Online Auction System

Background

An auction is a way to sell one or more items. Items are grouped into lots for the auction; an auction can have several lots and each lot has one or more item in it.

During the period of an auction, individuals state a price, called a bid, they are willing to spend to buy a specific lot. At the end of the auction period, the individual with the highest bid on a lot gets all the items in the lot for the bid value.

During an auction period, an individual can place multiple bids on a lot and they cannot withdraw or rescind a bid once it is made.

Auctions often have a minimum amount by which a bid can increase over the current highest bid. For example, if the minimum increment is $5 and the highest bid for a lot right now is $10 then the next higher bid for that lot must be $15 or more. A bid of $11 in that case would not be accepted. The minimum bid also applies to the opening bid: even if there is no bid on the lot, we won't accept a bid for the lot whose value is below the minimum increment.

We will consider an online auction where the bids arrive electronically. In an online auction, the auction period is several days to give people time to bid without being online all the time.

If ever the winning bidder for a lot re-bids on the lot with a new bid, that bid can only increase the remembered maximum bid. The bidder remains the winner of the lot.

Problem

Write a class that will track and report on several auctions.

The data that you are working with are

- A set of bidders
- A set of lots
- A set of auctions

A lot belongs to a single auction. An auction has 1 or more lots. A bidder can bid on zero or more lots from zero or more auctions.

The normal use of your class will have the following sequence:

- Create an auction, which assigns lots to the auction
- Add a bidder to the system
- Open an auction for bidding
- Retrieve the current bid on a lot
- Place a bid on a lot
- Close an auction

- Report on the winning bids for all lots in an auction

Multiple auctions can be open at the same time.

You will have classes of Auction, Bidder, and Lot.

The Auction class must include the following methods:

- A constructor Auction(String auctionName, int firstLotNumber, int lastLotNumber, int minBidIncrement)
- boolean openAuction ( )
- boolean closeAuction ( )
- String winningBids ( )
- String auctionStatus ( )

The Bidder class must include the following methods:

- A constructor Bidder (String bidderName)
- int getBidderId ( )
- String feesOwed ( )

The Lot class must include the following methods:

- int placeBid (int bidderId, int bid)

You may add additional classes or additional methods if you want.

Functionality

Auction createAuction (String auctionName, int firstLotNumber, int lastLotNumber, int minBidIncrement)

Create an auction with the given name and with lots given by the range of lot numbers (inclusive, so this auction contains both firstLotNumber and lastLotNumber). The lot numbers are positive integers and cannot overlap with any other auction.

Bidder createBidder (String bidderName)

Create a bidder with the given name. We are allowed to have more than one bidder with the same name. Assign bidder numbers to your bidders. Bidder numbers should be positive integers that increase from 1 as the bidders are created.

String auctionStatus ( )

Report on the state of the auction. The auction will appear as a line in the returned string and ending with a carriage return (\n).

For the auction, report the auction name, status (new, open or closed), and the sum of currently-winning bids in the auction. Each of these fields for the auction will be separated from the others with a tab character.

An example of the output for an auction called “auction A” that is open with $1024 in bids so far would look as follows, when printed:
auction A	open	1024

and, as a string in Java, would be seen as (adding surrounding quotes):
“auction A\topen\t1024\n”

Return null if there is an error, and a valid (possibly empty) string otherwise.

int placeBid (int bidderId, int bid)

Record a bid for a lot number according to the bidding rules described in the background section, including any automatic bids that result from this current bid. The bid value can only be a positive integer.

We will use return values to identify the outcome (advanced Java programmers may notice that exceptions would be better in some cases, but we will cover those later, so do not add exceptions):
0 – an outcome happened that isn’t covered by the rest of the return codes
1 – the bid was not accepted
2 – the bid was accepted, but it is not the currently leading bid for the lot
3 – the bid was accepted, is the leading bid for the lot, and there will be no further automatic bids made for this bidder on the lot

String feesOwed ( )

Report on the state of a bidder. The bidder will appear as a line in the returned string and will end with a carriage return (\n).

For the bidder, report the bidder’s name, the number of lots in _closed_ auctions that the bidder has won, and the sum of the bids for lots that the bidder has won in _closed_ auctions.

A sample output for A bidder (Alice), when printed, is
Alice	5	200
In which Alice won 5 lots totaling $200.

Return null if there is an error, and a valid (possibly empty) string otherwise.

boolean openAuction ( )

Sets an auction as open. That auction cannot already have been opened in the past.

Return true if the auction has been opened for bids and false otherwise.

boolean closeAuction ( )

Sets an auction as closed. You cannot close an auction that isn't open.

Return true if the auction has been closed for bids and false otherwise.

String winningBids ( )

For each lot in the auction (reported in order of lot numbers), report the lot number, the current maximum bid for the lot, and the id of the bidder who is winning the lot. Separate each field with a tab character and separate each lot line with a carriage return (\n).

If a lot has no bid then the bidder winning the lot is bidder number 0.

Consider an auction with 3 lots (123, 124, and 125). We would get the following output, when the string is printed to the screen, of

123 42 5
124 0 0
125 64 12

Representing lot 123 bid by bidder 5 for $42, lot 124 with no bids, and lot 125 by bidder 12 for $64. As a string, that output would be (with quotes added)

"123\t42\t5\n124\t0\t0\n125\t64\t12\n"

Return null if there is an error, and a valid (possibly empty) string otherwise.

int getBidderId ( )

For the bidder object, return the bidder id to be used in the bidding process.

Assumptions

You may assume that

- All bids are integers.
- Bidder names and auction names do not contain any special characters (including tabs).
- Each line in the file of bids is at most 80 characters long.
- You don't need to worry about two bids trying to happen at identically the same time.

Constraints

- Write your solution in Java. You are allowed to use data structures from the Java Collection Framework.