SY201 – Cyber Operations Fundamentals; Programming Assignment `08`

Host Analyzer

Name: _____ Alpha: _____

Begin this assignment by making sure your repo is up to date. Change to **`~/repo201`** and type:

`git pull`

All the files required for this assignment, including an electronic version of this handout, are available in:

`~/repo201/programming/pa08`

Read this entire handout at least once before you begin your research or start coding

1. **Background**:

Many programs that run on a modern computer will perform some kind of logging to keep track of various system events. For example, as users log on / off a system, those events are captured in a log than can be analyzed later.

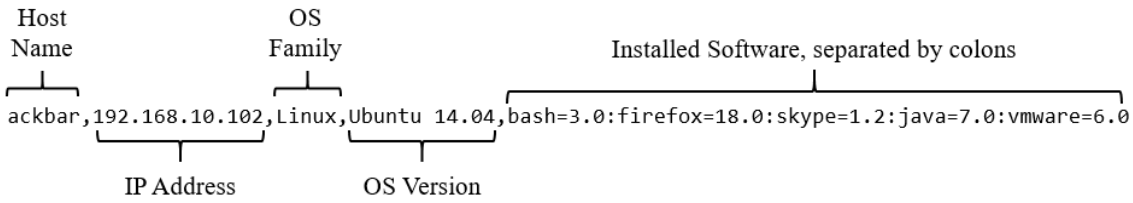
2. **Your Program Should Do This:**

For this assignment you're going to write a Python program using *Classes*. Your program will take a file name from the command line that contains a series of log entries about various machines on a network. Your program will then load the file into a data structure and offer the user menu options to query the file.

Your instructor will provide you with several files that will allow you to test your code. Each line in a file is terminated with newline (`'\n'`) and is a collection of Comma Separated Values (CSV). Lines conform to the following format (*for ease of reading, this is broken into two columns*):

```
1. Host Name (string)
2. 1 comma
3. IP address (string)
4. 1 comma
5. Host OS Family (string)
6. 1 comma
7. Host OS Version (string)
8. 1 comma
9. The remainder of the line
   contains all the installed
   software as one or more
   strings separated by colons.
```

Here's a sample of what a single line from a host file looks like:



3. **Design Requirements:**

   a. Think carefully about how you want to approach this assignment before you start coding. It's rarely effective when you open up your editor and just start typing. Think first about the kinds of variables you will use, their names, and how the different sections of your code will interact. Take out a piece of paper and pencil and sketch out the major components of your design (collect input, validate input, perform calculations, present results, etc.). Draw simple flow diagrams or write code-like snippets (called *pseudocode*) that capture the major steps in your program. Your design sketch and / or pseudocode is a required deliverable for this assignment.

   b. The required approach for this assignment is to begin by creating a Python `class` called `Host`, which has the following methods and ***private*** properties:

Host Class

Properties:	Methods:
__host	__init__(self, line):
__ip	getHost(self):
__osFam	getIp(self):
__osVer	getOsFam(self):
__swList	getOsVer(self):
	getSwList(self):

The initializer of your `Host` class should take a single line from a file (as a string) and break it into its component parts, storing each in the correct property. All properties of the `Host` class are strings, except `__swList`, which is a list of strings. All the remaining methods in the `Host` class are simply *getters*.

c. Next, you must use a Python dictionary in your main program where the *keys* are host names, and the *values* are objects of the type `Host` Class.

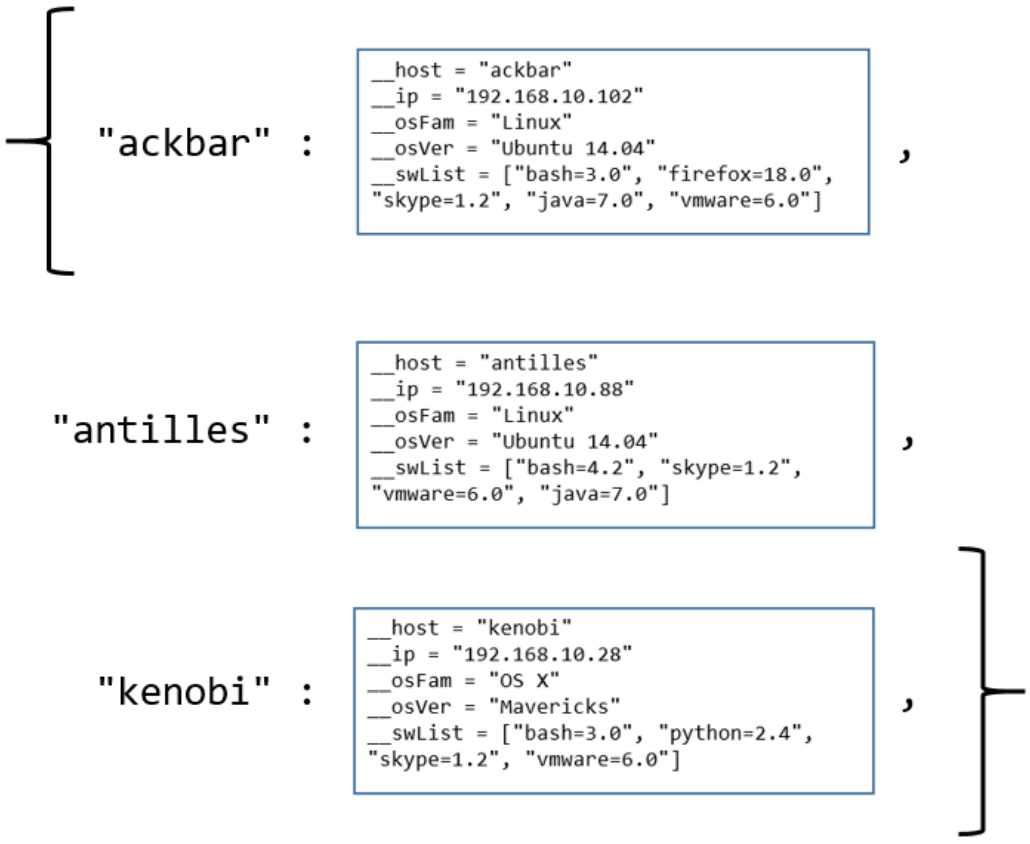
For example, if you had a log file with these three lines:

1. `ackbar,192.168.10.102,Linux,Ubuntu 14.04,bash=3.0:firefox=18.0:skype=1.2:java=7.0:vmware=6.0`

2. `antilles,192.168.10.88,Linux,Ubuntu 14.04,bash=4.2:skype=1.2:vmware=6.0:java=7.0`

3. `kenobi,192.168.10.28,OS X,Mavericks,bash=3.0:python=2.4:skype=1.2:vmware=6.0`

You would store that in a dictionary as follows:



d. Your class definition (for the `Host` Class) must reside in a separate file and be imported into your code. Review the lesson in `~/repo201/classwork/cw30` for instructions on how to create your own custom libraries. This one's a little tricky, so we'll discuss it in class.

e. You must design your program using a `main()` function.

f. You must accept the name of the log file to be processed from the command line.

g. Your program must provide three options:

```
1. Alphabetical listing of all hosts
2. Stats for a specific host
3. Quit
```

h. For the alphabetical listing of hosts, you must display:

   i. Host name
   ii. IP Address
   iii. OS Family
   iv. OS Version

i. When displaying stats for a particular host, you must display:

   i. Host Name
   ii. IP Address
   iii. OS Family
   iv. OS Version
   v. Installed SW

j. For both options, your output must be neat and well organized. See Section 5 (**Program Flow**) for examples of how to format your output.

k. Your program will run in a loop (repeatedly displaying and executing options) until `Quit` (Option `3`) is selected.

l. You must make use of functions in this assignment. At a minimum you must design and use the following function (*though you're welcome to use as many functions as you wish*):

```
def getInt(prompt, lower, upper):
```

This function prompts the user for an integer in the range: [`lower,upper`], using `prompt` and returns a valid integer to the calling program. The function must notify the user if an invalid integer is entered (e.g. a string) or if the integer entered is not within [`lower,upper`]. For these cases, keep looping until the user provides compliant input.

m. Your `getInt()` function must reside in a separate file (library) and be imported into your code. Review the lesson in `~/repo201/classwork/cw30` for instructions on how to create your own custom libraries.

4. **Error Handling**

a. Host names are case sensitive. You must warn the user if he / she enters a host name that doesn't exist.

b. Be sure to handle the case where a user tries to process (open) a log file that doesn't exist.

c. There may be other errors (edge cases) not described here. You should thoroughly test your code using all the sample files.

5. **Program Flow**

Here's a sample run of the program:

```
*** Host File Analyzer ***

1. Alphabetical listing of all hosts
2. Stats for a specific host
3. Quit

Selection: 1

ackbar     192.168.10.92    OS X       Mavericks
beru       192.168.10.90    Linux      Ubuntu 16.04
chewbacca  192.168.10.87    Linux      Ubuntu 16.04
jinn       192.168.10.113   Windows    XP
jubnuk     192.168.10.76    OS X       Mavericks
kylo       192.168.10.42    Linux      Ubuntu 14.04
leia       192.168.10.70    OS X       Mavericks
luminara   192.168.10.48    Linux      Ubuntu 14.04
mace       192.168.10.94    Windows    7
mail       192.168.10.14    OS X       Lion

*** Host File Analyzer ***

1. Alphabetical listing of all hosts
2. Stats for a specific host
3. Quit

Selection: 2

Host name: leia

   Host Name: leia
  IP Address: 192.168.10.70
   OS Family: OS X
  OS Version: Mavericks
Installed SW: java=7.0, python=2.4, chrome=26.0, vmware=6.0

*** Host File Analyzer ***

1. Alphabetical listing of all hosts
2. Stats for a specific host
3. Quit

Selection: 3
```

6. **Testing:**

Thoroughly testing your program can be a difficult task. For complex programs there may be hundreds or thousands of possible cases to test. Try to think about what kind of input will cause bad behavior in your program and run those tests. *I recommend you try all eight of the log files included in this assignment*.

7. **Assumptions You May Make:**

   a. Log files will be properly formatted, with no hidden anomalies.

   b. All hosts in log files will be unique (no duplicates).

   c. You can use generic notification messages for all invalid integers your user may try to input (e.g menu selections). For example, if someone selects option `4` for your program you could respond with: `Integer out of bounds.`

8. **Hints:**

   a. Give your variables meaningful, descriptive names.

   b. You need to print the contents of a dictionary, sorted by the key (host name). The challenge is that, unlike lists, dictionaries are an unordered data structure.

      There are many ways to handle this in Python, but the easiest is just to use the `keys()` method on dictionaries to extract all the dictionary keys, then sort them using the `sorted()` function. An example of printing all the values in a given dictionary, while processing the keys in sorted order, looks like this:

```
for key in sorted(D.keys()):
   print(D[key])
```

   c. The directory for this project includes an executable version (called `hostAnalyzer`) that you can run to get a sense for how your program should operate. It also includes several log files, stored inside a compressed (`.zip`) archive. You can double-click on the archive in Ubuntu Linux to get the log files.

      Let's assume you're working in your `~/shares/sy201` directory. You can run `hostAnalyzer` by first copying it to your working directory and then adjusting

its permissions with the following commands:

i. `cp ~/repo201/programming/pa08/hostAnalyzer .`
(*the trailing space and period in the command above are important*)
ii. `chmod 755 hostAnalyzer`

You only have to perform steps (i) and (ii) once. After that, you can run the program anytime you want by typing: `./hostAnalyzer <`*log file name*`>`.

9. **Deliverables And Due Dates:**

a. Using paper, and pen or pencil, complete your pseudocode / flow diagram *before* you start coding. You may use more than one page for this if you need it. Be sure to indicate your name and alpha on this page.

b. Your completed source code (the one ending in `.py`).

c. Any custom library functions you write, all contained in a single directory e.g. `utils`.

d. Submit parts (a) through (c) by 2359 on Monday, 26 November, in accordance with your instructor's directions.