell scripts it previously called)

- I *think* it is OK for this one to still complain to the screen if it gets bad arguments, given its nature.

- HINT: beware! if you REUSE local variable names, you may REGRET it here...! (That aspect "got" me here, in my first attempt at this function.)

After its definition:

- echo that you are about to call `mantra2` with the wrong number of arguments, and then do so, and then echo the exit status of that call to the screen in a message of your choice.

- echo that you are about to call `mantra2` with a mantra of inappropriate length, and then do so, and then echo the exit status of that call to the screen in a message of your choice.

- echo that you are about to call `mantra2` with a "bad" number of repeats, and then do so, and then echo the exit status of that call to the screen in a message of your choice.

- echo that you are about to call `mantra2` with appropriate arguments, and then do so.

– Submit your resulting `hw10-functions.sh`.

Problem 6

Now copy your functions from `hw10-functions.sh` into a file `hw10-lib.sh`, being careful to include JUST the functions, and none of the function calls. (Do include an opening comment block for the script, and the opening comment blocks for each function.)

Now `source` this new shell script in ANOTHER shell script, `use-hw10-lib.sh`, and then show that you can call all of the functions there:

• explicitly call each function "successfully" at least once,

• preceding each by an echo of a message saying you are about to do so,

• and as/if necessary echo a message demonstrating that it did succeed.

Submit your resulting `hw10-lib.sh` and `use-hw10-lib.sh`.