Fidelity International Python Developer Hiring Test - Exp... 15m to test end 0/4 Attempted Vivek Ratnaparkhi

1
2
3
4

Initial Public Offering

A company registers an IPO on a website *sellshares.com*. All the shares on this website are available for bidding for a particular time frame called the bidding window. At the end of the bidding window an auction logic is used to decide how many of the available shares go to which bidder until all the shares that are available have been allotted, or all the bidders have received the shares they bid for, whichever comes earlier.

The bids arrive from the users in the form of <*user Id, number of shares, bidding price, timestamp*> until the bidding window is closed.

The auction logic assigns shares to the bidders as follows:

1. The bidder with the highest price gets the number of shares they bid for
2. If multiple bidders have bid at the same price, the bidders are assigned shares as follows:
   Each bidder in the same price group gets assigned one share each consecutively, with each bidder being arranged inside the group based on their timestamp. Once a bidder gets the number of shares they bid for, they will be removed from the above iterative process and the process which then continues until all bidders are removed or the shares get exhausted, whichever comes first.

List the user Id's of all users who did not get even one share after the shares have been allocated.

For example, bids come in as *bids = [[1, 5, 5, 0], [2, 7, 8, 1], [3, 7, 5, 1], [4, 10, 3, 3]]*. There are *totalShares = 18* to allocate. The highest price bid is for user Id *2* for *7* shares at a price of *8*, so that user gets *7* shares leaving *11* to allocate to lower prices. Users with Id's *1* and *3* each bid *5* for *5* and *7* shares, with bidder *1* having the earlier timestamp. After *5* iterations, *10* shares have been allocated with *5* shares going to each of these two bidders. Bidder *1* has the full allotment, bidder *3* has *2* more shares to buy and there is *1* share left to allocate. It goes to bidder *3* and all shares have been allotted. Bidder *4* is the only bidder who gets no shares.

Function Description

Complete the function *getUnallottedUsers* in the editor below. The function must return a list of integers, each an Id for those bidders who receive no shares, sorted ascending.

getUnallottedUsers has the following parameter(s):

bids[bids[0],...bids[n-1]]: a 2D array of arrays of integers, *Id, shares, price, timestamp* named *u, sc, bp, ts* going forward

totalShares: an integer, the total shares to allocate

Constraints

-
- $1 \le n < 10^4$
- $1 \le u, sc, bp, ts, totalShares < 10^8$

Input Format For Custom Testing

Sample Case 0

Sample Input 0

```
3
4
1 2 5 0
2 1 4 2
3 5 4 6
3
```

Sample Output 0

```
3
```

Explanation 0

Fidelity International Python Developer Hiring Test - Exp... 15m to test end 0/4 Attempted Vivek Ratnaparkhi

1
2
3
4

YOUR ANSWER

We recommend you take a quick tour of our editor before you proceed. The timer will pause up to 90 seconds for the tour. Start tour

For help on how to read input and write output in Python 3, click here.

Original Code Python 3

```
#!/bin/python3···

#
# Complete the 'getUnallottedUsers' function below.
#
# The function is expected to return an INTEGER_ARRAY.
# The function accepts following parameters:
#  1. 2D_INTEGER_ARRAY bids
#  2. INTEGER totalShares
#

def getUnallottedUsers(bids, totalShares):
    # Write your code here

if __name__ == '__main__':···
```

Line: 10 Col: 1

Test against custom input

Run Code Submit code & Continue

(You can submit any number of times)

Download sample test cases *The input/output files have Unix line endings. Do not use Notepad to edit them on windows.*

About Privacy Policy Terms of Service