

## Exercise 1

We will begin by building parts of a computerized “game”. The game is rather simple. There is some number of players specified by the exercising system. Each player accepts a bid and sends a bid. The bid that is sent is a function of the internal parts of a player. The player may choose to keep all or part of the accepted bid. The player must keep track of its current balance. The formula for emitting the value of the sent bid is of your own design. Every player must be connected to at least one other player. The starting player will accept a bid from the exercising system. The game is played in rounds. A round ends when every player has emitted a bid. The exercising system will hold the last value emitted by the last player for the next round. At the end of each round the current balance of each player will be reported to the system.

Given this, you will need a class for the players and a class for the exercising system. It is possible to define more classes. The player will need to keep its state at least in terms of a balance and the player to which it is connected, have a method to accept a bid, have a method to compute an emitted bid and a method to emit the bid. It must have methods for reporting to the exercising system. The exercising system must keep its state at least in terms of desired rounds, completed rounds and the number of players and have methods for generating the first bid, reporting the game status including the printing of each player’s current balance at the end of a round and starting a new round if required.

The code should display your awareness of public and private visibility. The exercising system should create the required number of players and their connections.

The exercise will be expanded so that I will post the players that you have created and as a team exercise you will use those players in creating a new game. Thus someone else in the class will use your player code. Obviously good documentation is in order. There will be more modifications as we go along.

### Questions:

What class do you need to define?

What does each class do? What is its function (purpose, intent)?

How do you make an object from a class definition?

What is needed to keep the state of the object?

What are the messages to which a player object should respond?

What messages and states should be protected from other code?

What data types should be used for the bids?

How should the player objects be stored?

In the exercising system how should the state be stored?

What objects should be allowed to modify the state of the exercising system?