

Purdue Northwest, Hammond, CS 404

Homework 5

Due 5/02/2020 11:59 pm

This is an extension of HW-3. The purpose of this assignment is to write Java programs to demonstrate inter-process communication. We implement *autocomplete* for a given set of N strings and nonnegative weights. That is, given a prefix, find all strings in the set that start with the prefix, in descending order of weight.

For more information on Autocomplete, read the HW-3 write-up.

1 Server.java

Create a file `Server.java` that creates a socket at a port number that one reads from the command line. The server should receive a query prefix (this query is the same as prefix in the Autocomplete exercise of HW-3) and a file name to search the query in from *client*. Client is written in a separate file running on a different console as a separate process.

2 Client.java

Create a file `Client.java`, that creates a socket of its own and operates on the same port as the Server (`Server.java`), the port number again, would have to be read from the command line; along with the query prefix and the data file you wish to search the query in. The port number should be the same that you chose to run the Server on. This process should be able to send the prefix and the file name to the server process, which would eventually use the code from HW-3 to collect the results and transfer them to client process. The results should be declared on the client console.

3 Data

Data is the same as in HW-3.

4 Helpful Hints

- First, start the server process, of course.
- In the `hw-5_code` directory there is a working example of a simpler version of client-server communication, `Hello.Server.java` and `Hello.Client.java`. First run the server— `Hello.Server.java` and then open another console and run the— `Hello.Client.java`
- For more information about functionality in source code files in the directory `hw_5code`, visit:
<https://docs.oracle.com/javase/7/docs/api/java/net/ServerSocket.html>

5 Files to Submit

Write a `Report.txt` that-

- Discusses what you found difficult about this assignment, how you planned your approach to it, and what you learned completing it
- Include as many implementation details as you can.
- Have a short description of how you approached each problem, issues you encountered, and how you resolved those issues.
- *Cite all your references and collaborations (if any).*

Send me only (via email: arpitamhow@gmail.com):

1. `Term.java`
2. `BinarySearchDeluxe.java`
3. `Autocomplete.java`
4. `Server.java` – 50 points
5. `Client.java` – 50 points
6. `Report.txt` – 50 points