UMass Boston CS 240

Final Project

Due on December 17 at 5 PM (EDT)

This final project is optional, student can choose to either do final project or give final exam.

Description:

In this assignment, you will create a simple banking application that has two modules: a server and a client. The server and client will be connected using socket programming, and the client will be able to login and perform basic banking operations such as deposit, withdrawal, and checking the balance. The data will be encrypted while in transit using a cipher function, and the server will be able to handle multiple clients at once.

Requirements:

1. The server should be able to accept multiple client connections at once.
2. The client should be able to login using a username and password.
3. The client should be able to perform the following operations:
    a. Deposit: add funds to the user's account
    b. Withdrawal: remove funds from the user's account
    c. Check balance: view the current balance of the user's account
4. The data should be encrypted while in transit using a cipher function.
5. The server should save the data for each client, including their account information and login details.

Suggested Implementation Steps:

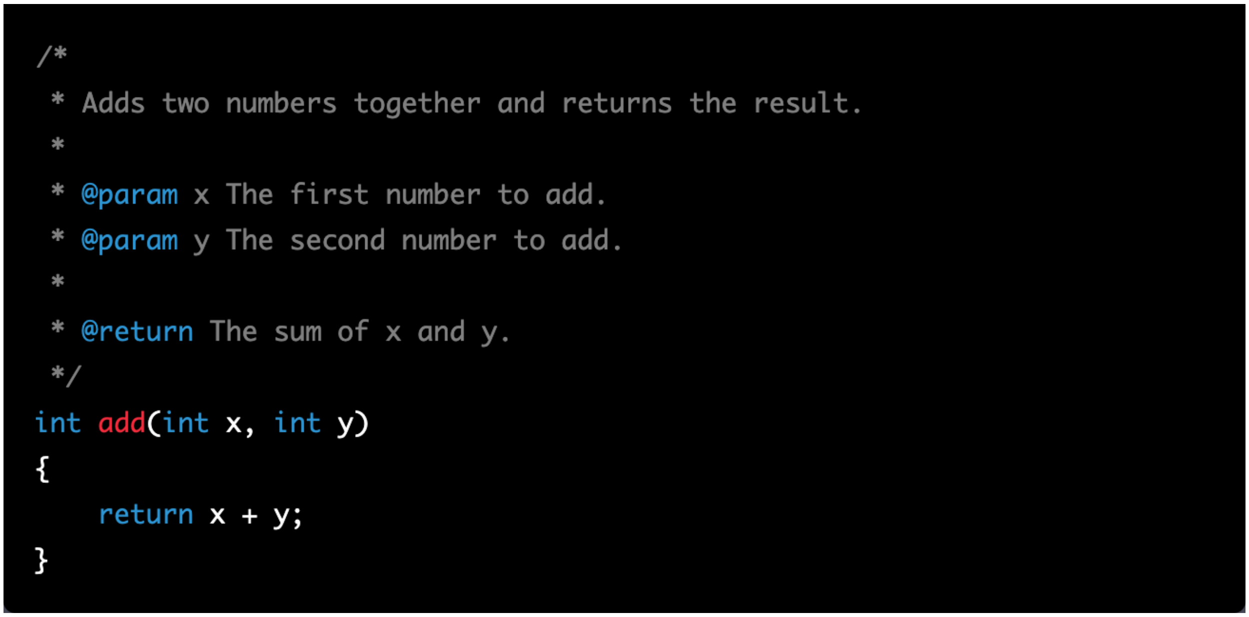
1. Create a server program that listens for incoming client connections using socket programming.
2. Implement the login functionality in the client program, allowing the user to enter a username and password.
3. Implement the deposit, withdrawal, and check balance functionality in the client program.
4. Use a cipher function to encrypt the data while it is in transit between the client and server.
5. Update the server program to save the data for each client, including their account information and transactions.
6. Test the program to ensure that it functions correctly and can handle multiple clients simultaneously.

Documentation

Please include detailed documentation for each function in your program, including a description of what the function does, the input parameters, and the return value. This will make it easier for others to understand and use your code.

Example

Here is an example of the documentation for a function that adds two numbers together:



```
/*
 * Adds two numbers together and returns the result.
 *
 * @param x The first number to add.
 * @param y The second number to add.
 *
 * @return The sum of x and y.
 */
int add(int x, int y)
{
    return x + y;
}
```

Please include similar documentation for each function in your program.

Grading Rubrics:

The grading will be done using the following rubric:

1. Program working as required, and have all the features listed in requirements
2. Documentation + Instructions + Test Screenshots

There will be no partial points of projects which are not executing or have all the required features. Student will either get full marks for final exam and 5% bonus to total grade or none. Please do not share your code and copy from others, if plagiarism is found, then the student will be failed.

Submission:

Please submit the following in the Gradescope:

1. Your completed program
2. Documentation how the program works
3. Instructions for running the program
4. Screenshots PDF that shows test runs

Good luck!