COP4342 - Fall 2018
Assignment 2
Setup and Initialize a Simple Text Database

Objectives: Learn how to assign values to shell variables, read values into shell variables, invoke shell scripts from a shell script and check the exit status, use the touch, grep, and wc Unix utilities, use command substitution, and append output to a file.

Instructions: Your assignment is to write three shell scripts that are used to setup and initialize a simple text database. Create additional shell scripts as needed to abstract out common operations. While you are allowed to invoke other shell scripts that you have written and the Unix utilities specified above, you are not allowed to invoke other executables (i.e. you cannot write your solution in a conventional programming language like C++ or Java). If you need to store information into a temporary file, then you can use the file *tmp.txt*. But be sure to delete it before you exit your script.

ns.sh The first shell script should be named *ns.sh.* Its purpose is to create a new schema. It accepts as arguments a *schema_name* and a list of up to eight *field_name*s. For instance, below is an example invocation of the *ns.sh* script and response.

```
% ns.sh person name age weight
Schema "person" has been created.
```

The information regarding schemas should be stored in a file called *schemas.txt*. Each line of that file contains the *schema_name* followed by the *field_name*s. Below is an example *schemas.txt* file:

```
person name age weight
grades name exam1 exam2 exam3
company employee SSN age
```

Your script should check if the *schema_name* already exists and if so it should print an appropriate error message and exit. If it doesn't already exist, then the script should append the information to the end of the file and print a message that the schema has been created.

nd.sh The second shell script should be named *nd.sh* and its purpose is to create a new database. It accepts two arguments, where the first is a *schema_name* and the second is a *database_name*. For instance, below is an example invocation of the *ns.sh* script and response.

```
% nd.sh person medical
Database "medical" using schema "person" has been created.
```

The name of the database and the schema it uses should be appended to the *databases.txt* file. Below is an example *databases.txt* file:

```
medical person
cop3330 grades
intel company
cop4342 grades
```

If either the *schema_name* does not exist or the *database_name* already exists, then you should print an appropriate error message and exit. If the *schema_name* exists and the *database_name* is not already used, then the script should append the information to the end of the *databases.txt* file, create a new empty database file of the form <*database_name*>*.db*, and print a message that the database has been created.

ir.sh The third shell script should be named *ir.sh* and its purpose is to insert a record into a database. It accepts one argument, which is the *database_name*. The *ir.sh* script should then prompt the user for the values of the fields in that database. After that the script should read in the field values and append these values in a new line to the end of the <*database_name*>*.db* file. Finally, the script should print a message that the new database record has been added. For instance, below is an example session of using the *ir.sh* script. Note the field values would be entered by the user.

```
% ir.sh medical
Enter values for name age weight:
Johnson 42 220
Record added to "medical" database.
```

If the *database_name* does not exist in the *databases.txt* file or the <*database_name.db*> does not exist, then you should print an appropriate error message and exit.

Submission: You should create a tar file formatted as ‘Lastname_Firstname_Assignment2.tar’ containing these three shell scripts and submit the file through Canvas’s ‘Assignments’ section **before** the beginning of class on 09/20/18. If you find that you need additional shell scripts that can perform a function that is needed by more than one of the three scripts described previously, put the additional shell scripts in the tar file as well. Sample command to put files in a tar file: tar -cvf yourfile.tar inputfile1 inputfile2. Sample command to extract the files from the tar file: tar -xvf yourfile.tar.