CS 455/555 Programming Assignment #1

Due Date: **Sunday Sep 11th, 2016 - 11:59pm**

Motivation:

The programming assignments during the semester will involve reading user input from the command line, parsing strings returned in network messages, and parsing URLs. This assignment is designed to refresh your memory on (or help you learn about) these particular parts.

The reason we are using Java/Python is that the socket programming constructs used to develop network applications are much simpler than in C or C++. Although I encourage to learn/use Java/Python, you still could use C++. However, it is your total responsibility to figure out the right API and to debug any issue with your code while developing networking components.

Note that the 6th edition of the textbook has Socket programming examples in Python, while earlier editions use Java. It is expected that you should be able to complete this assignment within few hours. Please come talk to me if you couldn't.

Rules:

- Your code should run on the CS department Linux machines. If you do not have an account, create one today!
- As with all projects, you are not permitted to work with anyone else (even students not in the class) - all of the coding and documentation must be your own.
- Your program must compile (if Java/C++) and run on the CS Unix machines.
- You must write neat code and document it well. You will lose points for sloppy programs that contain little or no comments.

Tasks:

1. Read in an input file called 'input.txt'. The input file will contain ***n*** lines. Each line is in the format of 5 non-negative integers that the first four are separated by a dot and the last one is separated with a colon, no space is included. Example of such a line is: '127.0.0.1:8080'
2. Create an output file called 'output.txt'.
3. The first two lines in your output file should be your name and your UIN respectively.
4. For each line you read from the input file, you need to do the following:
    a. Copy the current line exactly to the output file
    b. Use the colon as a separator to split the line into two parts: ***addr***, and ***prt***
    c. If the input line has any error such as:
        i. Contains less or more than 5 integers
        ii. Contains inputs are not integers
        iii. Contains negative integers

iv. Dots and Colon are not in right place
You generate 'Error in Input Line'

d. Otherwise, generate another line with following format:
'Address:***addr***, Port:***prt'***

5- At the end of the File print a summary containing the number or lines read and number of errors found as shown in the example next.

Testing:

Input.txt	Output.txt
127.0.0.1:8080 127.a.0.10:8081 127.0.0.1:8080.1 127.0.10.5:-8080	Name: XXXX UIN: XXXXXX 127.0.0.1:8080 Address: 127.0.0.1, Port: 8080 127.a.0.10:8081 Error in Input Line 127.0.0.1:8080.1 Error in Input Line 127.0.10.5:-8080 Error in Input Line Summary: Lines Read: 4. Number of Errors: 3.

Submission Materials:

- Make sure your program compiles and executes on Dept's Linux machines.
- Create a "Readme.txt" file for your program that lists how to compile and execute the program. Include your name and your UIN as the first line in the Readme.txt.
- You must name your source file Prog1.java, Prog1.py, or Proj1.cpp (capitalize only the first letter).
- For Java and C++, make sure that you submit all files necessary to compile your program. But, do not submit compiled files
- Zip all files of your program and name them prog_assign_1.
- Submit through Blackboard.

Submitting Assignments using Blackboard

Instructions from the official Blackboard 9.1 help pages (with screenshots)

- https://en-us.help.blackboard.com/Learn/9.1_Older_Versions/9.1_SP_10_and_SP_11/Student/060_Tests_and_Assignments/Submitting_Assignments

My step-by-step instructions:

- Log in to the course Blackboard page
- Click the **Programming Assignments** link in the left sidebar
- Select the current assignment
- Under **Assignment Submission**, there is an **Attach File** section.
  Click **Browse My Computer**. Your web browser will open a window in which you can select the file to attach.
- Once you have successfully attached a file, its name should appear in the **Attached files** list.
- You may attach as many files as needed.
- When you've finished adding all of the files that are needed for the assignment, click **Submit** at the bottom of the page.
  Important: If you click Save as Draft instead of Submit, your assignment will not be turned in.
- After you've clicked **Submit**, an assignment receipt page will display.
  Click **OK**.

Important: *If you submit your assignment and later realize you have made a mistake, Blackboard will allow you to re-submit the assignment. The time of your last attempt will be counted as your submission date. Re-submissions submitted after the assignment deadline are not guaranteed to be graded. If I have started grading your assignment, I will not grade another version submitted after the due date.*

Getting Started

Java

If you've never programmed with Java before, here are some helpful tips:

- Java syntax is very, very similar to C++.
- Only one class is allowed per file, and the name of the class must match (exactly!) the name of the file. For Program 1, you are to name your source file Prog1.java, so you must name your class Prog1.
- The "**import**" statement is used give access to built-in libraries, much like the "**#include**" statement in C/C++.
- To print output (instead of **printf**), use **System.out.print()** or **System.out.println()**. **println()** adds a newline character to the end of the String given.
- The main method should be declared as **public static void main (String argv[])**. The String array, argv, gives you access to the command-line arguments. The first element of the array (argv[0]) is the first argument (and not the program name as in C/C++).
- To compile your program, use javac. *Ex: javac Prog1.java*
- To run your program, use java. *Ex: java Prog1*
- The Java String class reference will be useful in completing this assignment. You will also need to handle text files in Java.
- See the other Java references on the **Links** page on the course webpage.

Here is the skeleton for the first program to be saved as Proj1.java

```
// filename : Prog1.java
// Insert Your Name
// Insert Your Class (CS 455 or CS 555), Spring 2014
// Program 1 - Java/Python Refresher

import java.io.*;

class Prog1 {

  public static void main (String argv[])
  {

     // insert your code here

  }
}
```

Python

If you've never programmed with Python before, here are some helpful tips:

- Python syntax is very different from Java, C, C++
  - proper indention is essential. It's used to group statements (such as branches in an if..else statement)
  - newline delimits the end of a statement, not a semicolon (;)
- The "**import**" statement is used give access to built-in libraries, much like the "**#include**" statement in C/C++.
- The list **sys.argv** gives you access to the command-line arguments. The first element of the array (argv[0]) is the program name (Prog1.py).
- Python is interpreted, so there's no need for compilation. To run your program, use python. *Ex: python Prog1.py*
- The Python String methods reference and Python File Objects will be useful in completing this assignment.
- See the other Python links on the **Links** page on the course webpage.

Here is the skeleton for the first program to be saved as Prog1.py:

```
# Insert Your Name
# Insert Your Class (CS 455 or CS 555), Spring 2014
# Program 1 - Java/Python Refresher

import sys

# insert your code here
```

Unix

If you've never used Unix before, here are some helpful tips:

- See Dr. Zeil's page on Software needed. You'll want to get Xming and PuTTY for your own machine. Computers in the CS lab have XWin32 (similar to Xming) and either PuTTY or the ssh Secure Shell Client.
    - Xming - allows Unix windows (such as terminal windows or Emacs editing windows) to be displayed on your PC
    - PuTTY (or ssh Secure Shell Client) - allows you to securely login to a Unix machine
- Dr. Zeil has put notes from his entire CS 252 - Intro to Unix course online. If you have never used Unix before, please look at these notes. You may want to pay special attention to Section 4 on using X Windows and editing files with emacs. See also the Unix links on the course's links page.
- Start Xming (or XWin32)
- Securely login to one of the CS Unix machines. See the detailed instructions.