CMSC 331

Homework 4

Due Friday 3/16, 11:59PM, through GitHub

This homework will give you practice with the Lua programming language. All work is to be done on your own and should run on the GL servers. We will be using GitHub classroom to handle submission this semester. To get started, accept this homework at https://classroom.github.com/a/vZagVoVA. This will give you all the datafiles and directory structure you need.

Important information about git on GL: Git on GL is outdated, so whenever you see an instruction like "git clone https://github.com/REPO_NAME", you need to modifiy it by adding your GitHub username, so it reads "git clone https://GITHUB_USERNAME@github.com /REPO_NAME".

In addition, this will by default try to pop up a window to ask for your GitHub password. If you are on terminal, this will obviously not work, so you need to enter one of the following commands so you are prompted for your password on the terminal

```
unset SSH_ASKPASS
unsetenv SSH_ASKPASS
```

Important: please edit the README.md file to include your name and section number in it.

Simple Functions

1. Average of an Array of Numbers (10 Points)

Write a function `avg(array)` that takes in one argument, an array of numbers, and returns the average of them. You may not use any library functions to complete this problem. You

CMSC 331

2. Integer Division by Repeated Subtraction (10 Points)

Because Lua only has one data type for numbers, it does not provide any ability to do integer division. For this problem, write a function, `int_divide(a,b)`, that takes in two numbers, and returns the result of dividing `a` by `b`. You should use some type of loop to achieve this, repeatedly subtracting `b` from `a` until you cannot do so any more. You may assume both the numerator and denomenator will always be positive. You may not use any library functions to complete this problem.

Advanced Functions

3. Volume and Surface Area of a Cone (10 points)

For this problem, you need to write a function called `cone_facts(r,h)` that takes in the radius of the base and the height of the cone, and returns two values, the volume of the cone, and the surface area of the cone. They must be returned in that order

For reference, the volume of a cone is $\frac{1}{3}\pi r^2 h$ and the surface area can be found using the formula $\pi r^2 + \pi r\sqrt{r^2 + h^2}$

4. String Join (10 points)

Write a function `join(a,b,c,d)` that takes *up to* 4 strings, and joins them together with a comma. Your function should still work when given less than 4 strings, including printing out just one string with no commas attached when given only one string as an argument.

Reading from a file and sorting

5. Read in from a file (10 points)

Write a function `read_elements(f_name)` that reads in the file located at the location specified by `f_name`. This file consists of lines with two space-separated columns, the first column being the name of a chemical element, and the second column being the year it was isolated. Your function should return a table of tables, represeting this information from the file.

Hint:

```
for word in string.gmatch("Hello Lua user", "%a+") do print(word) end
Hello
Lua
user
```

6.Sorting the Table By Year (10 points)

Take a table of the format generated in the previous example, and write a function `sort_by_year(table_name)` that sorts the elements by the year they were discovered, and prints the information to the screen. An example run is shown below.

```
sort_by_year({{'Hydrogen',1776},{'Helium',1868},{'Lithium',1817},{'Phosphorus',
Phosphorus      1669
Hydrogen        1776
Lithium         1817
Helium          1868
```

Lexical Scanner

7. Number Format Scanner (20 points)

Write a function `scan(num_string)` that reads in a single character at a time, and when it reaches end of the string, reports whether it has read in a decimal, octal, or hexadecimal number, and what that number is (as a string), or "NONSENSE" if it isn't a well-formed number.

For the purposes of this assignment, use the following definitions of decimal, octal, and hexadecimal numbers

- decimal - A single zero, or any continous sequence of decimal digits that starts with a non-zero decimal digit and includes 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- octal - An octal number always starts with a 0, and is followed by a single 0 or a non-

- hexadecimal - A hexadecimal number always starts with a 0 followed by a lowercase 'x' and either a single 0 or a sequence of hexadecimal digits. Hexadecimal digts include 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, A, B, C, D, E, and F.

All forms can have an optional positive or negative sign directly infront of the number, with no space between the symbol and the number.

Closures

8. Bank Account (20 points)

Write a function `make_account(currency_string)` that takes in one argument, that is a string representing a currency, and returns three functions. The first function takes one argument and allows the user to deposit money into their bank account. The second function also takes one argument, and allows users to withdraw money from their bank account. The third function takes no arguments, and allows users to see their balance, displayed according to the currect currency. You must use a closure so the balance is remembered between transactions.

The bank account should start out with 0 dollars, and is allowed to go into the negative. An example use is shown below.

```
> deposit, withdraw, report= make_account('dollars')
> report()
You have 0 dollars in your account
> deposit(10)
> report()
You have 10 dollars in your account
> withdraw(4)
> report()
You have 6 dollars in your account
```

Running your Code

If you place your test code inside the same filse as your function definitions, you can run

How You Will be Graded

The first 7 problems will be graded by supplying the functions a range of inputs, to test the correctness and robustness of your code. Depending on the number of test cases, you will lose between one and 0.5 points per test case that does not produce the expected output. For problem 8, you will recieve 5 points for each of the 3 returned functions working correctly, as well as 5 points for the closure being created correctly.

Your code will be run and graded using automated scripts for the most part. It is very importat that you do not name your file anything other than hw4.lua, and that all function names stay as they were given. If you have some incomplete code in your file, it is reccomended that you comment it out, so that it does not prevent successful testing of the working functions in your file.

Submitting Your Homwork

Your homework should be done on the GL system, as that is where it will be tested. All your functions should be defined in a single file, named hw4.lua. Commiting your code to your local repository and pushing it up to GitHub is your submission for this project. Your last submission before the due date will be taken as your submission.

Please test out your ability to commit and push your code to GitHub before the due date. If you have any problems submitting, email me or stop by office hours.