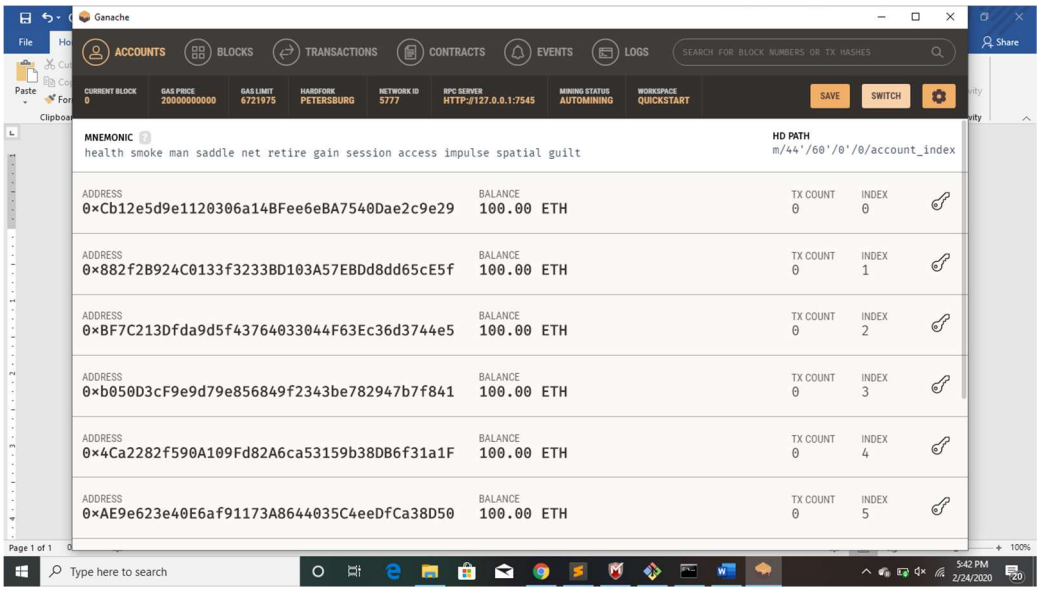
Q.1) Brief explanation for each function you wrote and how it works. It would be great to show some experiments on sending/withdrawing bids to show that your contract works correctly

1. **function bid () public payable ():** The **bid function** allows user to place the bid. Now whether the bid will be placed depends on the amount of new bid. If incoming bid is higher than the already existing bid then the new bid becomes the highest bid and old one is added to pending returns which shall be invoked by user to retrieve the money back in case their bid does not remain the highest. In case the incoming bid is smaller than highest bid the money is sent back to the bidder.

2. **function withdraw () public returns (bool):** The function withdraw sends back the money to the user when they call withdraw function to get their money back in case, they lose the bid. Now, in order to handle the reentrancy, attack I have set a condition where before entering the logic of the code it checks if the pendingReturns for that particular id is 0 then it sends back an exception.

3. **function auctionEnd () public:** Here, the beneficiary is allowed to call this function to end the auction. Once this function is invoked by the beneficiary the highest bid is transferred to the beneficiary and can be seen on the Ganache console.

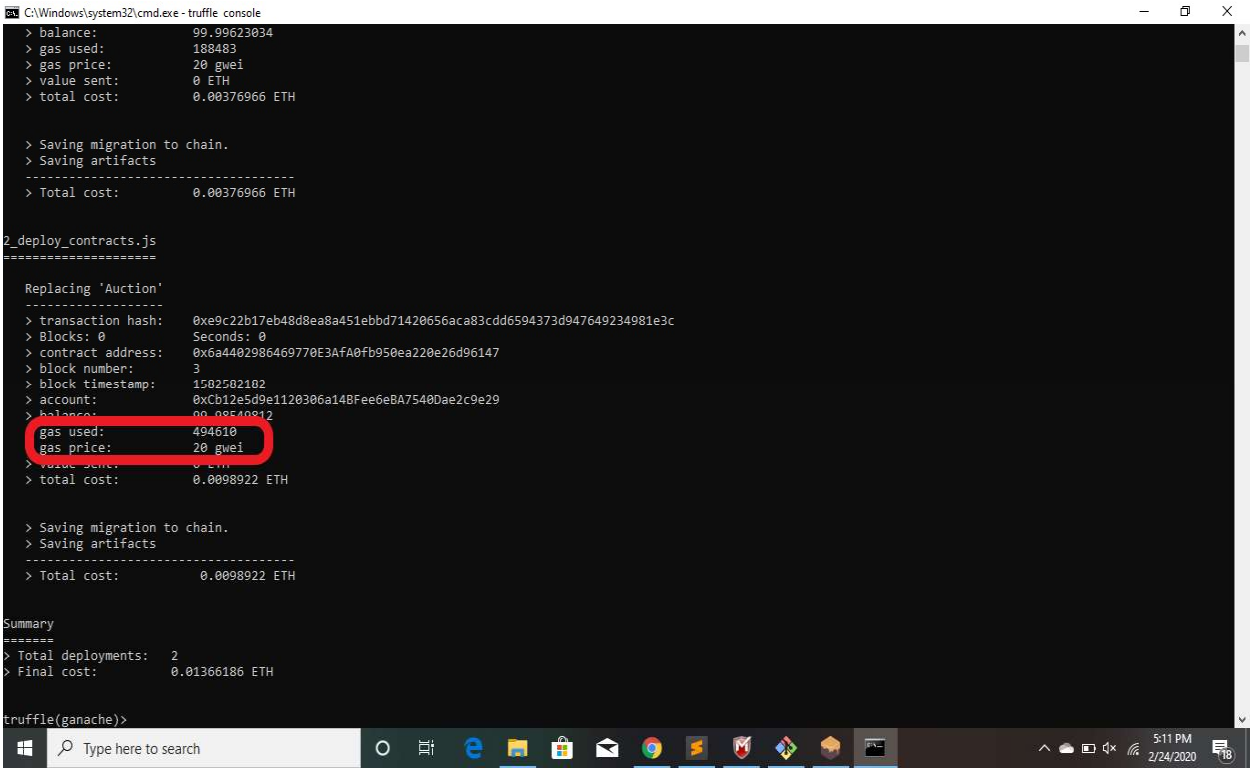
Initial states of the accounts

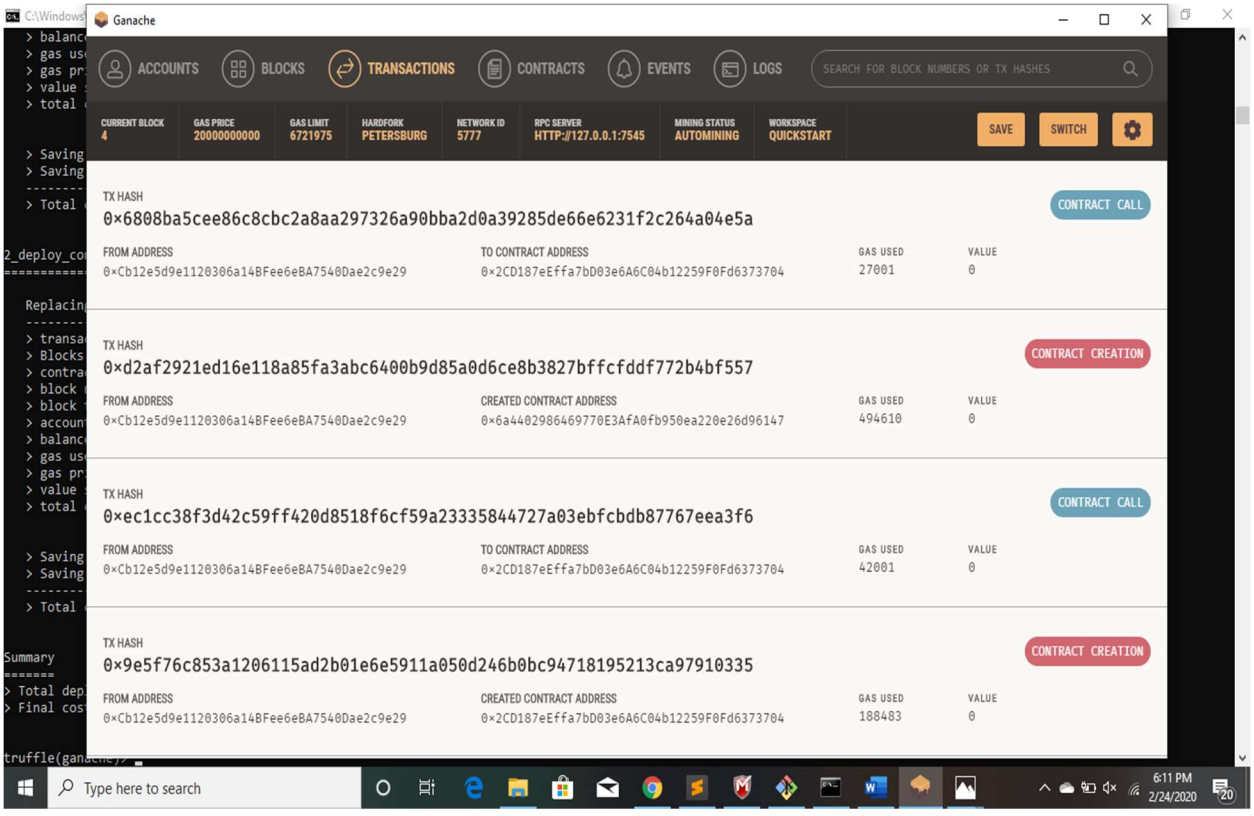


Console after compile and migrate

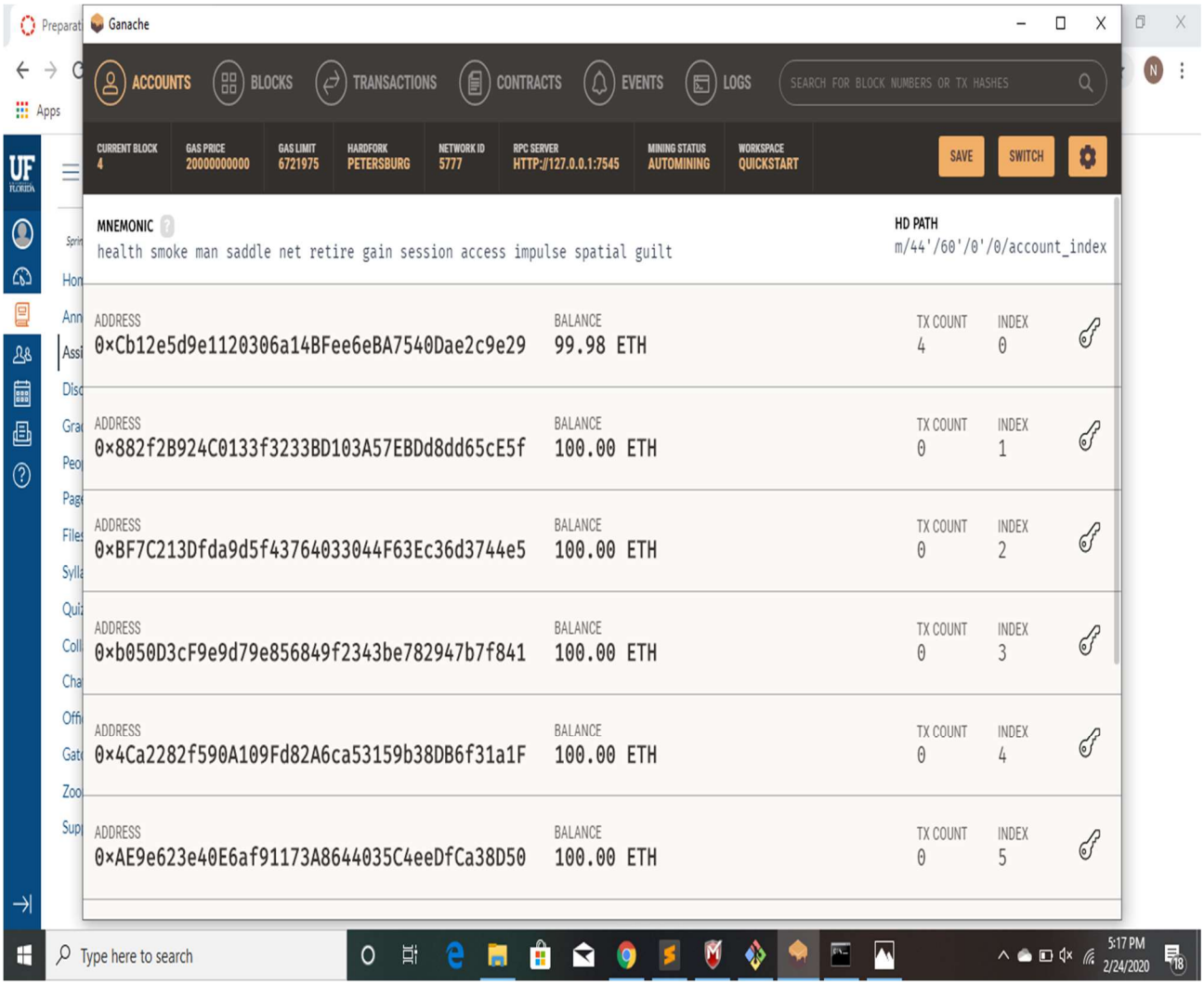
```
Compiling your contracts...
===========================
> Compiling .\contracts\Auction.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to D:\UF sem 2\Blockchain\Homework1\hw1-source\build\contracts
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang


Starting migrations...
======================
> Network name:    'ganache'
> Network id:      5777
> Block gas limit: 0x6691b7


1_initial_migration.js
======================

   Replacing 'Migrations'
   ----------------------
   > transaction hash:    0x9e5f76c853a1206115ad2b01e6e5911a050d246b0bc94718195213ca97910335
   > Blocks: 0            Seconds: 0
   > contract address:    0x2CD187eEffa7bD03e6A6C04b12259F0Fd6373704
   > block number:        1
   > block timestamp:     1582582181
   > account:             0xCb12e5d9e1120306a14BFee6eBA7540Dae2c9e29
   > balance:             99.99623034
   > gas used:            188483
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00376966 ETH


   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00376966 ETH
```

Gas used for deploying the contracts

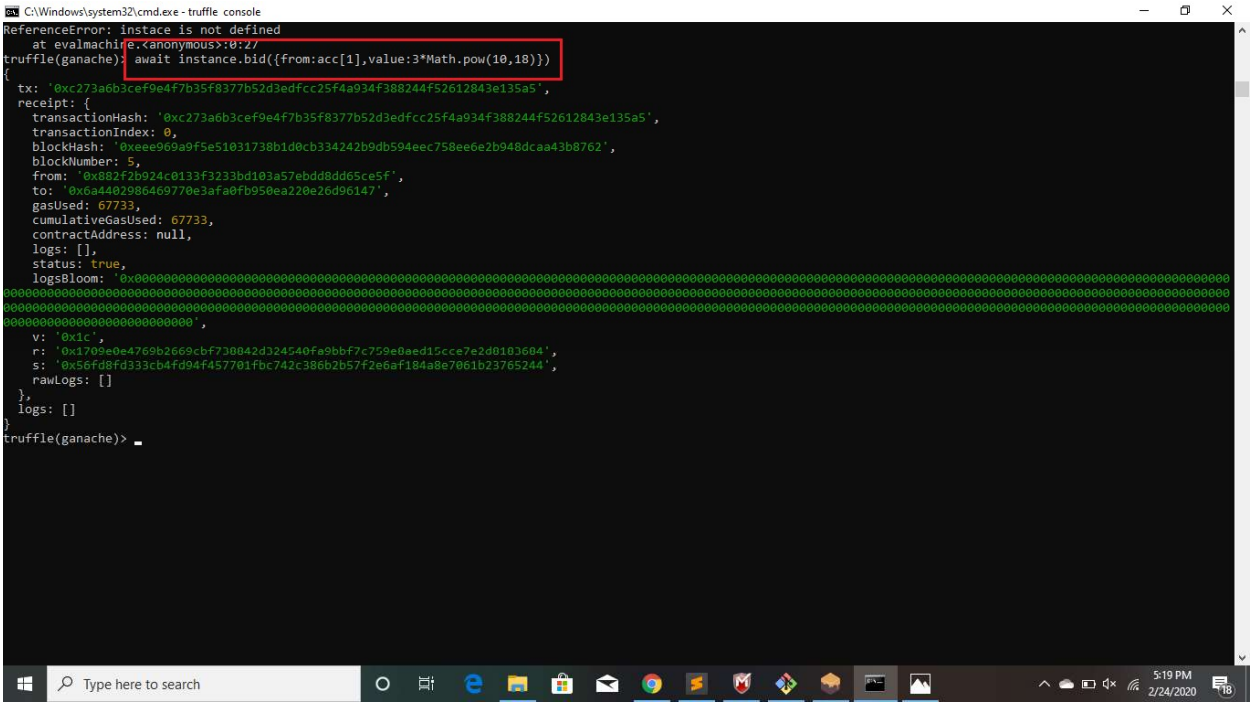




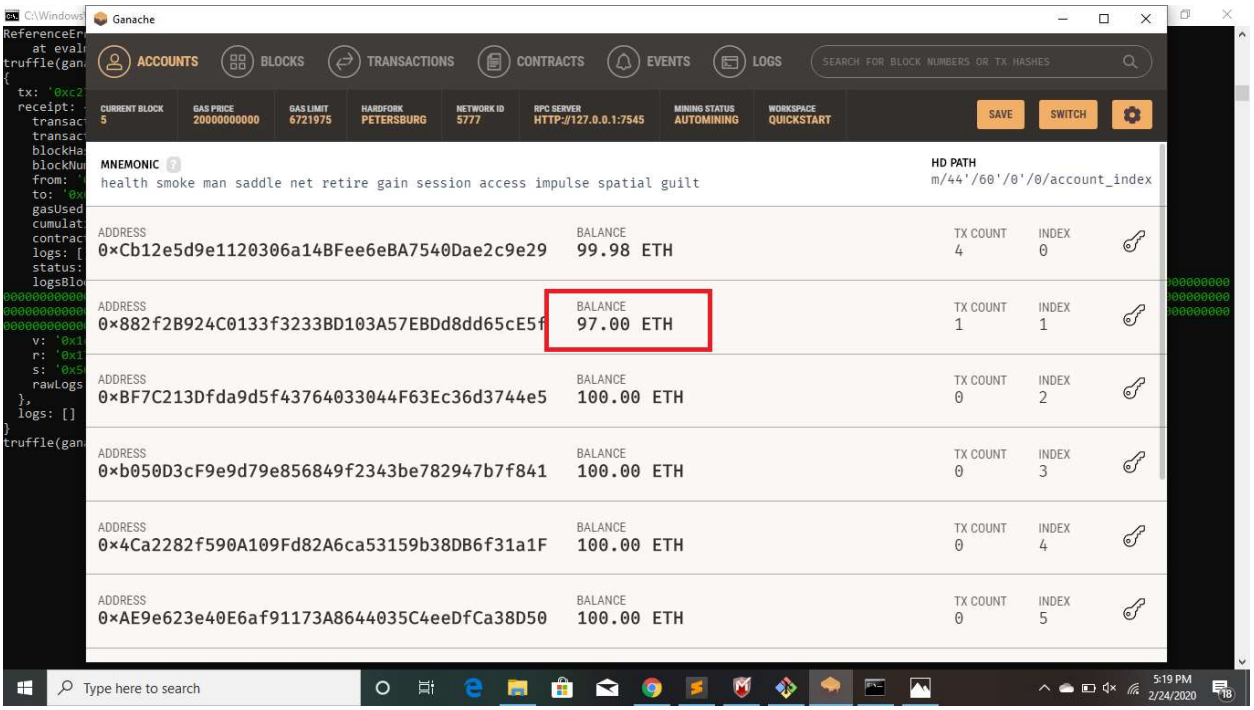
Auction started by the beneficiary



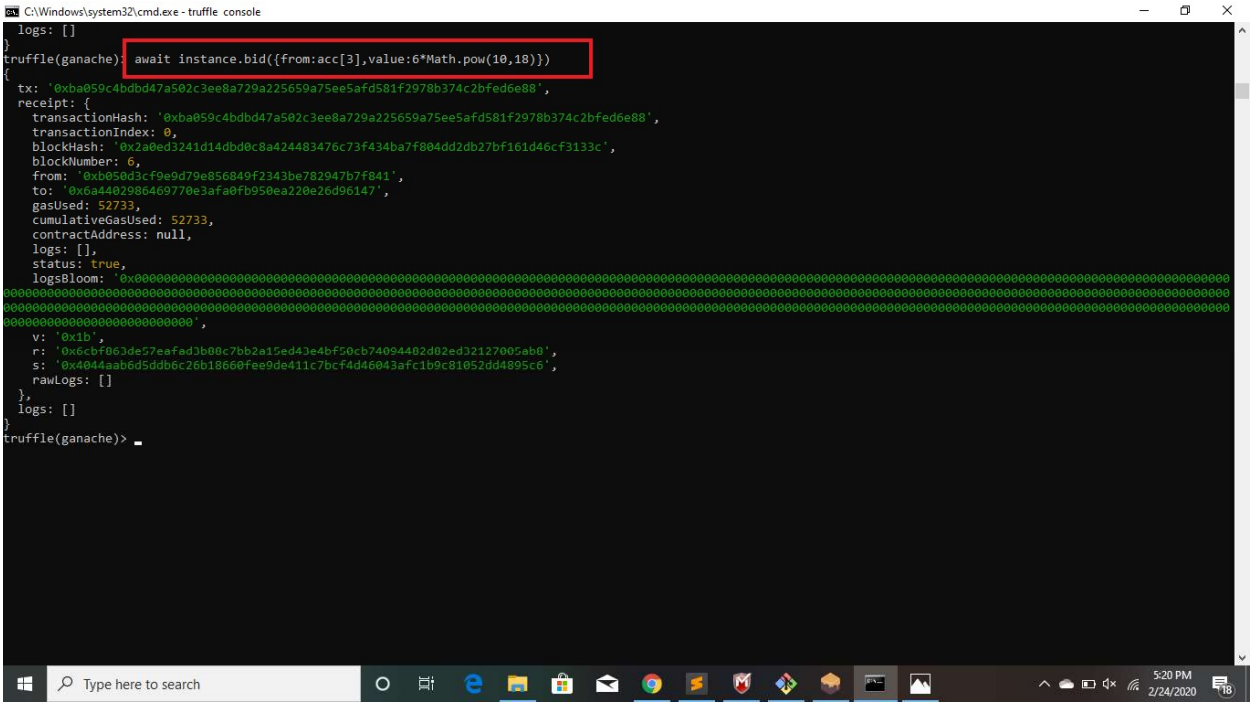
User with Id 1 places a bid of value 3 Ether



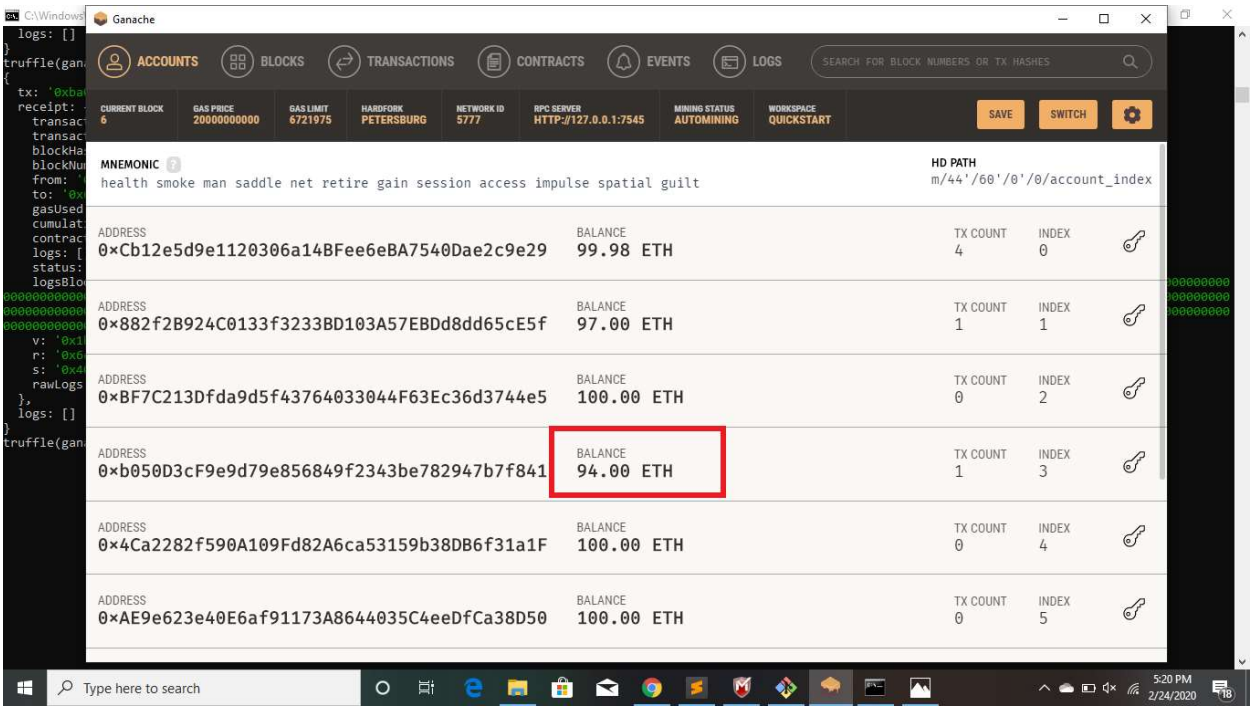
3 ethers deducted from user 1. Value goes from 100 ethers to 97 ethers



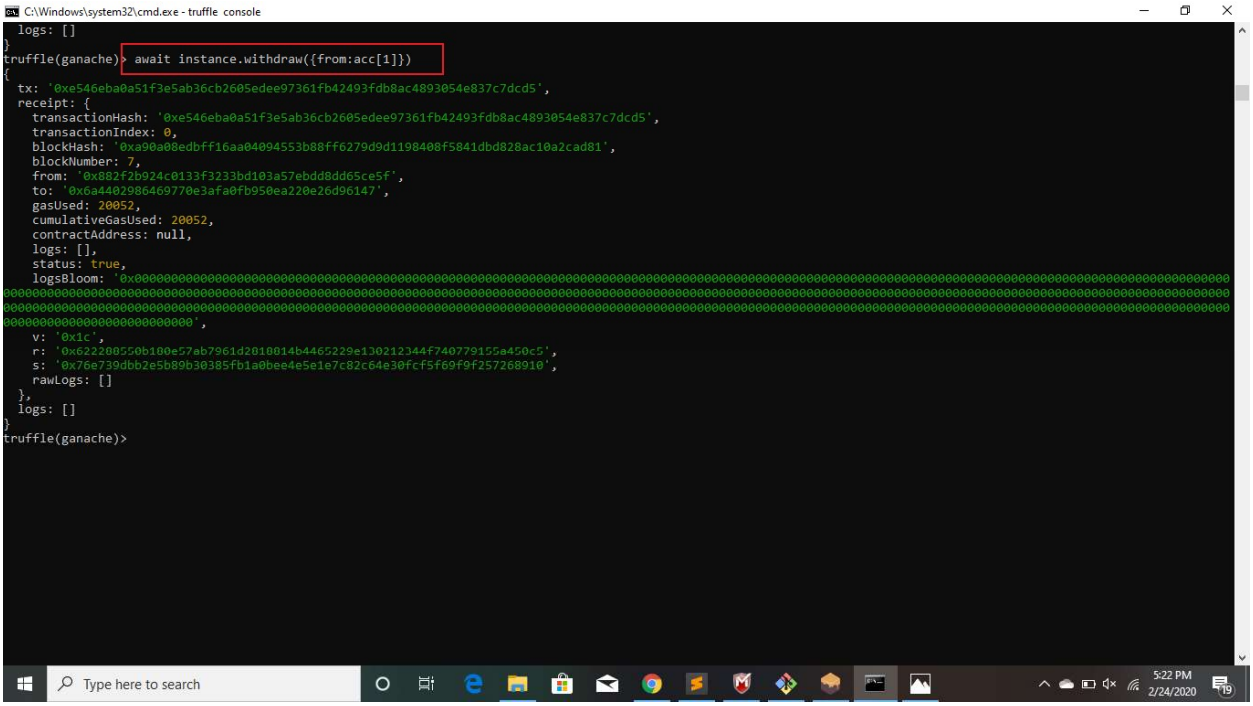
User with id 3 places a higher bet of 6 ethers



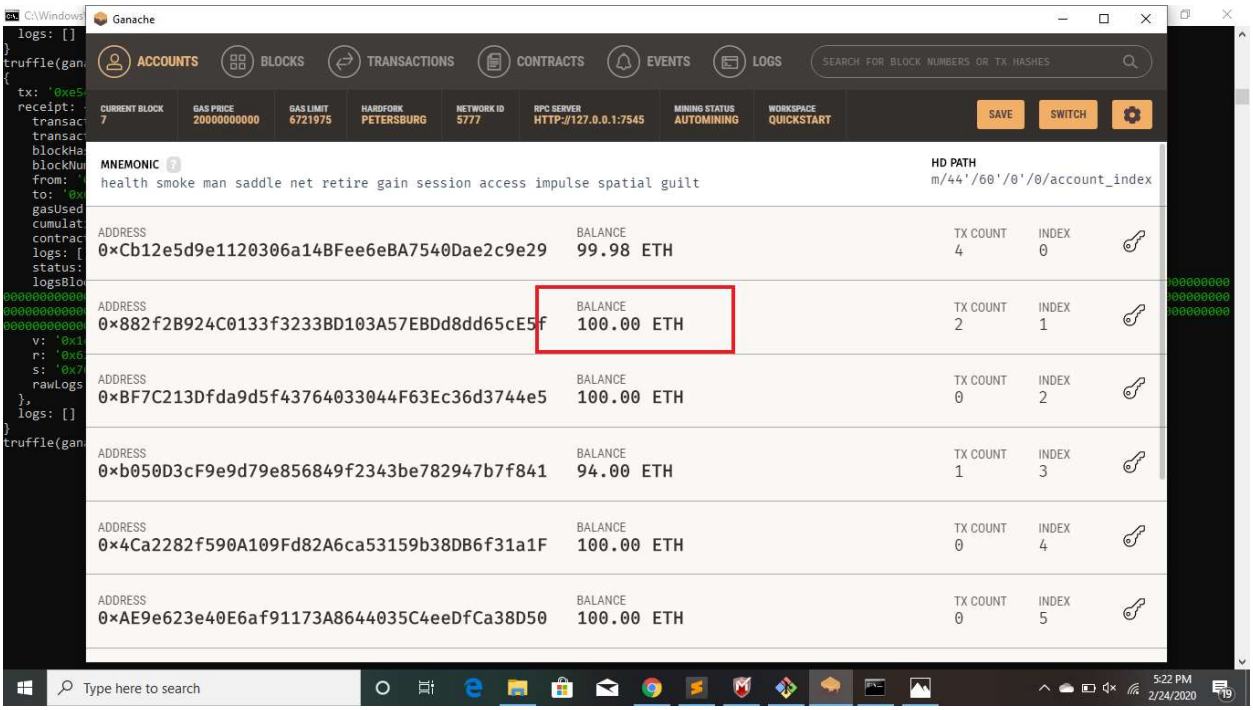
6 ethers deducted from User 3. Value goes from 100 ethers to 94 ethers



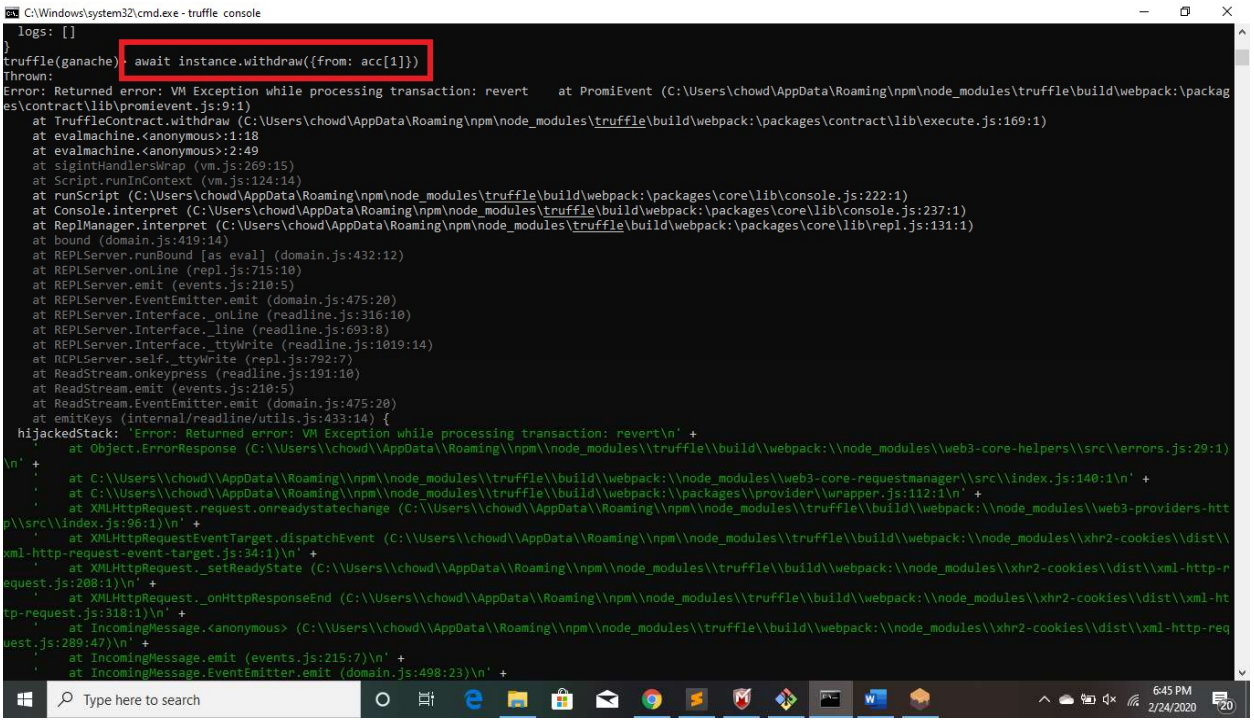
User 1 withdraws



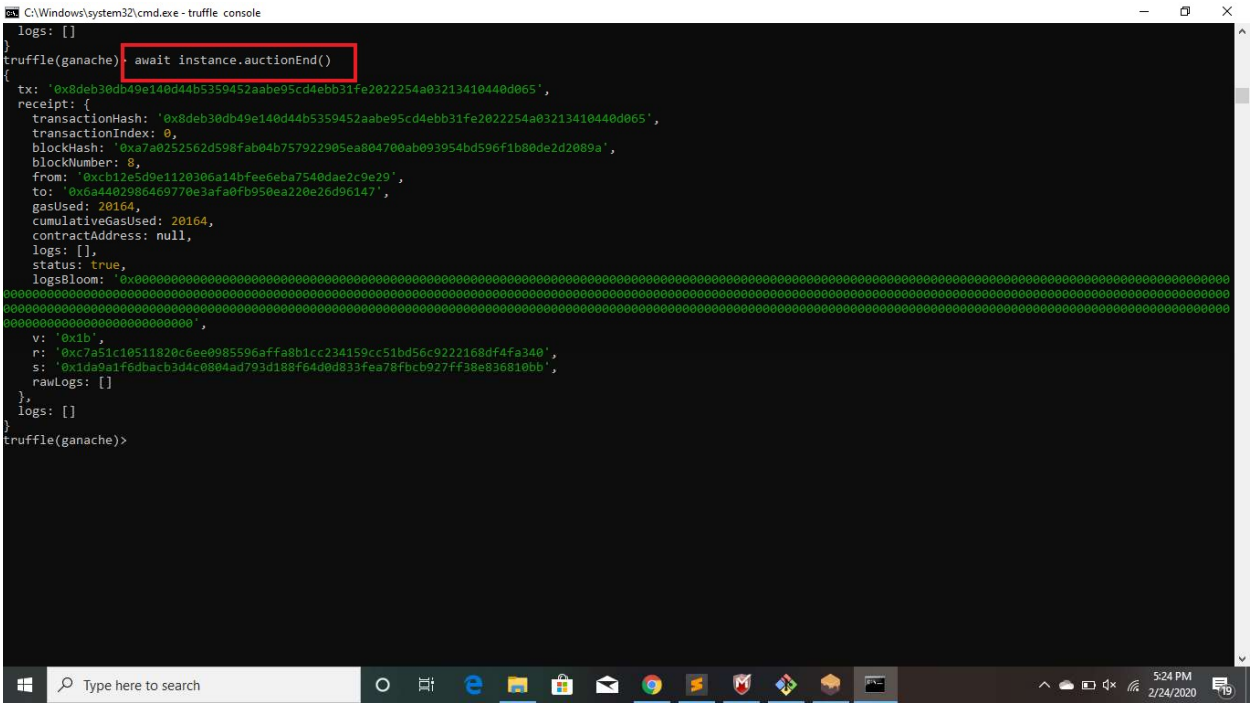
3 ethers sent back to user 1 the ether count changes from 97 to 100



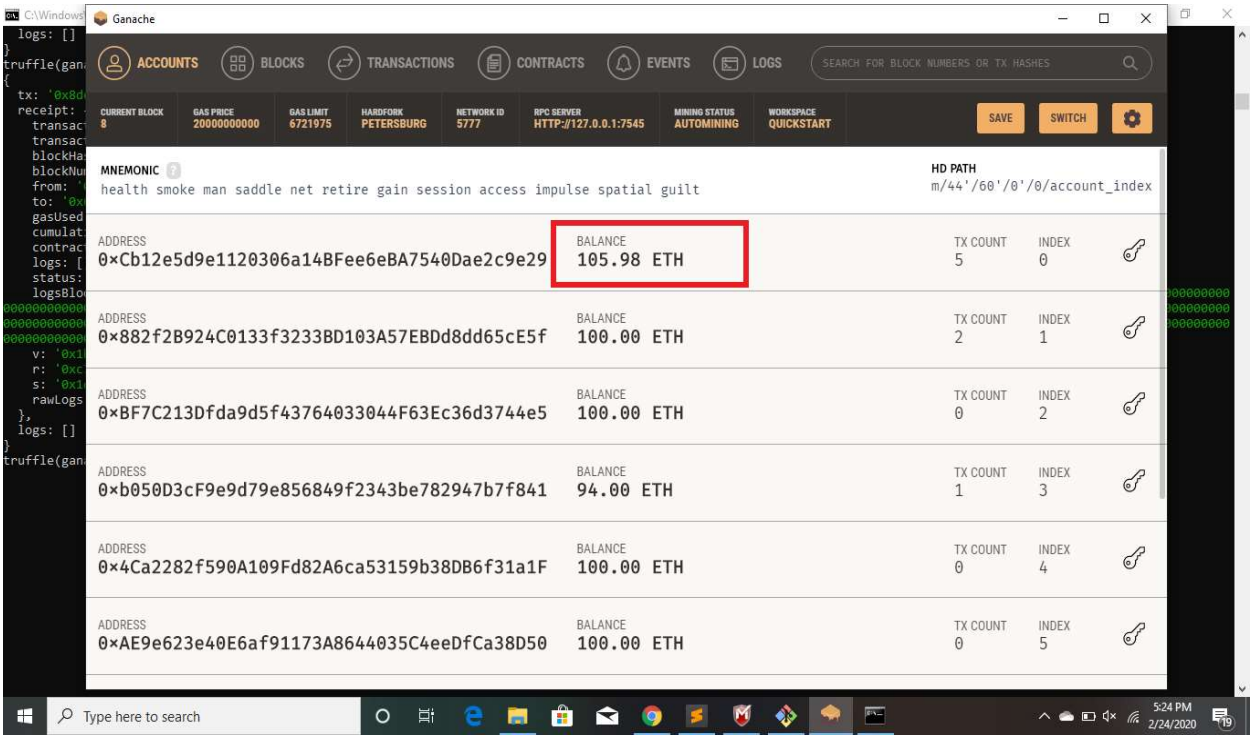
User 1 again tries to withdraw after getting its money. But an error is thrown thus handling reentrancy attack



Auction end is called



Beneficiary gets 6 ethers in his/her account as 6 was the highest bid placed. Value goes from 99.98 ethers to 105.98 ethers



2. Transaction fees

For migration

```
Starting migrations...
======================
> Network name:    'ganache'
> Network id:      5777
> Block gas limit: 0x6691b7


1_initial_migration.js
======================

   Replacing 'Migrations'
   ----------------------
   > transaction hash:    0x9e5f76c853a1206115ad2b01e6e5911a050d246b0bc94718195213ca97910335
   > Blocks: 0            Seconds: 0
   > contract address:    0x2CD187eEffa7bD03e6A6C04b12259F0Fd6373704
   > block number:        1
   > block timestamp:     1582587801
   > account:             0xCb12e5d9e1120306a14BFee6eBA7540Dae2c9e29
   > balance:             99.99623034
   > gas used:            188483
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00376966 ETH


   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00376966 ETH
```

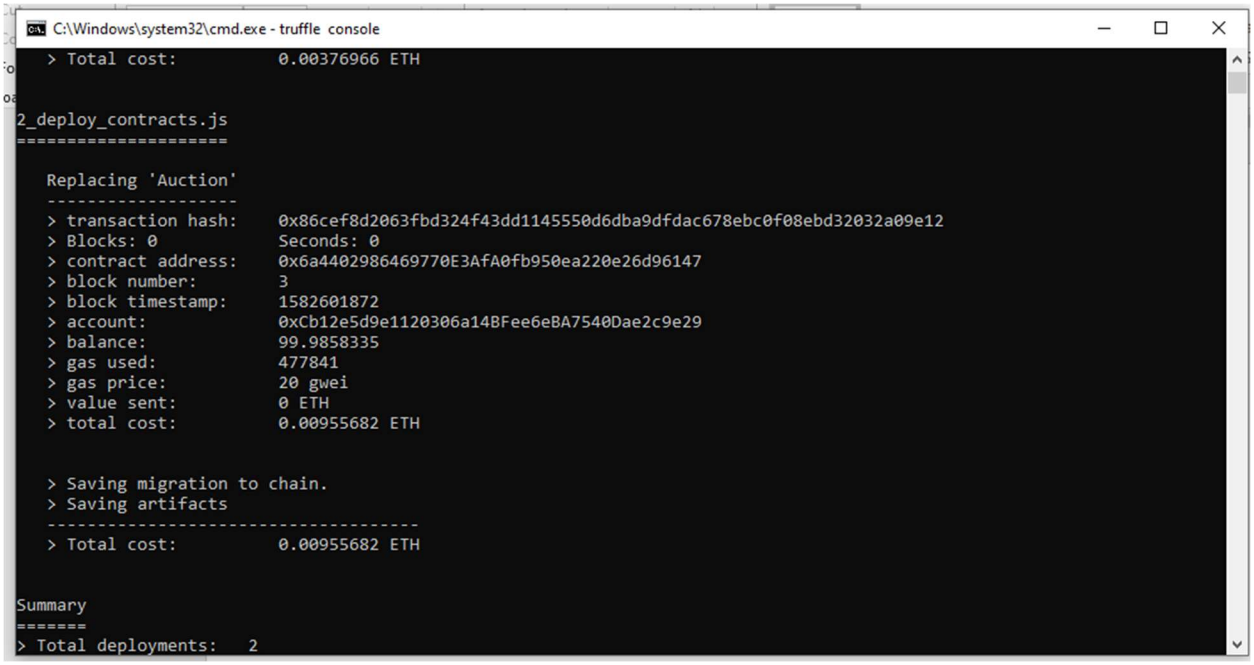
Transaction Fee: gas used * gas price

=188483wei * 20gwei

= 188483*20000000000

=3769660000000000

Deploying Contracts



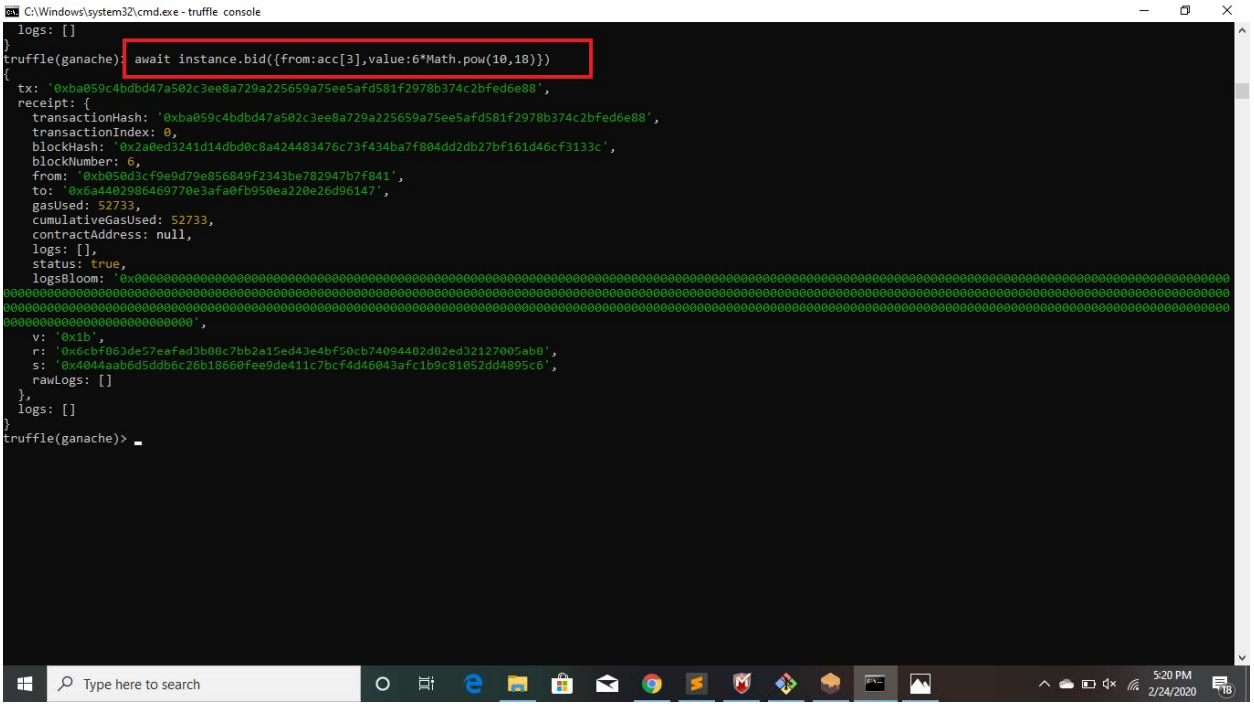
Transaction Fee: gas used * gas price

=477841wei * 20gwei

= 477841*20000000000

=9556820000000000

For function bid ()



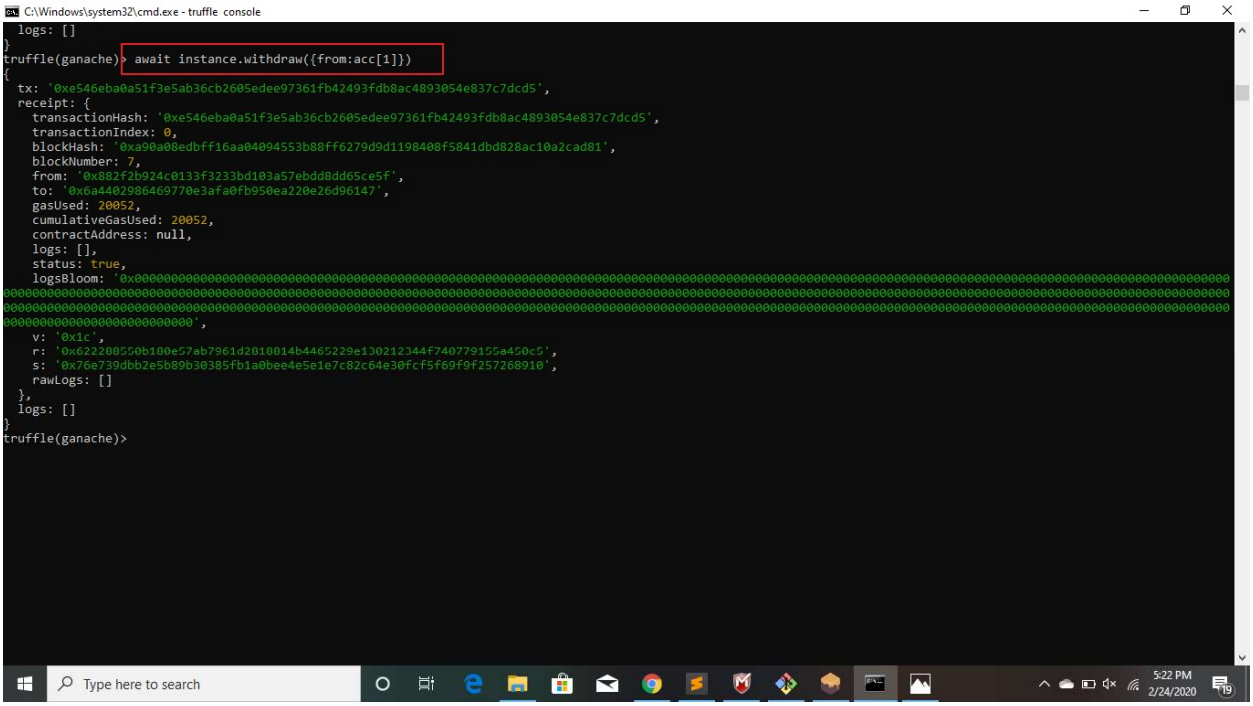
Transaction Fee: gas used * gas price

=52733wei * 20gwei

= 52733*20000000000

=1054660000000000wei

For function withdraw ():



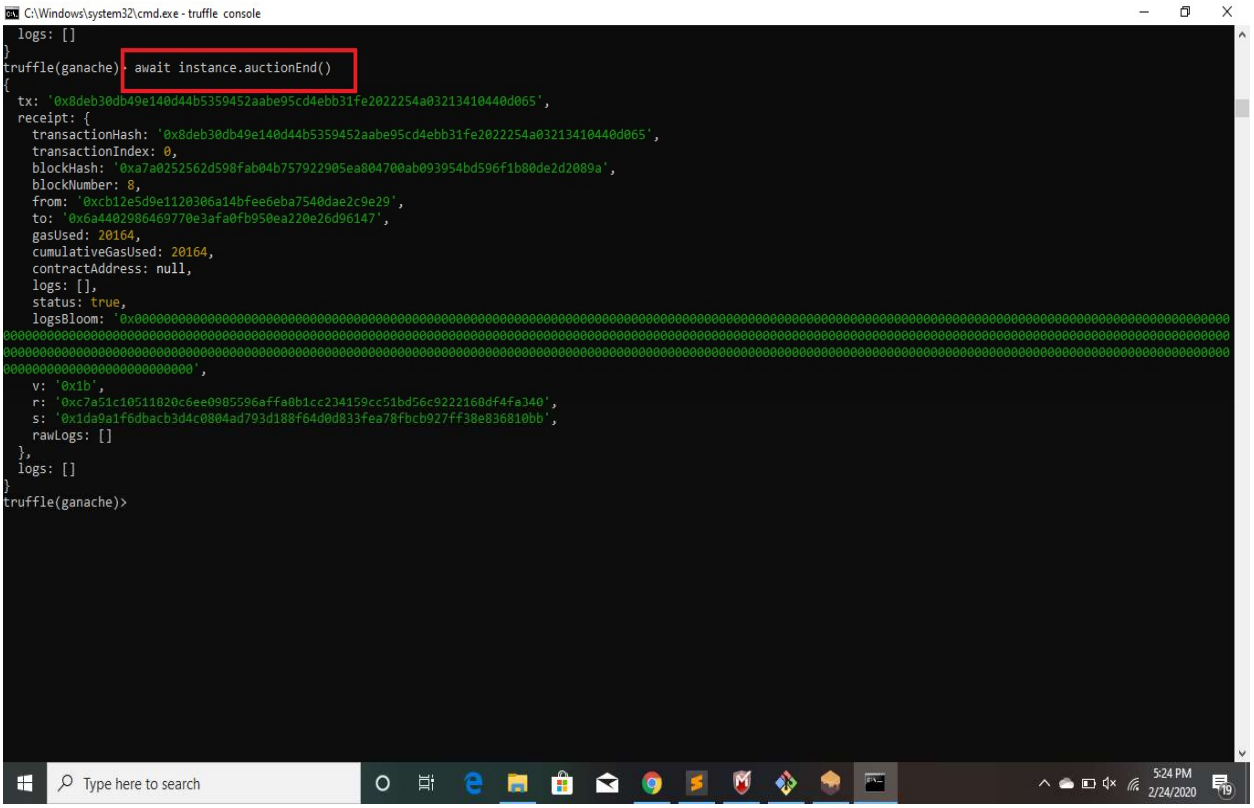
Transaction Fee: gas used * gas price

=20052wei * 20gwei

= 20052*20000000000

=401040000000000wei

For function auctionEnd ():



Transaction Fee: gas used * gas price

=20164wei * 20gwei

= 20164*20000000000

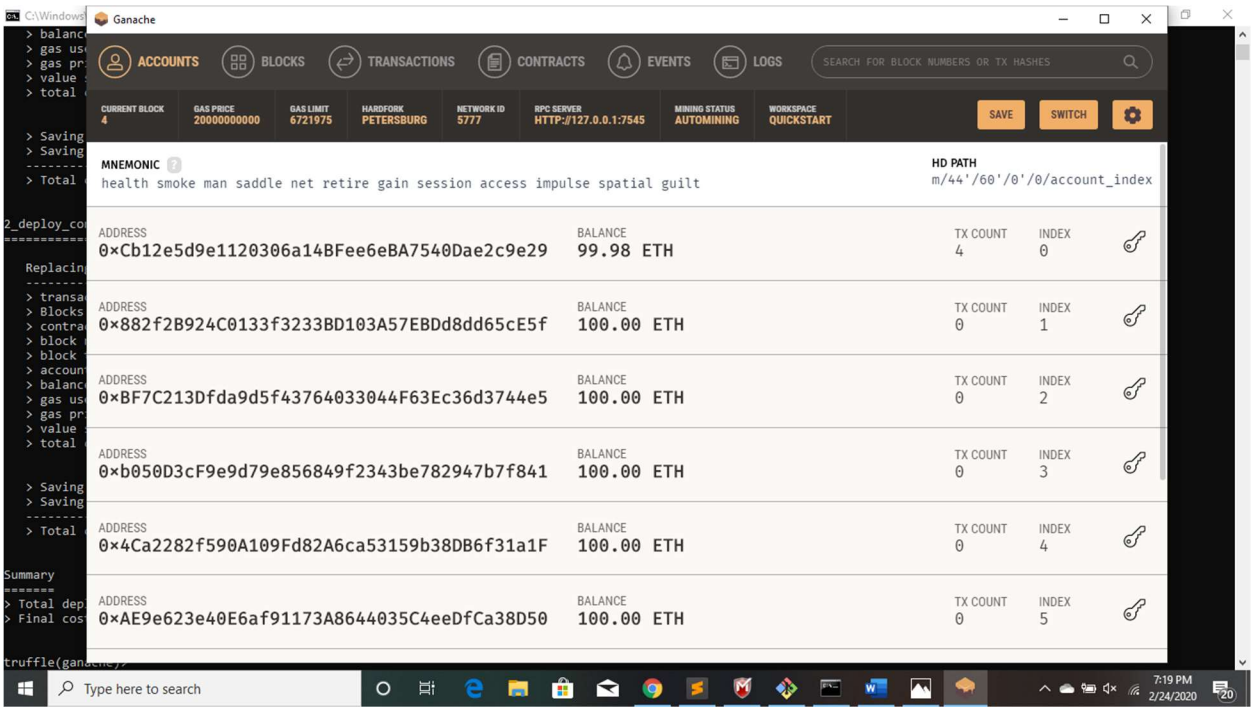
=403280000000000wei

Q.3

Beneficiary/ account [0] balance before the auctionEnd() is called:

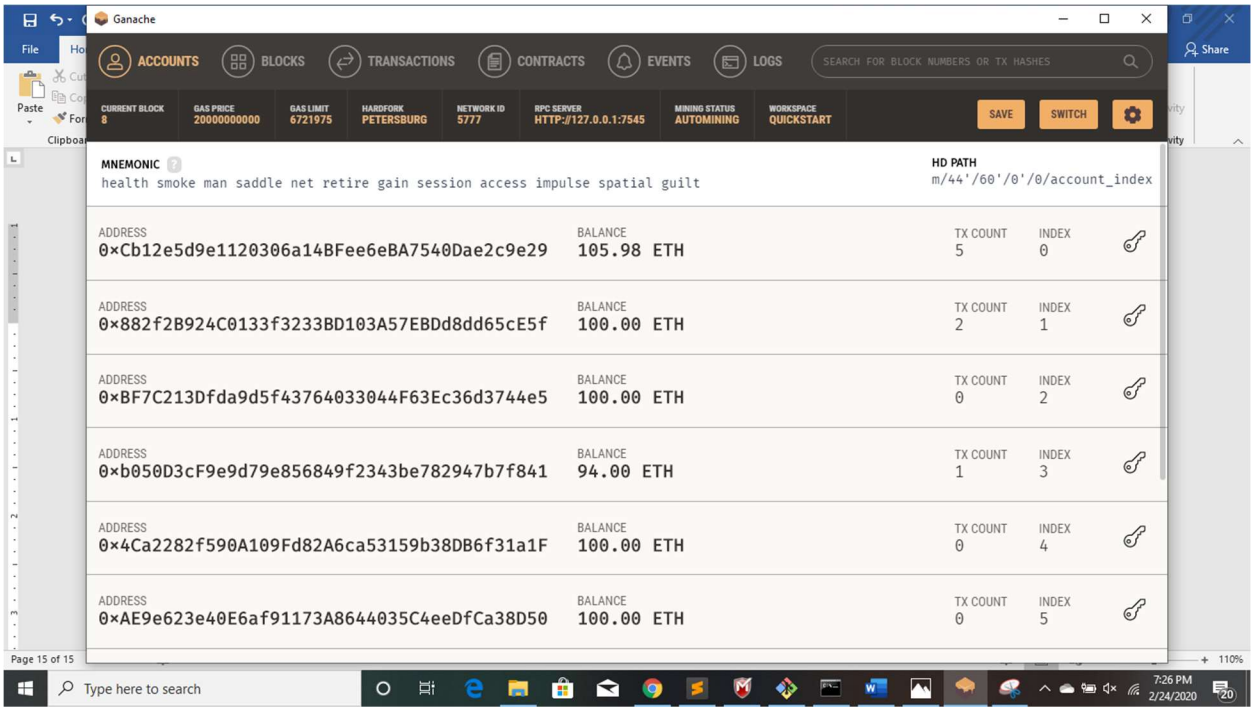
```
}
truffle(ganache)> web3.eth.getBalance(acc[0])
'99984958100000000000'
```

Same is in ganache 99.98ETH for account[0]



Beneficiary/ Account [0] after auctionEnd() is called. Value goes from 99.98 to 105.98 (total increase of 6 ethers that matches the highest bid)

```
truffle(ganache)> web3.eth.getBalance(acc[0])
'10598455482000000000'
truffle(ganache)>
```



Calculation:

Function auctionEnd():

Balance[0]=Balance[0] (before auctionEnd) + highest bid in auction – gas fees used

=9998495810000000000 + 6* Math.pow(10,18) – 29356

=10598455482000000000

Thus, our calculation matches the displayed figure after auction ends.