Pumpkin trading platform

Your task is to create a Java component for a pumpkin trading platform. The signature of the exposed class must contain “buy” and “sell” methods using which the client can submit the suggested price he wants to buy/sell one pumpkin for. The buy and sell orders must be matched by these rules:

* Buy and sell orders are matched if the buy price is not less than the sell price
* If client makes a buy/sell order and there are multiple suitable sell orders, the one with the lowest/highest price is chosen. Ties are resolved by order submission time, i.e. the earliest counter-order wins. The resulting trade price is the price specified by the buyer/seller.
* If there are no valid matching order for the order, it must remain in the system until a matching order is found (it might as well not happen).
* An order can be matched only once.

The exposed class must also provide a method to obtain the array of trades performed which can be output in human readable format.

Example

1. Client A wants to buy a pumpkin for 10€. This is the first order so it can’t be satisfied yet.
2. Client B wants to buy a pumpkin for 11€. Still no one wants to sell a pumpkin.
3. Client C wants to sell a pumpkin for 15€. Neither A nor B wants such an expensive pumpkin.
4. Client D wants to sell a pumpkin for 9€. Finally a trade can happen. Following the rules above, D will sell his pumpkin for 9€ to the client B.
5. Client E wants to buy a pumpkin for 10€. Still no trade.
6. Client F wants to sell a pumpkin for 10€. Trade is created. F sells his pumpkin for 10€ to the client A (as A stated his order earlier than E).
7. Client G wants to buy a pumpkin for 100€. Client C is the only one selling a pumpkin now, so the deal is made for 100€.

Using the trade retrieval method, system should be able to output such 3 rows:

* D sold a pumpkin to B for 9€
* F sold a pumpkin to A for 10€
* G bought a pumpkin from C for 100€

Bonus points

Extra points will be given if

* Code is self-explanatory and doesn’t need comments;
* There are Unit tests;
* Code uses the best data structures for the job available found in java.util package;
* Component is thread safe, i.e. it will work correctly if used from multiple threads simultaneously;
* Component state is protected from illegal modifications, i.e. only adding orders should be possible;
* You can describe how you would implement these features:
  + Add a parameter specifying the amount of pumpkins to buy or sell
  + Add possibility to specify expiration time for the order
  + Return an object which implements the java.util.concurrent.Future interface when buy/sell method is called.