CS6150: Advanced Programming
Mid-Term Exam

Instructor: John Augustine, CSE, IIT Madras

Sept 23, 2024

Name & Roll Number: Pradeep Peter Murmu CS24M033

You wish to sell some old coins from ancient India. You expect them to go for a good price, but unable to decide how much it should be sold for. So you decide to auction it. Being a computer science major, you decide to automate the process. Moreover, you wish to try out multiple auction methods in the hope of getting the best price.

Important:

- You have no clue how many people will be interested in your coins. So you CANNOT use an array (except in Question 4 where you are allowed to use an array of size $O(k)$). You must use a suitable linked list to maintain the bids in your auction.
- Ties in bids must always be broken in favour of the earlier bid.
- Each question carries 5 marks. However, they are not of equal difficulty. Earlier ones are easier. Complete a problem to the best extent possible before you move on. It will greatly help if your code compiles and can execute test cases effectively.
- You must create your own test cases. We will not be providing them.

Typical Input Format. The first line of input always starts with the auction type (`traditional`, `max`, etc) and the last line is (as usual) just `done`. All lines in between the first and the last are commands that are issued one by one. Most of the commands are bids (indicated by the first string in the line being `bid`), but there will be other commands as well (e.g., `withdraw` that will come up in some auction types).

Typical output Format. The output will be just the name of the bidder followed by the price she/he must pay (with a space in between). Note: The output required in Question 4 is a bit more general.

1. (5 marks) The first auction you implement is called `traditional`. It takes a series of bids as input. Each bid includes the name of the bidder and the price he/she is bidding (i.e., willing to pay). The price must be a positive integer. The same bidder can bid multiple times. Each bidder must bid a value (strictly) larger than the previous valid bids. If she/he bids something smaller (or even equal) to a previous valid bid, then you must ignore that bid. In this traditional auction, bidders cannot withdraw their bids. At the end of the bidding (indicated by a **done** in the input), you must print the winning bidder's name and his bid amount (with a space in between). The winning bid is the maximum valid bid.

   Here is a sample input:

   ```
   traditional
   bid Alice 20
   bid Bob 25
   bid Carol 30
   bid Carol 34
   bid Dave 30
   bid Eve 32
   bid Frank 34
   done
   ```

   The output must be

   ```
   Carol 34
   ```

   because Dave, Eve, and Frank placed invalid bids.

   Important Requirements.

   1. For proper record keeping, you must maintain ***all valid bids*** in order in your linked list. (You are free to choose whether the bids must be ordered from head to tail or vice versa.)
   2. The winning bid must be reported in $O(1)$ time once **done** is encountered.

2. (5 marks) The second auction type you implement is called `max`. Each bid has a name and a bid amount (a positive integer). In this case, ***all bids are valid*** and must be recorded in the linked list in the order in which they are provided as input. When `done` is encountered, the winning bid must be reported in $O(1)$ time. Apart from placing bids, the bidders can also choose to withdraw from the auction. After someone withdraws, you can assume that they will not place any more bids.

Sample Input:

```
max
bid Alice 20
bid Bob 25
bid Carol 30
bid Carol 34
bid Dave 30
bid Eve 32
withdraw Carol ~~eve~~ eve
bid Alice 15
bid Eve 31
bid Dave 31
done
```

The output should be

```
Eve 31
```

because Carol withdrew and Dave was too late (i.e., his bid is the same as Eve's bid, but comes later in the sequence).

Important Requirements.

1. For proper record keeping, you must maintain ***all bids*** (including bids from bidders who have withdrawn from the auction) in your linked list in the order they were input. (You are free to choose whether the bids must be ordered from head to tail or vice versa.)
2. The winning bid must be reported in $O(1)$ time once done is encountered.

A 20 - 25 - 30

31 - 71

3. (5 marks) The third auction type is called the `vickrey` auction (named after the Nobel Laureate William Vickrey). In this type of auction, the winner is the highest bidder (who has not withdrawn), but the price she/he pays is the second highest bid (of a bidder who has not withdrawn).

   Sample Input:

   ```
   vickrey
   bid Alice 20
   bid Alice 15
   bid Bob 25
   bid Carol 30
   bid Carol 34
   bid Dave 30
   bid Eve 32
   withdraw Carol
   withdraw Dave
   bid Eve 33
   done
   ```

   The output should be

   ```
   Eve 25
   ```

   because Eve is the highest valid bidder; Carol does not count because she withdrew. The price Eve pays is only 25 because Dave withdrew, so the second highest valid bidder is Bob, who bid 25.

   For a Vickrey auction to run properly, you need at least two bidders. So with any fewer active bidders (i.e., not withdrawn), the output must be just a single string `incomplete`. Thus, for the following input,

   ```
   vickrey
   bid Alice 20
   bid Bob 25
   bid Carol 30
   withdraw Alice
   withdraw Carol
   done
   ```

   the output must be the following.

   ```
   incomplete
   ```

Some Important Requirements.

1. All bids (even the ones from withdrawn bidders) must remain in the linked list and in the same order as given in the input.
2. Upon encountering `done`, the output must be printed in $O(1)$ time.

4. (5 marks) Suppose you have k coins and you wish to sell them to the top k bidders in a single auction. Thus, you use a generalized form of the Vickrey auction. The first line should be `general 5` indicating that you wish to implement the generalized Vickrey auction with $k = 5$.

   In the generalized Vickrey auction, the top k bidders are the winners and they will each have to pay the bid of the $(k + 1)$th bidder in order to collect their respective coins. For a generalized Vickrey auction to work, you will need at least $k + 1$ valid bidders (i.e., bidders who have not withdrawn). If the auction has fewer than $k + 1$ bidders, you must output `incomplete`. For this problem, you are allowed to use an array that can contain $O(k)$ items.

   Sample Input:

   ```
   general 3
   bid Alice 20
   bid Bob 25
   bid Carol 34
   bid Dave 30
   bid Eve 32
   withdraw Carol
   withdraw Dave
   bid Eve 33
   done
   ```

   The output should be `incomplete` because the number of active bidders is fewer than $k+1 = 4$. However, the following input will have a more interesting output.

   Sample Input:

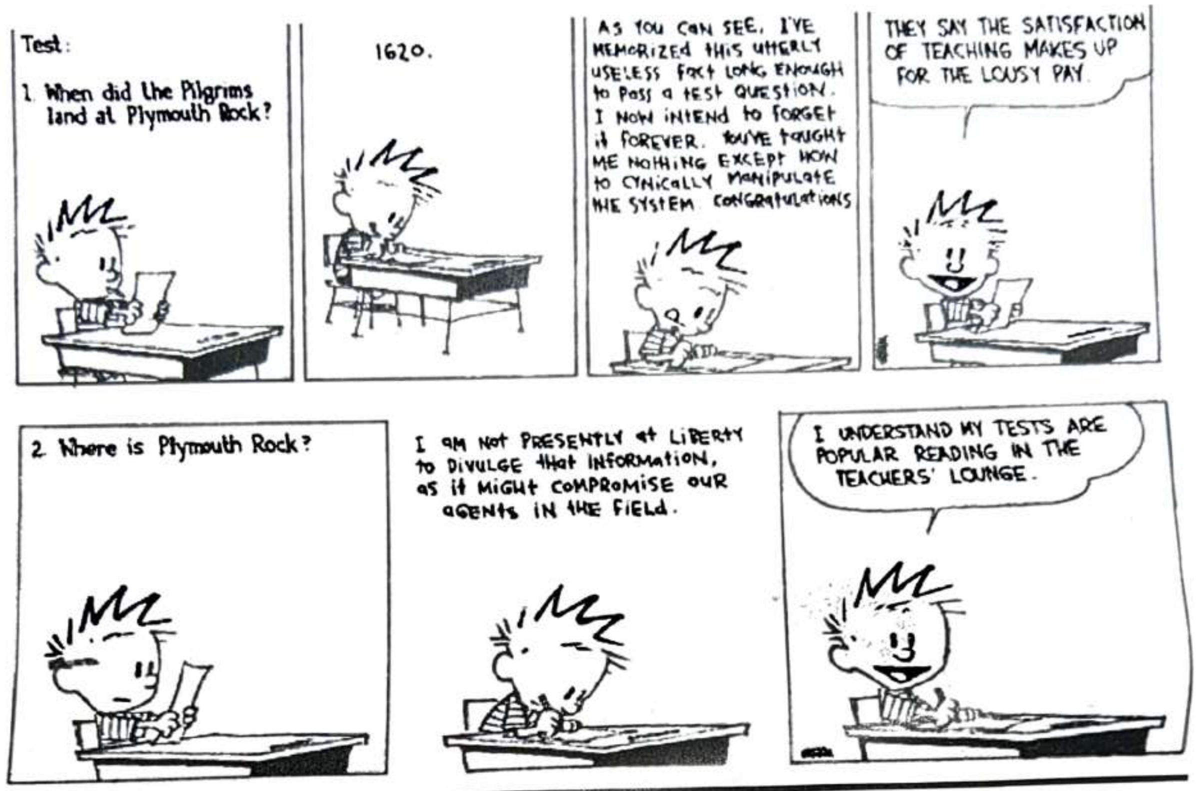
   ```
   general 3
   bid Alice 20
   bid Bob 25
   bid Carol 34
   bid Dave 30
   withdraw Carol
   bid Eve 33
   bid Dave 25
   bid Frank 30
   bid Alice 27
   withdraw Alice
   done
   ```

   The output should be

   ```
   Eve Frank Bob 25
   ```

   in that order because Eve, Frank and Bob are the first, second, and third highest bidders, respectively. Dave is the fourth highest bidder. Notice that Alice has a higher bid than Dave's bid, but she will not count as she has withdrawn. In fact, Dave is tied with Bob, but Bob is considered third highest bidder because his bid for 25 came before Dave's bid for 25.

   Upon encountering `done`, the winners must be computed in $O(k)$ time.


Test:
1. When did the Pilgrims land at Plymouth Rock?
1620.
AS YOU CAN SEE, I'VE MEMORIZED THIS UTTERLY USELESS FACT LONG ENOUGH TO PASS A TEST QUESTION. I NOW INTEND TO FORGET IT FOREVER. YOU'VE TAUGHT ME NOTHING EXCEPT HOW TO CYNICALLY MANIPULATE THE SYSTEM. CONGRATULATIONS.
THEY SAY THE SATISFACTION OF TEACHING MAKES UP FOR THE LOUSY PAY.
2. Where is Plymouth Rock?
I AM NOT PRESENTLY AT LIBERTY TO DIVULGE THAT INFORMATION, AS IT MIGHT COMPROMISE OUR AGENTS IN THE FIELD.
I UNDERSTAND MY TESTS ARE POPULAR READING IN THE TEACHERS' LOUNGE.