CS380 — Exercise 2

January 11, 2017

Due: Wednesday, January 18, 2017 before midnight (50 points)

You are allowed to work with a partner to complete this exercise. If you do work with a partner, only one person should create the codebank project and add the other as a member with at least developer privilege. You should also include a comment at the top of any source code files with both partners' names to ensure grades are entered correctly.

Preparing the Project

1. In exercise 1, you forked an existing project. In this exercise, you will create a project from scratch.
2. Go to `https://codebank.xyz` and create a new project named `CS380-EX2`.
3. On your local machine, from a terminal or git bash navigate to the folder you use for storing CS380 related files and create a new directory to store this exercise. I'd recommend calling it `CS380-EX2` to match the repository. From now on, we'll call this directory the *working directory*.
4. `cd` in to the working directory and run:

   ```
   $ git init
   $ git remote add origin https://codebank.xyz/username/CS380-EX2.git
   ```

   where `username` is your bronconame.
5. Now, the directory on your machine is a git repository with a reference to the remote repository on `https://codebank.xyz`.
6. As an alternative to creating a local repository yourself, you can `git clone` the empty repository from the website to accomplish the same thing.

The Program

Description

1. Create a Java source file[1] named `Ex2Client.java` with a class named `Ex2Client` that contains the `main` method. You can also create any other classes or files as needed.
2. Your program should create a `Socket` connection to `codebank.xyz` port number 38102.
3. In this program, I will send you 100 bytes, however I will only send half of a byte each time. For example, if one of the 100 bytes has value `0x5A`, I would send you two bytes with values `0x05` and `0x0A`.

[1] If you want to use a different language, let me know.

4. For each of the 100 bytes I send, read from the `InputStream` twice and put the two pieces together to form a single byte. Using the previous example, I sent you `0x05` and `0x0A`. You should create from this the byte value `0x5A`.
5. After reconstructing the 100 byte message, use Java's `java.util.zip.CRC32` class to generate a CRC32 error code for the 100 bytes.
6. Send this CRC code as a sequence of four bytes back to the server.
7. If the server constructs the same CRC32 code as you, it will then respond with a single byte having value 1, otherwise it will respond with a byte having value 0.

Sample Output

```
$ java Ex2Client
Connected to server.
Received bytes:
  7DB5E296C14E005CDB77
  CD415A4C9B99ACF863C9
  582B3AC552737A554F7D
  20C76A447116BC1AC30E
  E13932CECBB6CF41DE3E
  A3C7C2EA09C20C04B2E5
  0B43C3FD01CCDA60FFED
  5615DB7C7A76D54CD4DF
  4CD313CA69E11972A9E0
  7B12B877064CB32CB2F4
Generated CRC32: FA128A79.
Response good.
Disconnected from server.

$ java Ex2Client
Received bytes:
  51E0908A32F6867BD5DF
  384AC3A040D75D699CCB
  9B137545AEA15BA078C7
  C0FAF205BA90E5A4C7C9
  5DC2358DD7DC89CBBBA4
  08933D13C39B3945716F
  153C1840E77C184D0A60
  661BBF20FC4C9F0BFB04
  3C0990437604AE6BC601
  02483A127DBC7220BD84
Generated CRC32: 7BD05977.
Response good.
Disconnected from server.

$ java Ex2Client
Received bytes:
  B0D1E7091595BA488AEA
  52E58A10DEB6F2F36A93
  881F370163816FBF1153
```

```
  3A0B1484224BDA08DB19
  CC540760D748F737A8C6
  6B9474C4183BAFCB8855
  B2E4D1972C95DFC18FBC
  113E217D1B7A5F77064E
  5A3B25EAF906A81ED83F
  428AEC7B23DABD558560
Generated CRC32: 12AC8770.
Response good.
Disconnected from server.

$ java Ex2Client
Connected to server.
Received bytes:
  915559DBF723B26316E2
  BAA13284A0EF10B768C4
  D4D204423D4516EC37C1
  E77C01E9A4662F89920F
  729D2757EE8DECAD90E5
  71371F7C470AF4AFBD30
  C22BD5227C8666C2CC2A
  B063B9CA38A7A3B5020D
  2FE1D97BE98A5322A350
  9E422E8A4AA40ED0E150
Generated CRC32: 28DF6CDD.
Response good.
Disconnected from server.
```

Submission

You can use `git add`, `git commit`, and `git push` to push the changes to https://codebank.xyz. You can make as many commits and push as many times as desired before the deadline.