Project 2 (Optional)
(Due on Scheduled Final Day - Presentation) (100 Points)
If you do it, you don't have to take a final exam (This is much easier)

Option 1 (100 points – Full Grade)

In this project, you will design and implement a client server battleship game. There will be two different components of this game and ***you can only collaborate with one other person on it***. You must demo this and explain the code in detail. You can use Python or Java.

Requirements:

1- TCP Server/ TCP Client.
2- Maximum of two clients can play a game.
3- Game rules are managed by the Server.
4- Players play in turns.
5- You must design a protocol for the following functions:
   a. Login by a client to the server – login command. (You can keep logins and passwords in a file or in a database connected to the server)
   b. A logout command.
   c. A client can see a list of all logged in other clients by issuing a "who" command. The "who –l" command should result in a listing of who is online, and what their total score history has been.
   d. A client can invite another client to start a game: "invite" command – and then "accept" or "reject" by the other party. Only a client who is not in a game can be invited. Once one party accepts an invitation, the Server will initiate the game as described below.
   e. A client can issue a hit command with coordinates "hit (2,3)". The server would issue an announcement to both players of the result of each play.
   f. Once someone reaches a winning state, the game should be terminated by the server and the result announced.
   g. Design a scoring system for each win, and keep a running

total for each player.

h. A client could issue a “who –all” command to see the running score of all players in the system (logged in or not).
i. A client can broadcast messages to a group.
j. A client can be a spectator and watch a game (this part is a little more involved – do it last)
k. An admin client can broadcast messages.

6- Once a game starts, the server will create a random size grid (square) that is not smaller than 8x8 and not larger than 24x24 and place a number of ships randomly (random placement and random number of ships in each game -10 max – of course they must be the same for each player), and will only announce the size of the grid, the number of the pieces and location of the pieces to the party who owns them). The game will start with the server tossing a coin of who would have the first turn, and the server would alternate turns between the two players.
7- Your game must be multi threaded – and handle synchronization of lists.
8- Have a celebratory flash screen when a win happens.
9- You must design a protocol (text or objects are fine). Your final Latex project report should highlight the message structure of your protocol.
10- Install the server on a cloud server or a pi that you put in the cloud and make sure you can connect to it as a battle ship client.
11- Doing a GUI on the client would be a lot of extra credit.

Option 2 (100 points – Full Grade)

Get a raspberry pi device or use a cloud Linux server. Implement your Patient Doctor Application using on the pi or in the cloud as a make the GUI application connect to it as a client. You can use Python or Java.

Option 3 (80 points – Partial Grade)

Write a technical paper (not a business or marketing paper) on the use of Object Oriented Programming using YOUR own examples of all the topics we covered in class.

1. ***Designing Classes***
2. ***Inheritance***
3. ***Abstract Classes***
4. ***Interfaces***
5. ***Overriding and Overloading***
6. ***Visibility modifiers***
7. ***Lambda Expressions***

You must explain the concepts using your own code examples (no copying of ideas or code from any source – that would get you a zero). You must also discuss new advances in OOP or trends in programming languages appropriate for emerging technologies such as AI, IoT, etc. (Hint: Discuss differences/or complementary aspects of OOP vs. Functional Programming (FP)).

You may want to read:
https://msdn.microsoft.com/en-us/magazine/ee309512.aspx

https://www.manning.com/books/real-world-functional-programming

Your paper must not be less than 10 pages in IEEE Latex format.
Use sharelatex.com to write your paper.