Computer Networking and Network Programming (ECE 5650)
Project 1
Fall 2020

Instructions:

- **Original Work**: this assignment must represent the original work of the team members.
  - You must not look at other solutions or show your solutions to anyone else.
  - You must not collaborate or discuss the assignment with other groups or individuals.
  - You must not get any portion of the code from Internet sources, other current or prior students, or other groups or individuals.
  - You must not ask or pay others to help you with the solution.

  Policy on Cheating, Fabrication, and Plagiarism: Cheating, fabrication, plagiarism, and helping others to commit these acts are all forms of academic dishonesty, and they are wrong. Therefore, avoid all appearance of improper behavior! Students who witness cheating should report the incident to the instructor as soon as possible. Academic misconduct will result in at least failing the course.

- **Late submissions:** No late submissions will be accepted.

- **Teamwork**: You must work with one other student and **must submit only one copy with both your names.**

- **Python version: your program(s) must be in Python 3.8.x or later**.

- **Socket programming: your program must use low-level socket programming for all communication/networking activities**. For these purposes, you must use only the socket library. For other purposes, you are free to use other libraries, such as sys, time, signal, os, etc.

- **Documentation:** Each required program must be well documented and well commented and must include the **authors and dates of creation and modification**.

- **Submission: please follow closely all submission instructions.**

- **Have questions?** Please contact the GTA/Grader. You can ask him by e-mail or send an e-mail to schedule a meeting on Microsoft Teams. No help will be given on the day the project is due.

Assessed Penalties:

Situation	Penalty
Late submission	Not accepted
Plagiarism or disallowed collaboration	At minimum negative grade
Not using the provided server skeleton code	Not accepted
Not using low socket programming for all networking aspects	Not accepted
Not using Python 3.8.x	Not accepted
Failure to include the full report using the template and including the test procedure, thoughtful screenshots, and other requirements	Up to 40%
Failure to have a well commented and documented program(s), including the authors and dates of creation and modification	Up to 10%

Assignment

In this project, you will develop a Cloud File Processor in Python using **low-level socket programming**, as discussed in the class. Recall that in all class projects, you must use basic socket programming for all communication aspects. You must write the programs in **Python 3.8.x.**

You will develop a client and a corresponding cloud-computing server that handles one client at a time. The client asks the user for the specific file processing operation to be performed and then sends the input text file and other necessary information to the server, which in turn performs the desired operation on the received file and returns the output or the modified file content to the client. The client program must properly display the output or store the file content, depending on the operation. The client program must check for the validity of the user input.

The following operations must be supported:

- **(25 Points)** *SearchWord* – This operation asks the server to determine *the number of*

occurrences of a specified word in a specified text file. The client must prompt the user to enter the name of a text file and the word to search for in the file. The client then transfers the file content and the word to the server, which in turn calculates the number of occurrences of the specified word in the received text file and then returns the result to the client. Finally, the client must display the output in a well-formatted message.

- **(25 Points)** *ReplaceWord* – This operation asks the server to replace a specified word by another specified word in a specified input text file. The client must prompt the user to enter two words, the name of an input text file, and the name of the output file. The client then transfers the input file content and the two words to the server, which in turn produces a modified file content with every occurrence of the first word being replaced by the second word. Finally, the client must store the received modified file content as a file with the specified output file name.

- **(25 Points)** *ReverseWords* – This operation asks the server to reverse the order of words in a specified text file. The client must prompt the user to enter the names of the input and output files. The client then transfers the input file content to the server, which in turn reverses the order of words and returns the modified file content to the client. Finally, the client must store the received modified content as a file with the specified output file name.

- **(10 Points)** *DisplayFile* – This operation asks the client to display the content of the specified local filename. This is a local operation, which does not interact with the server.

- **(15 Points)** *Exit* – When the server receives this command from the client, it must close the connection with the client and wait for a new client connection. The client program then terminates.

The first three operations require asking the user of the client to enter the name of a text file in the client and then the file must be transferred to the server for processing to execute the requested operation. You can assume that the file is in the *same* folder as the client program.

All operations must be **case-insensitive**.

Thoughtful Screenshots: You must provide thoughtful screenshots of the command lines of both

the client and server to prove that your programs work as expected, with an adequate number of possible test cases. You must use the DisplayFile operation to show the modified file contents. Be creative!

Notes and Hints:

- Again, all file operations must be performed by the server, not the client.
- **Recall that the content of a file may not be received by the other host using a single call of the socket recv() method. Thus, the length of the file (in bytes) must be determined and then transferred to the other host. The other host will then call the socket recv() method as many times as needed until the entire content is downloaded.**
- **For the server to know how many messages to receive from the client and the types of these messages, the client must send first the operation name to the server.**
- Some of the operations to be supported will require the use of standard string processing functions.

Submission Requirements

- Please follow carefully all submission requirements.

- **You must submit all the following files (a total of three files)**:
  - The client Python program in .py extension.
  - The server Python program in .py extension.
  - **A professional report in pdf format, using the attached template and including the following sections:**
    a. **team member names,**
    b. **a copy of each source code,**
    c. **testing procedure used to verify the correctness of the program(s), including a description of inputs,**
    d. **screenshots and their explanations**, and
    e. **completion status (self-critique)**, including answers to the following questions **for each required program**.
       - Does your program meet all requirements? If not, explain the problem.
       - Does the program run correctly all the time? If not, explain the problem.
       - Did you adequately test the program? If not, specify.
       - Is the program well documented?
  - **File Naming:** the filenames must contain your last name(s): yourlastname1-yourlastname2-keyword.properExtension. The keyword describes the overall functionality of the program.

- The files must be submitted using **Canvas**:
  - Open this assignment on Canvas.
  - Select “Submit Assignment”
  - Select “File Upload”.
  - Browse for the first file to upload and select it.
  - Choose “Add Another File” to upload the other files one at a time.
  - Hit the “Submit Assignment” button.

Good Luck!