CS 371 –Project

Code Submission Deadline: Friday, November 17, 2023

Demonstration meetings with the TA are from Nov 20-Nov 30

Create a Pong Game Server & Client

Maximum points: 115/100

Project Description:

For this project, you will have the opportunity to design and implement the classic Atari game Pong with a twist; it will be a multiplayer game with client-server architecture. You will create both the client and server logic using what you have learned in the socket programming lessons, allowing players to compete against each other in a game of Pong over a network connection.

You are not expected to know how to create the actual game logic, so that has been provided for you.

Primary Task:

Client. The client needs to communicate with the server to relay and receive information about the current game state. The client is responsible for sending the location of the user's Pong paddle to the server. It will receive from the server the location of the other player's paddle, the location of the ball and the current score. A framework for the client code has been provided inside of pongClient.py. You can build off of this code, or discard it entirely.

Server. The server needs to communicate with two clients simultaneously over a network using sockets. It will need to use threads to handle the two simultaneous clients. It is responsible for relaying the location of the other player's paddle to the client, the location of the ball and the current score. A framework for the server code has been provided inside of pongServer.py. As with the client, you can build off of this code, or discard it entirely.

Groups:

The project should be completed with a group no larger than three people. You can work alone but it is not recommended. You will rarely be tasked with a solo project in your career as a software developer and learning to work with other people now will help you immeasurably when you graduate. If you are having problems finding a partner, please email the TA as soon as possible and they will match you with someone.

Project Deliverables & Submission:

pongClient.py	The client side of the pong game
pongServer.py	The server that hosts the game
readme.<md|txt>	Text or markdown file explaining how to run your code.
<LastName>_<FirstName>_CS371_Fall2023.pdf Ex: Smith_John_CS371_Fall2023.pdf	Your project report. Each group member must submit an individual report.

Please place your group's report and readme in the top level of your project and zip your entire project together. If your project requires extra libraries, please include a pip requirements file called "requirements.txt" in the top level also and note that in the readme. If you're unsure how to create a requirements file, follow this guide: https://pip.pypa.io/en/stable/reference/requirements-file-format/

TA Meeting:

You will need to present your project to the TA. The presentation is only a demonstration of your code working, or an explanation as to why it isn't, and how far you were able to get (you do not need to prepare a PowerPoint). Please bring at least two computers so that one can act as the host, the other a client, and I will connect as the second client. You may need to use UK's VPN. If you've not set it up yet, instructions are here: https://www.cs.uky.edu/docs/users/vpn.html. You will be asked questions about how and why you chose to implement specific features of your project. Please schedule this meeting as far out as possible and show up on time to the meeting. Punctuality is very important for these meetings, please be early.

Points Distribution:

The points awarded will be gradients between the point descriptions below. For Example, if your report has minor spelling mistakes but is otherwise well written and thought out, you may only lose one or two points, you won't automatically jump down to 20pts.

(10) TA Meeting:

- 10: You were on time & you understood every aspect of your project
- 5: You were on time & understood your portion of the project only
- 0: You didn't understand any portion of the project

(40) Report:

- 40: Your report has every required section, is well thought out and is written in clear & precise English
- 20: Your report is lacking key sections or is difficult to understand

(50) Code:

- 50: Your code is complete and works perfectly with no bugs, is well written & well commented
- 40: Your code has minor bugs but is complete, well written & fairly well commented
- 30: Your code has major bugs but mostly works, is well written & commented
- 20: Your code is missing key functionality (client doesn't connect to server, etc.) **or** badly written **or** not commented
- 10: Your code does not work at all but it is well commented as to why

(15) Bonus:

- 2: You and your partner use git as a version control & you have a relatively long commit history
- 2: You do the above and: implement the client & server in a way to allow any number of client connections at once (i.e. more than just the two that are required)
- 2: You do the above and: you implement a "Play Again" feature where the players can play another game after one is won.
- 4: You do the above and: you implement a leaderboard as a webpage served at port 80 that displays the initials of the players and the number of times they have won.
- 5: You do the above and: you implement player registration, authentication & encryption for all communication with the server without negatively affection the playability of the game.

Coding Style:

This section will determine how "well written & commented" the coding aspect of your project is.

Type Hinting. Type hinting is required for all methods you write. If you don't know what type hinting is in Python, please read this: https://dev.to/decorator_factory/type-hints-in-python-tutorial-3pel

Project Files. Please use the following template for your project files:

# Contributing Authors:	<Anyone who touched the code>
# Email Addresses:	<Your uky.edu email addresses>
# Date:	<The date the file was last edited>
# Purpose:	<How this file contributes to the project>
# Misc:	<Not Required. Anything else you might want to include>

Methods. Please use the following template for all methods you write:

# Author:	<Who wrote this method>
# Purpose:	<What should this method do>
# Pre:	<What preconditions does this method expect to be true? Ex. This method expects the program to be in X state before being called>
# Post:	<What postconditions are true after this method is called? Ex. This method changed global variable X to Y>

Report

Your group will need to write a report with sections outlined below. This is a technical paper so please do not be pedantic in the hopes of earning more points. Write exactly as much as you need per section and no more. The goal of this paper is to quickly explain to someone unfamiliar with the project (or yourself in a few years) why and how you did it. If English is not your first language and you have difficulty with English grammar or spelling, please ask a peer to review it, or utilize the University of Kentucky's writing center: https://wrd.as.uky.edu/writing-center

Required Sections.

- Background: Briefly describe the problem you are solving.
- Design: Describe how you designed your implementation (design before you start coding)
- Implementation: How was your design implemented (a UML diagram and/or pseudo code may be helpful here)
- Challenges: What didn't go as planned and how did you adapt?
- Lessons Learned: What did you not know before starting this project that you now do?
- (Optional) Known Bugs: Document the bugs you were unable to fix here and how you would solve them given more time. You'll still lose points but not as many.
- Conclusions: Summary of the work and final considerations.