Hedgey UAT Testing

Purpose: This document outlines all of the tests required to be completed successfully by the smart contracts in consideration below sufficiently to move onto a production environment. For each function within the smart contract, this UAT process will confirm that the when calling a function the result is the intended result and smart contract interaction. This testing does not dive into attempts to break or find vulnerabilities within the smart contract, as the contracts are subsequently going through an audit for that sole purpose.

Smart Contracts in Consideration found in this repository:

HedgeyCalls.sol
HedgeyPuts.sol
HedgeyCeloCalls.sol
HedgeyCeloPuts.sol
HedgeyAnySwap.sol
HedgeyCeloAnySwap.sol

Testing & Functions

Prenotes: We will often refer to the seller of an option as the "short" or "short position", where they have an obligation to fulfill their side of the contract. The buyer of an option we will refer to as the "long" or "long position" to clarify that the long has the Right to do something with the option contract, but is not obligated. We think about options as a pair, with the Asset Currency noted as the underlying currency in which the option contract is based on buying / selling. The Payment Currency is the currency in which the option premium and the strike are priced in. For example a ETH / DAI Call is where ETH is considered the Asset Currency and DAI the Payment Currency. The price paid to purchase a call option would be denominated in DAI, for example 100 DAI. The size (notional or asset amount) of the option is based on ETH, and so it could be an option of size 1 ETH or 2 ETH or 10 ETH. The Strike is denominated in DAI and based on and exchange rate of 1 ETH : DAI, so for example 4,000 would be a strike of 1 ETH : 4000 DAI. The strike is indifferent of the asset size, it is always 1 asset amount : strike payment currency. Note in the testing parameters on paper we write down "decimal" versions, while in the contract they are all uints with 18 digits or 'decimals'.

The "AnySwap" contracts are single use specific contracts that are called from the Calls and Puts contracts for multi-path flash swapping to exercise in the money options in a 'cash light' manner. The Calls and Puts contracts have the following same 16 functions and parameters (including the *normal* vs Celo contracts). The primary difference between the Celo and the Normal version is that the Celo network does not require a msg.value to transfer its native token Celo, it acts as an ERC20, and therefore gas savings are taken into account handling everything as an ERC20. In addition, based on the understanding of the Celo Network ecosystem, many of the pairs will involve 'm' tokens, which are a forke of AAVE and require special handling to handle accrued interest built up in the smart contracts.

Test Number	Functions	Arguments	Purpose	Expected Results
1	newBid	(uint _assetAmt, uint _strike, uint _price, uint _expiry)	For a potential long of an option to place a bid that can be filled by a short	Payment Currency (price amount) delivered into the smart contract as escrow, and creation of the new struct 'Call' or 'Put' with the details of the bid correctly setup
2	cancelNewBid	(uint _c)	Cancel a bid that has been placed but not filled yet	Payment currency (price amount) withdrawn from the smart contract, and the Bid struct showing as 'exercised' such that no one can interact with it anymore
3	sellOpenOptionToNewBid	(uint _c, uint _d, uint _price)	If an option long position (ie already owns an option) sees a newBid in the market, then that person can sell their option to that newBid, and receive the Price for that sale.	Payment Currenct (price amount) delivered from the smart contract to the Open Option Long, and then the structs updating such that the open option long position is replaced with the new bidder's address, and the newBid is shown as 'exercised' and can no longer be interacted with
4	sellNewOption	(uint _c, uint _assetAmt, uint _strike, uint _price, uint _expiry)	When a new bid is in the market, this function is how a seller (short) receives payment for selling the option, and delivering into escrow the asset (calls) or total purchase (puts)	Call: Seller receives payment currency (price) from the smart contract, and delivers in the asset amount to the contract, the struct updates such that the msg.sender (seller) is now the short position, and the call is shown as Open and !Tradeable and !Exercised
5	changeNewOption	(uint _c, uint _assetAmt, uint _minimumPurchase, uint _strike, uint _price, uint _expiry)	Function to change a newBid or a newAsk before anyone has taken the other side of the trade. All parameters can be changed	If its a newBid, then should see the updated struct and a possible change in the price held in escrow (to account for any differences), if a newAsk then should see the updated struct and a change in the assetAmt (calls) or total purchase (puts) to account for any difference in the assetAmt or strike
6	newAsk	(uint _assetAmt, uint _minimumPurchase, uint _strike, uint _price, uint _expiry)	Function to create a new Ask - whereby buyers can then purchase amounts up to the total and greater than the minimum increment	Msg.sender delivers in the escrow crypto (asset for Calls and total purcahse for Puts) to the smart contract, and the struct is created evidencing an option that can be purchased
7	cancelNewAsk	(uint _c)	Function to cancel a newAsk and recevie	For Calls, receive back the asset amount, for puts - receive back the total purchase

			back the escrow crypto	and show the option as exercised
8	buyNewOption	(uint _c, uint _assetAmt, uint _strike, uint _price, uint _expiry)	Function to purchase some amount of a newAsk, with the minimum being the _minimum, and max being the entire assetAmount	Deliver correct payment currency to the Short for the size chunk purchased, evidence that if only a portion was purchased, that portion is created as a new struct with the details matching the Short, Long and amounts purchased, and the 'old' newAsk is still a newAsk updated with the remaining amounts that can be purchased. Need to evidence that two users can buy small chunks at almost the same time without interfering with each other's transactions
9	buyOptionFromAsk	(uint _c, uint _d, uint _price)	Function for an existing Short position to buy an option from an newAsk that matches their current short position, and thus removing them from the obligation and effectively closing out their position	Payment currency price delivered from msg.sender to the short position of the newAsk / open Long, and then the open Ask is updated with the details assigning the short from the ask to the short of the open call
10	setPrice	(uint _c, uint _price, bool _tradeable)	Function to update the price of an option that is currently owned by the Long, or a newAsk. Or to make the option not tradeable	Update the struct to account for updated price, and tradeable boolean
11	buyOpenOption	(uint _c, uint _assetAmt, uint _strike, uint _price, uint _expiry)	Function to purchase an option from an existing long on the secondary market that has been made available for sale	Payment currency price deliverd to the open option seller (long), and then the struct adjust whereby the msg.sender address is now the long address
12	exercise	(uint _c)	Function to physically exercise the option contract	Calls: payment currency (total purchase) delivered to the short, and then asset amount released from escrow delivered to the long. Puts: asset amount delivered to the short, and then the total purchase crypto released from escrow delivered to the long.
13	cashClose	(uint _c, bool cashBack)	Function to exercise the option contract but by which liquidity on a linked AMM is used to deliver the crypto to each party	Calls: a portion of the asset used to swap into the total purchase (payment currency), delivered to the short, and then the reaminng assetAmt either deliverd to the long or swapped again for payment currency and that delivered to the long. Puts: a portion of the total purchase used

				to buy the Asset amount and delivered to the short, and the remainig payment currency delivered out to the long
14	returnExpired	(uint[] memory _calls)	Function for the Short position to have their underlying escrow crypto returned to them after an option has expired without being exercised	Asset / total purchase withdrawn from escrow and the struct(s) updated to show "exercised"
15	rollExpired	(uint[] memory _calls, uint _assetAmount, uint _minimumPurchase, uint _newStrike, uint _newPrice, uint _newExpiry)	Function for the Short position to use their underlying escrow crypto to create a new option "roll" it forward, assuming the option(s) have expired without being exercised	All options rolled should evidence as 'exercised' and one new option generated with the input data fields.
16	transferAndSwap **	(uint _c, address payable newOwner, address[] memory path, bool cashBack)	Function to transfer as the Long position to another address, and specifically to leverage the AnySwap.sol contracts to perform multi-path flash swaps to effectively cashClose when there is not a direct pairing for the two crypto currencies.	Should feel like a cash close, but using a multi-path swap instead of direct pair swap

*** This is the only function that utilizes two smart contracts (ie HedgeyCalls.sol interacting with HedgyAnySwap.sol)*

Test Accounts

- Account A: 0x0C4FAb8d9DBE774708EeC313bf0295278E307bcD
- Account B: 0xde06fEee4c4e3A0B9eEA4f0Ed7a9b21F80a65C58
- Account C: 0xe31D847B47465cC2745319dAc9E0c6ac711cA10b
- Account D: 0x040B6bD961eEd76667D7b4F2a6615657C6b9a303

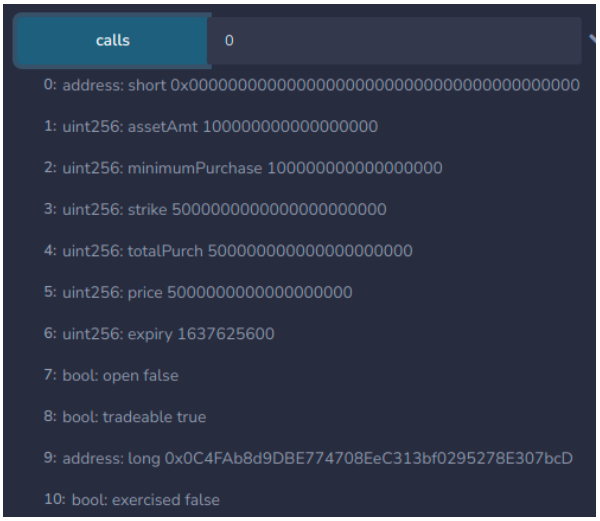
Test Currencies

Rinkeby WETH: 0xc778417E063141139Fce010982780140Aa0cD5Ab
Rinkeby DAI: 0x5592ec0cfb4dbc12d3ab100b257153436a1f0fea
Rinkeby BAT: 0xbf7a7169562078c96f0ec1a8afd6ae50f12e5a99

HedgeyCalls.sol Testing

Rinkeby Address: 0xC85A80a6B989b53a44c7672B96Cf9399b2Eb7BDD
Asset: WETH
Payment Currency: DAI

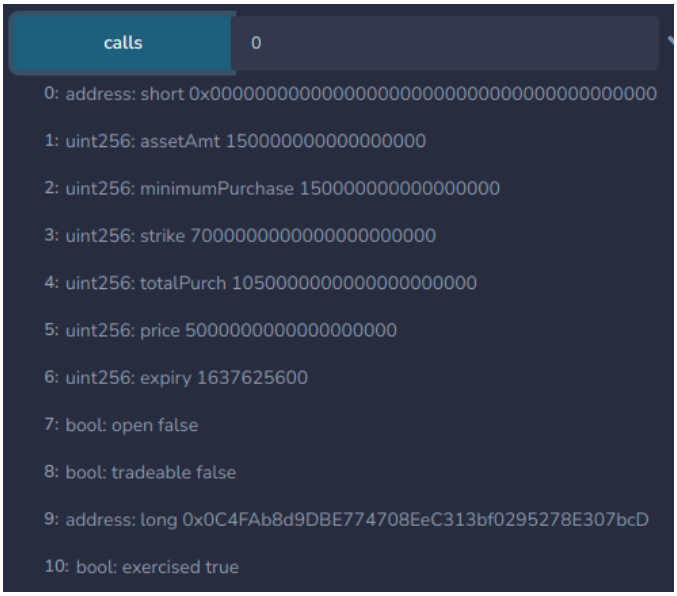
1. **2Account A creates a newBid**
   Parameters:
   Asset Amount: .01
   Strike: 5,000
   Price: 5
   Expiry: 1637625600
   Transaction Hash:
   https://rinkeby.etherscan.io/tx/0x457e4faffa2ca05b60ab71561ca9ecc91c1f470ab58caab554852947b04a627d
   Screenshot:



   Result: SUCCESS

2. **Account A Cancels newBid**
   Parameters:
   Call Index (_c): 0
   Transaction Hash:
   https://rinkeby.etherscan.io/tx/0x30f2849fc7d7ee8df9b5426edfe281c343fd7d94fcf997cafd2a1a7af72da91b

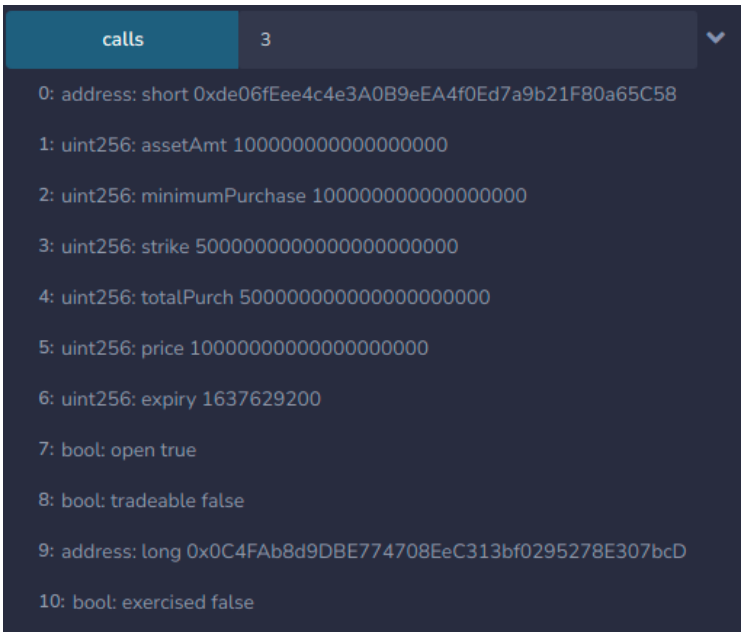
Screenshot:



Result: SUCCESS

3. **Account A sells an existing option to a new bid owned by Account C**
    a. *(First Account A (long) & B (short) have an open option index 3, with following params:*

        AssetAmount: 0.1
        Strike: 5,000
        Price: 10 dai
        Expiry: 1637629200



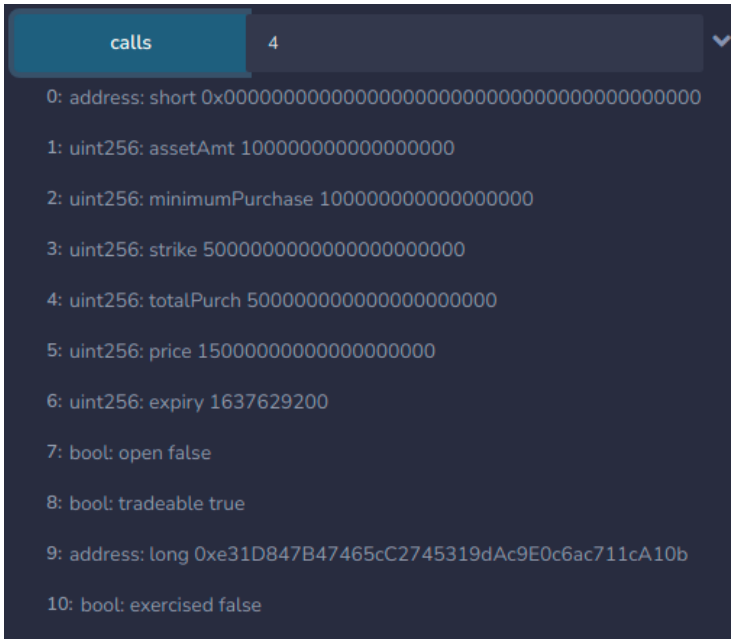
    b. **Account C creates a newBid**

Parameters:

AssetAmount: 0.1
Strike: 5,000
Price: 15 dai
Expiry: 1637629200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x20df7a20924498537f7352936f7a1c97efac34312d85fe90d2d05cb0783879aa

Screenshot:



c. **Account A calls the function to sell its option to account C**

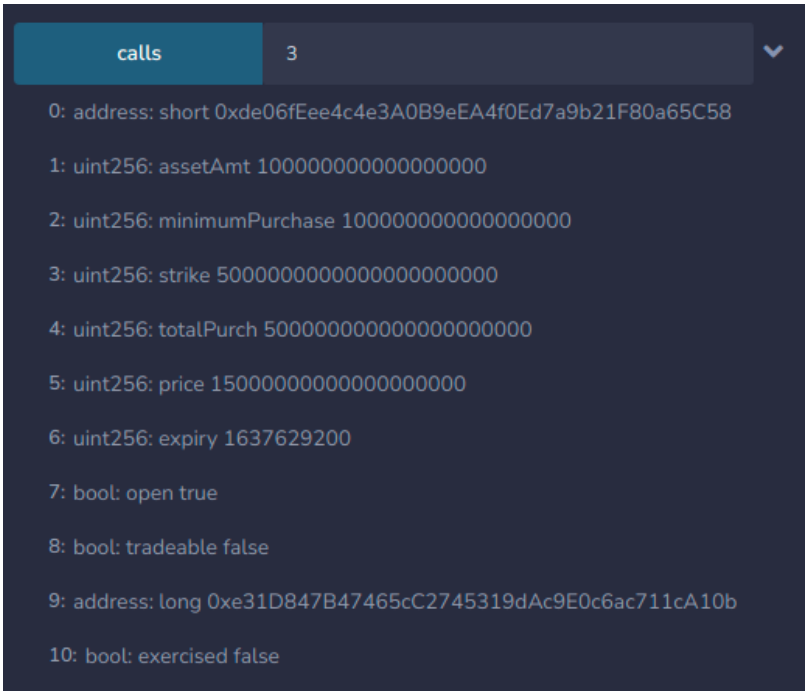
Parameters:

openCall index (_c): 3
newBid index (_d): 4
Price: 15

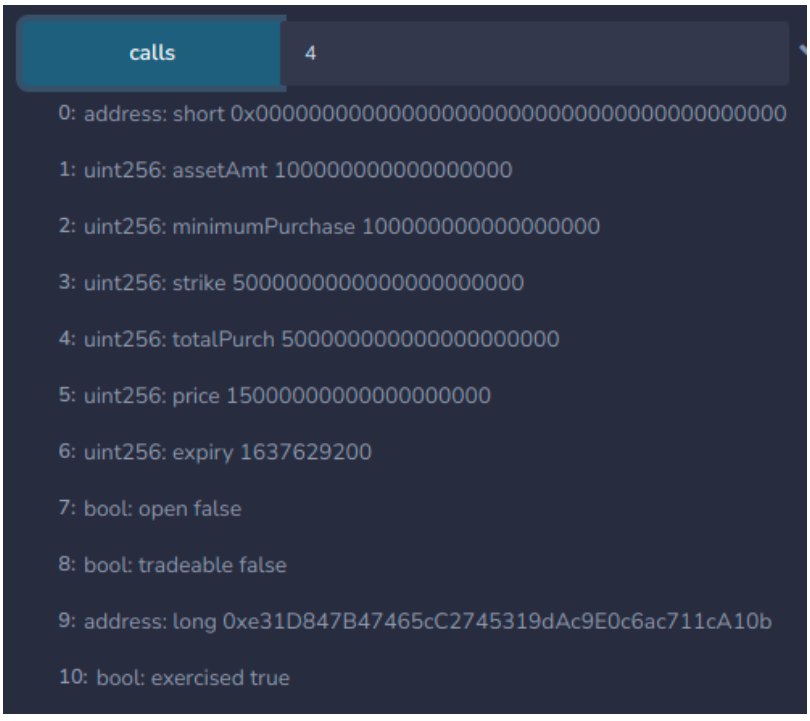
Transaction Hash:
https://rinkeby.etherscan.io/tx/0x17d5af95aceaa9eafdcdc733ed243da53341fc39faf01315a1897a0102954659

Screenshots:

Open option index 3 (with now account C as the long)



Closed option index 4



Result: SUCCESS

4. **Account B sells an option to Account A**
    a. Account A creates a new Bid

Parameters:

AssetAmount: 0.1
Strike: 5,000
Price: 10 dai
Expiry: 1637629200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x6b5301893ed2bb6b8c3bd64c93c768b365bf79f333aeedb2f09b4aa766969641

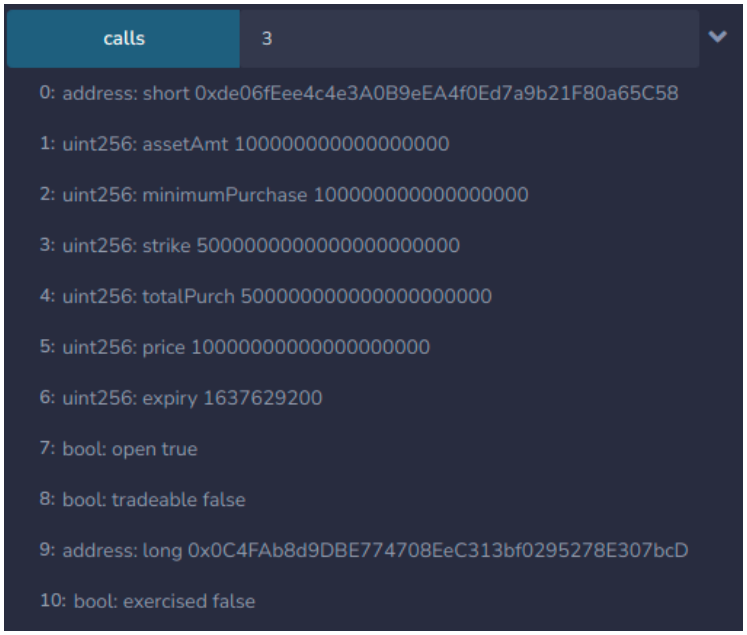
b. **Account B sells the option**

Parameters:

Index (_c): 3
AssetAmount: 0.1
Strike: 5,000
Price: 10
Expiry: 1637629200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xcb40220e3debafa16ea816a5c9dfa4cf0dacc2c43ba7ff0791b7b663f707f759

Screenshot:



Result: SUCCESS

5. **Account A changes a newBid and a newAsk**
   a. **Account A changes a newBid**
      i. **Account A creates a newBid (using index 0)**

Parameters:

Asset Amount: .01

Strike: 5,000
Price: 5
Expiry: 1637625600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x457e4faffa2ca05b60ab71561ca9ecc91c1f470ab58caab554852947b04a627d

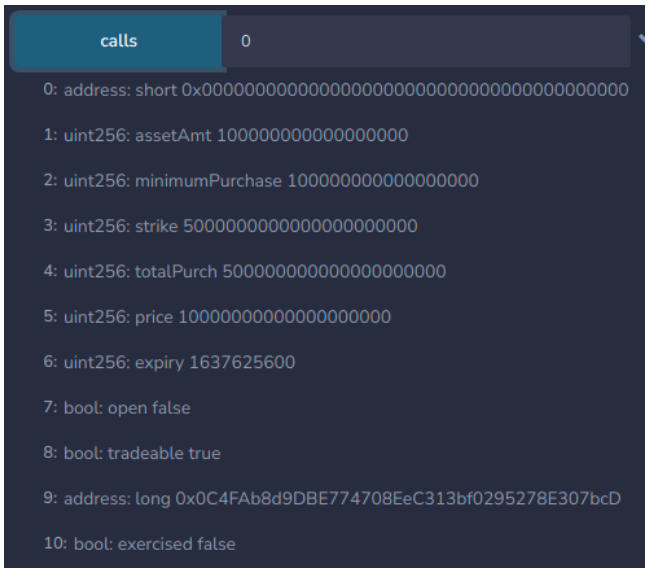
ii. **Account A increase the price by 5 DAI (deliver 5 DAI into escrow)**

Parameters:

index(_c): 0
AssetAmount: 0.1
Minimum: 0.01
Strike: 5,000
Price: 10 dai
Expiry:

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x1bb0e19f7a7d8549cdf53ded3d4b014983e7c92a43bc3e29b29511807a663370

Screenshot:



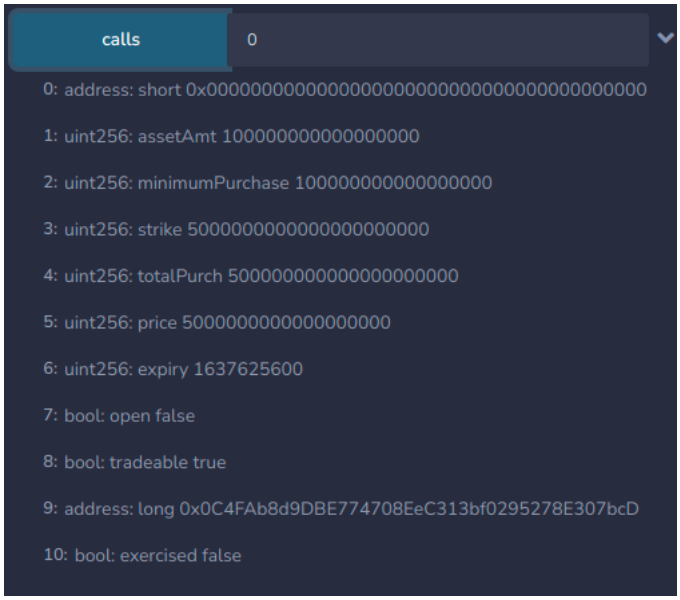
iii. **Account A decreases the price by 5 DAI**

Parameters:

index(_c): 0
AssetAmount: 0.1
Minimum: 0.01
Strike: 5,000
Price: 5 dai
Expiry: 1637625600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x85d59067055f5daca3aef92699a0fae0476862777cf2ae7dd124e447b1a08552

Screenshot:



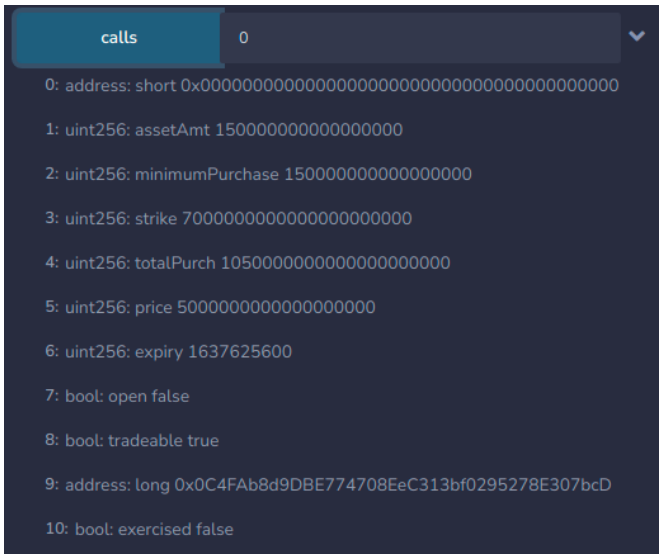
iv. **Account A changes the asset amount, strike**

Parameters:

Index (_c): 0
AssetAmount: 0.15
Minimum: 0.7
Strike: 7,000
Price: 5 dai
Expiry: 1637625600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x2d025edbf3e99e9f53ead6a0b78d254528ec01f3db876cc6dbd7c3125d6d1a29

Screenshot:



b. **Account A changes a newAsk**

i. **Account A creates a newAsk (using index 1)**

Parameters:

AssetAmount: 0.1
Minimum: 0.01
Strike: 5,000
Price: 10 dai
Expiry: 1637625600

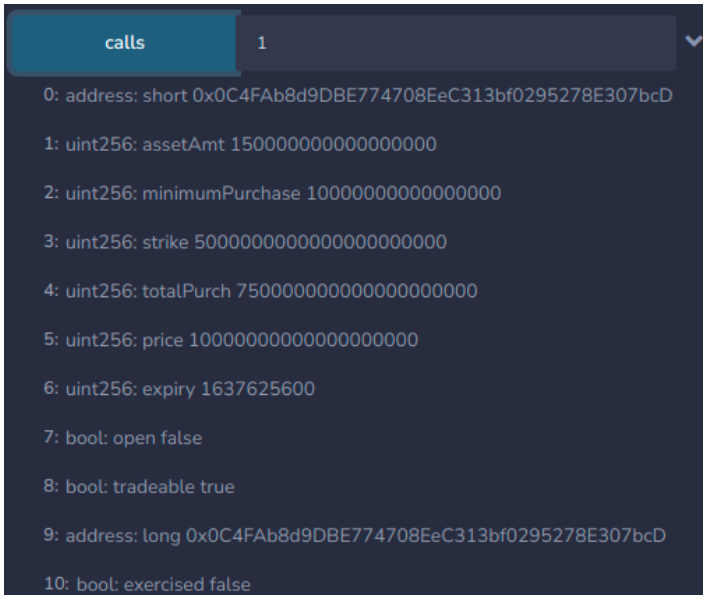
ii. **Account A increases the Asset Amount to .15**

Parameters:

Call index (_c): 1
AssetAmount: 0.15
Minimum: 0.01
Strike: 5,000
Price: 10 dai
Expiry: 1637625600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xd769feeab0e4b21ae393dae2a652cbbedd417f920ae43d4888f9aabda83fecde

Screenshot:



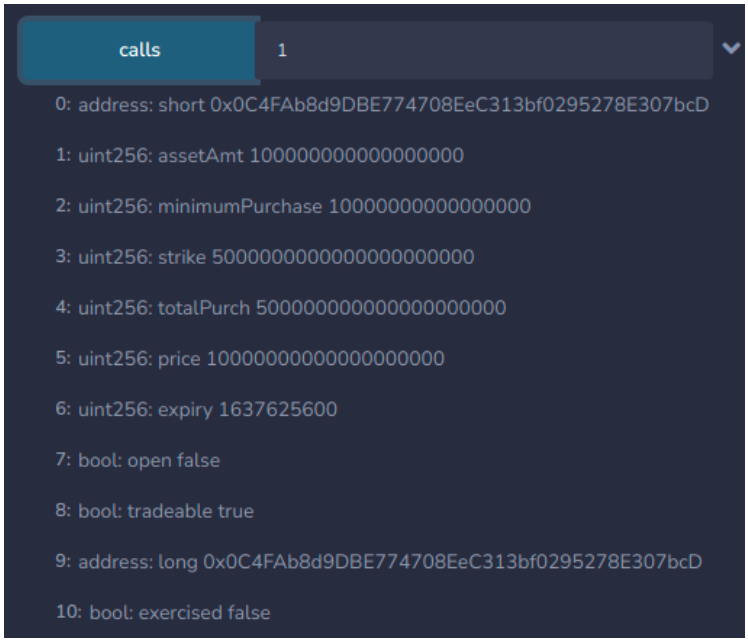
iii. **Account A decreases the Asset Amount**

Parameters:

Call index (_c): 3
AssetAmount: 0.1
Minimum: 0.01
Strike: 5,000
Price: 10 dai
Expiry: 1637614800

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x51aaf56302f6ead5101c9fc0cac655981517f0db61026fa71d744afcee8fd4fe

Screenshot:



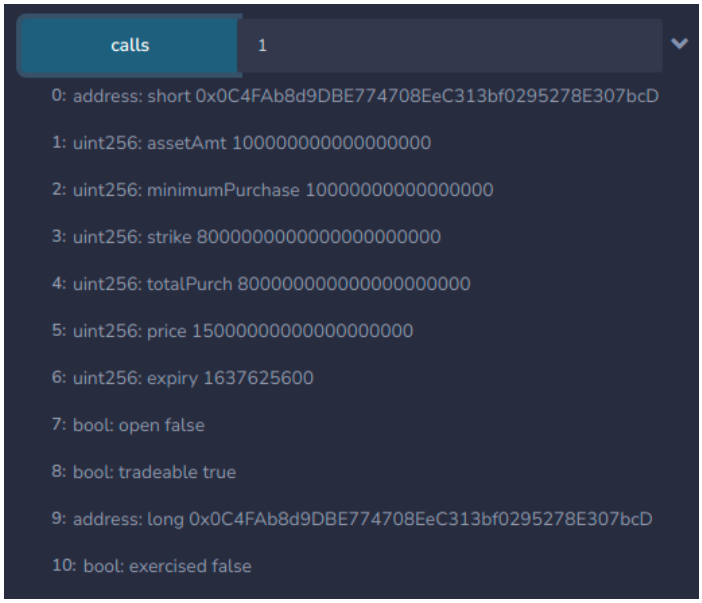
iv. **Account A changes the Minimum, Price, Expiry**

Parameters:

Call index (_c): 1
AssetAmount: 0.1
Minimum: 0.01
Strike: 8,000
Price: 15 dai
Expiry: 1637625600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x66fdc0e936178c323dfca6c450078186aea47e0fa96e8f3621ab48e609fc0512

Screenshot:

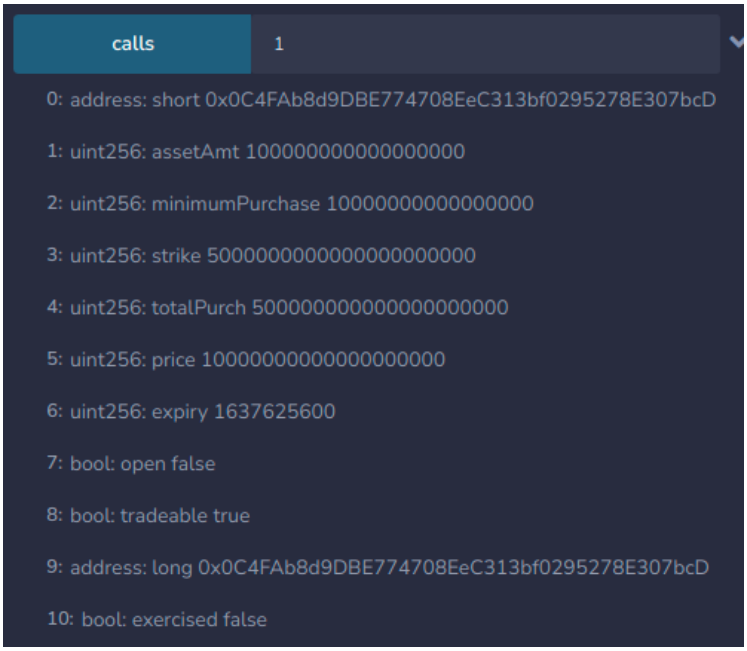


Result: SUCCESS

6. **Account A writes a newAsk**
   Parameters:
   AssetAmount: 0.1
   Minimum: 0.01
   Strike: 5,000
   Price: 10 dai
   Expiry: 1637625600

   Transaction Hash:
   https://rinkeby.etherscan.io/tx/0x9291b7fc0a1d3f8966b482d7c4e29540c30ec32f0ac30498e372c6a387679bda

   Screenshot:



   Result: SUCCESS

7. Account A cancels a newAsk
   Parameters: call index (_c): 1
   Transaction Hash:
   https://rinkeby.etherscan.io/tx/0x16e7bbcba4b95b537aef2dec988f7c4ba5532bdce06f5903b9fc81f6686fc898

Screenshot:

Result: SUCCESS

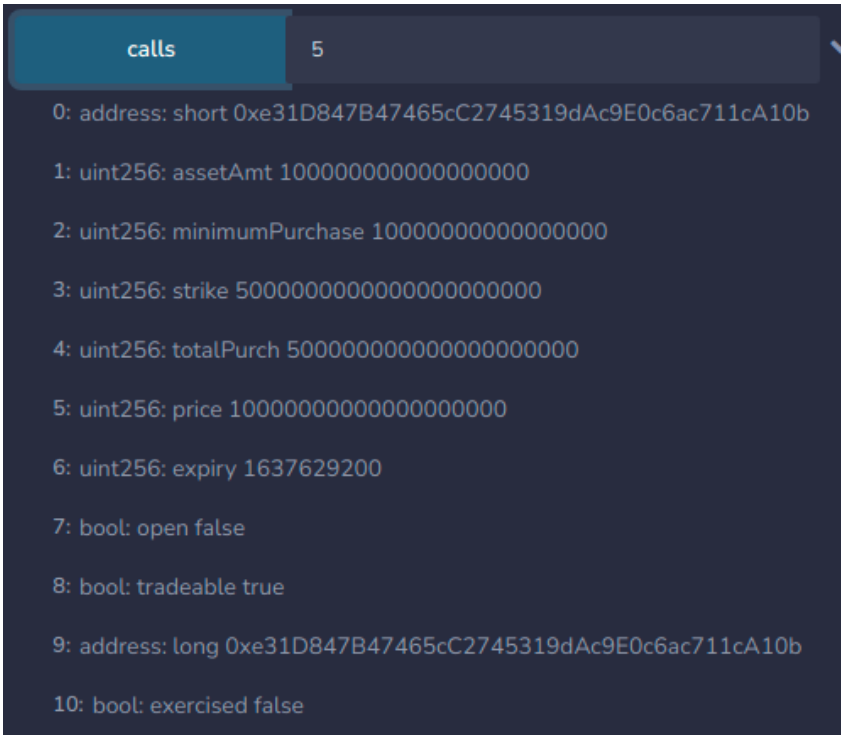
8. A**ccount B & A & D purchase options from account C**
    a. **Account C creates a newAsk**

Parameters:

AssetAmount: 0.1
Minimum: 0.01
Strike: 5,000
Price: 10 dai
Expiry: 1637629200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xacb93255d1c3a401a93ac9c7cc6543e07497b610e89ae0df4416cd34cd867c79

Screenshot:



b. Account B purchases 30% of the newAsk

Parameters:

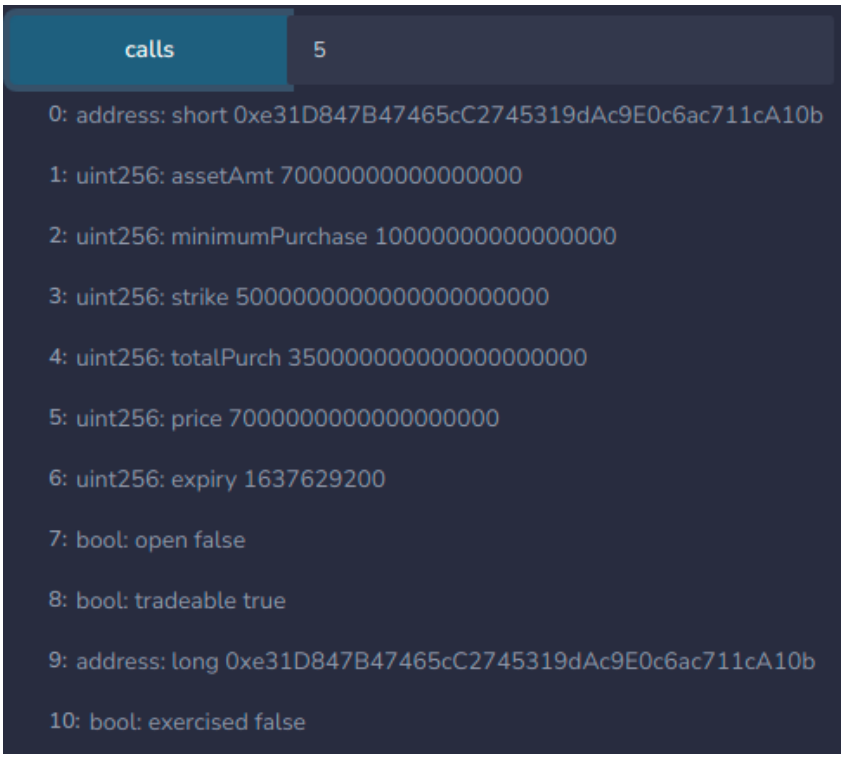
Index (_c): 5
assetAmount: .03
Strike: 5,000
Price: 3
Expiry: 1637629200

Transaction Hash:

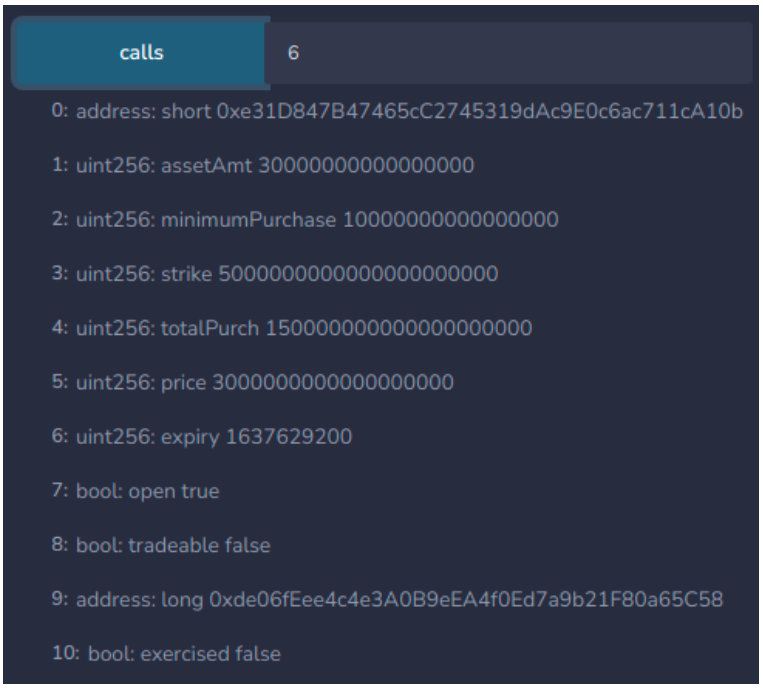
https://rinkeby.etherscan.io/tx/0x512a05a441c4a90ddf2469264f64520644b888de8ca73b86662d2ba208281d21

Screenshot:

Remaining newAsk with the updated asset amount, price, total purchase



New call created with the long and short positions matching what account B purchased:



c. **Account A purchases the same 30% of the newAsk**

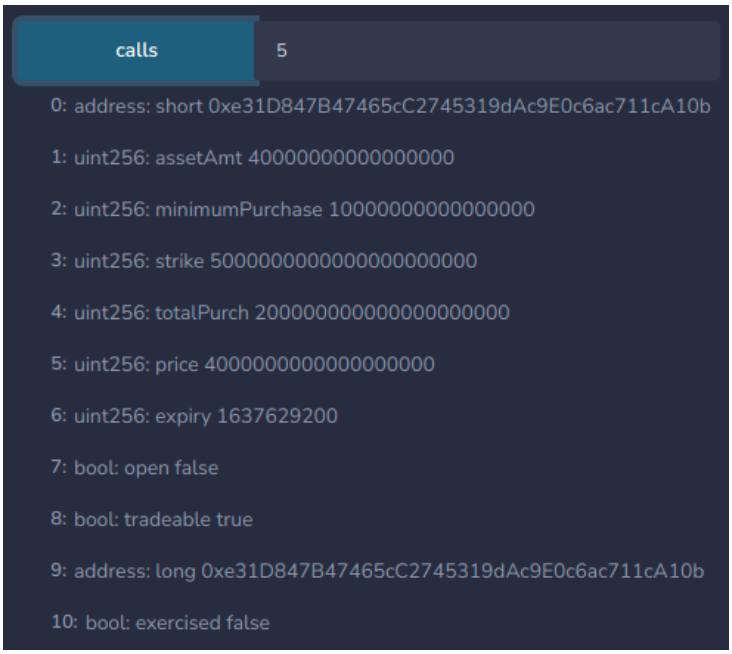
Parameters:

Index (_c): 5
assetAmount: .03
Strike: 5,000
Price: 3
Expiry: 1637629200

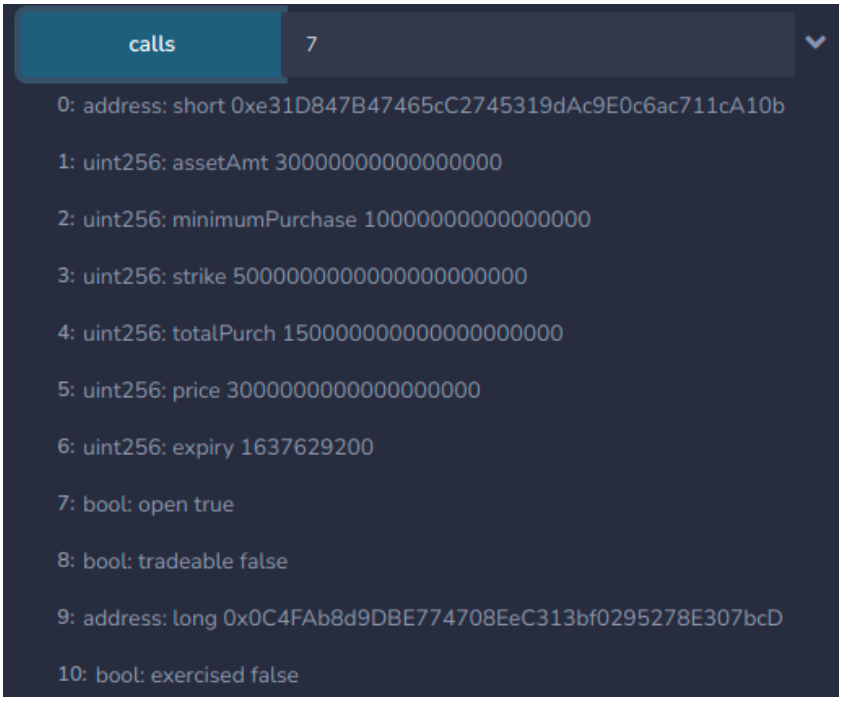
Transaction Hash:
https://rinkeby.etherscan.io/tx/0x790882e3605905739f015de9b67e93ba8a9efe37707c5bf7c354cab5f247770c

Screenshot:

Updated newAsk with asset amount remaining, price and total purchase re calculated



New open call option with all parameters calculated for index 7



d. **Account C purchases 40% of the newAsk**

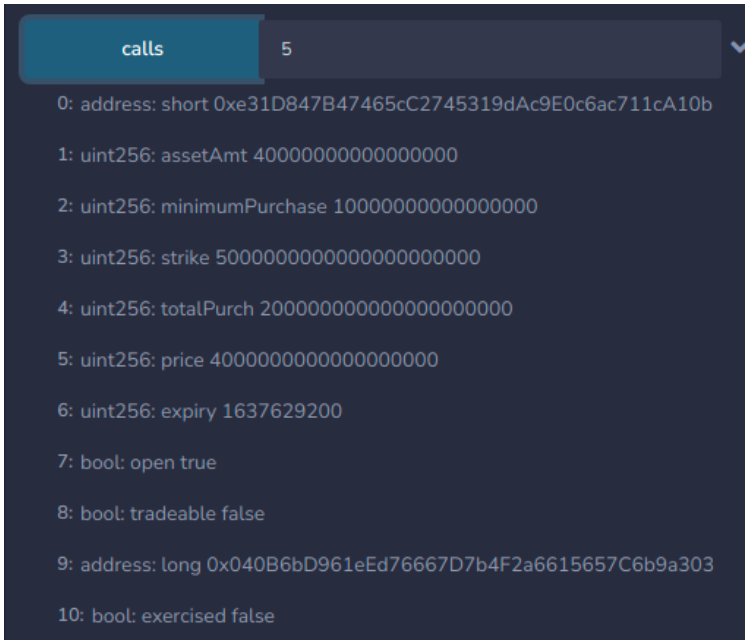
Parameters:

Index (_c): 5
assetAmount: .04
Strike: 5,000
Price: 4
Expiry: 1637629200

Transaction Hash:

https://rinkeby.etherscan.io/tx/0xff2e37ba0c667924591854cfb374ace670d110bf4c4124e3f7e8d5f6cca250e8

Screenshot:



Result: SUCCESS

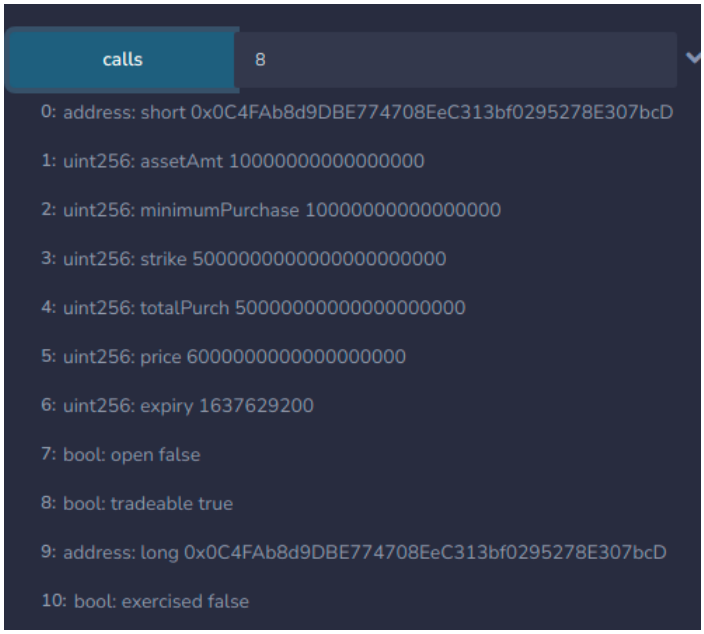
9. **Account A writes a newAsk and account C buys from account A**
    a. Using Index 5 from previous section sold by C
    b. **Account A writes a new Ask**

Parameters:

assetAmount: .04
Minimum amount: 0.01
Strike: 5,000
Price: 6
Expiry: 1637629200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x2a099b393210869add6f207baa664f0245135fc1e8e13b2eba9683534125d258

Screenshot:



c. **Account C sells call option to A new ask index 7**

Parameters:

openCall index (_c): 5

newAsk index (_d): 8

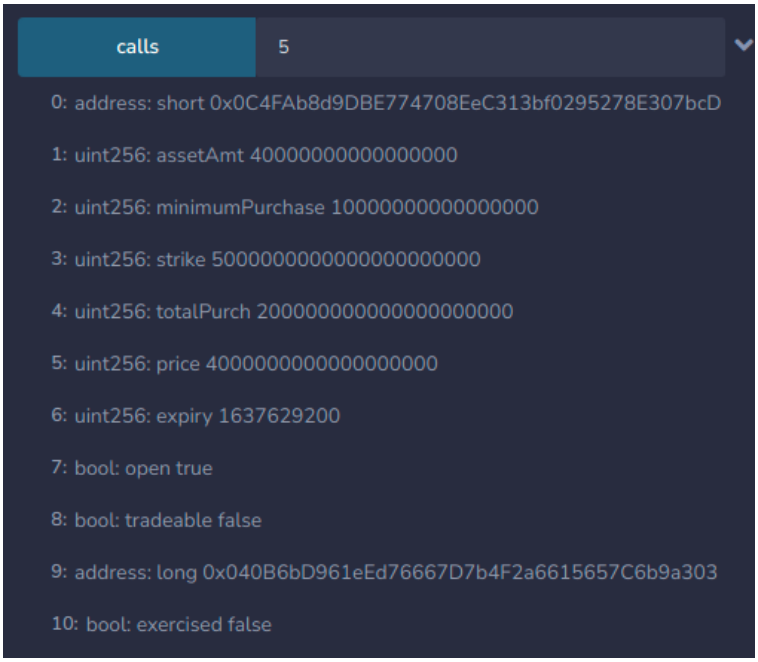
Price: 6 dai

Transaction Hash:

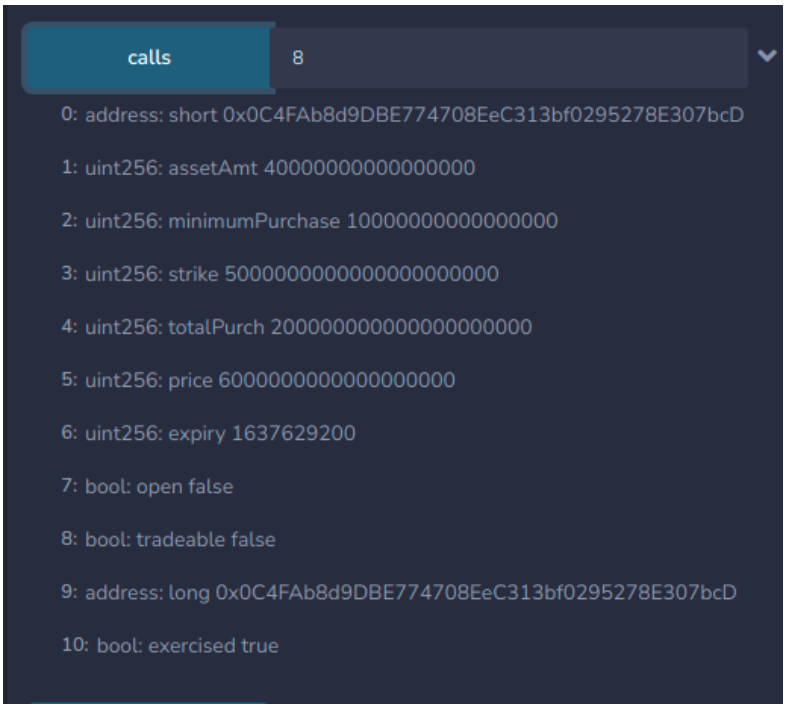
https://rinkeby.etherscan.io/tx/0xbeab4f593164c13707d8887e7cf9a0c7204a78b38a917cc2ffc9a208085ae524

Screenshot:

Call update showing that Account A is now the short



And the newAsk is closed out by being exercised == true



Result: SUCCESS

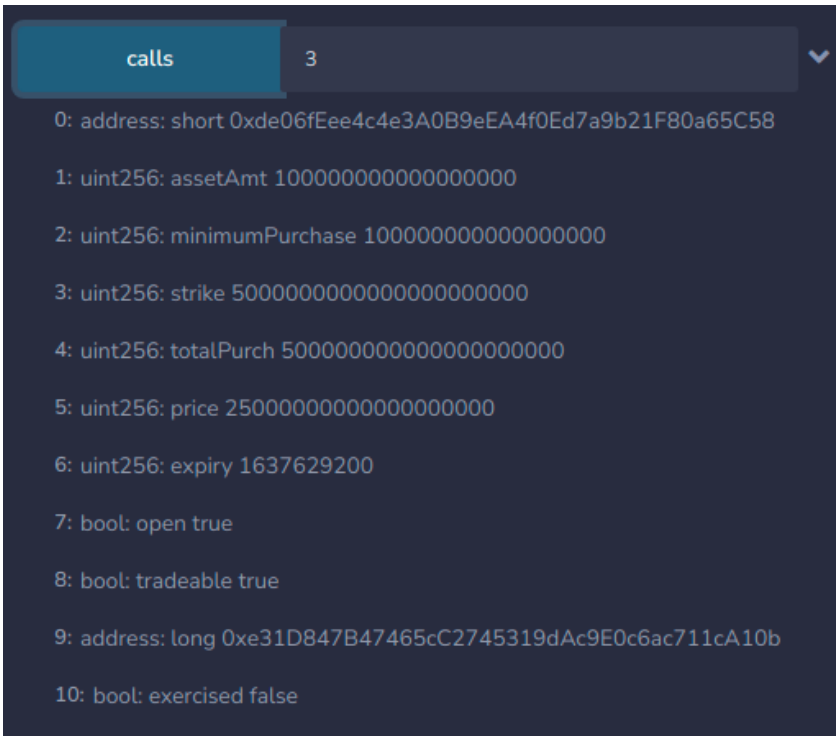
10. Account C puts call option index 3 up for sale

Parameters:

index (_c): 4
Price: 25 dai
Tradeable: true

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x0c3ca5161dcfdfa6557fb7cbc4b80c45a5e48fc90ef19c8800ec489143a7597d

Screenshot:



Result: SUCCESS

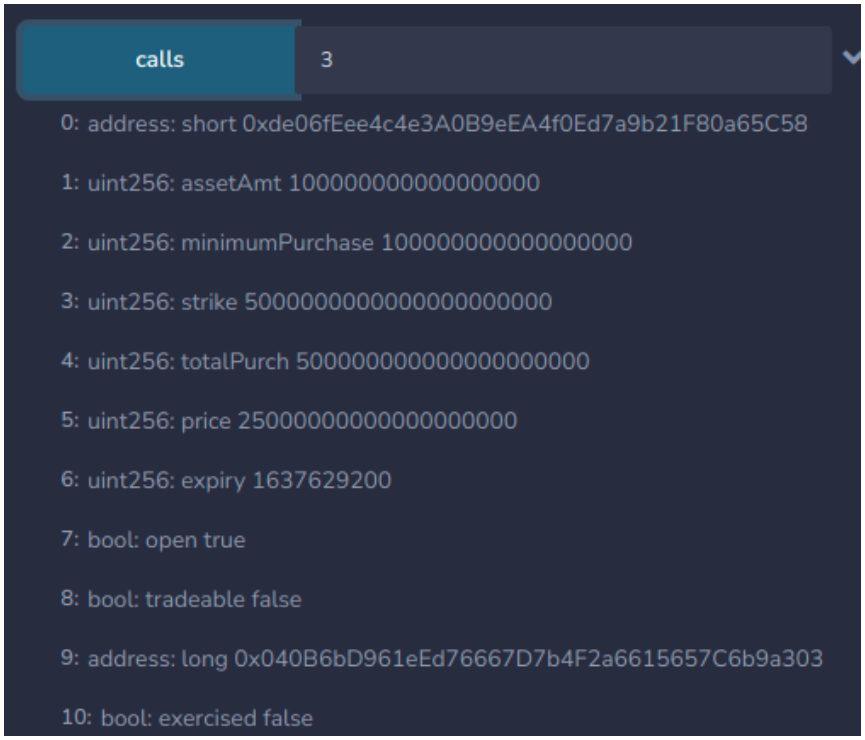
11. **Account D buys an open option from Account C**

Parameters:

index (_c): 3
assetAmt: .1
Strike: 5,000
Price: 25 dai
Expiry: 1637629200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xc1b571bca8e5fce8e292c1a5c1adf17b733452a9d409301a930ec651de408904

Screenshot:



Result: SUCCESS

12. **Account D exercises an option (index 5)**

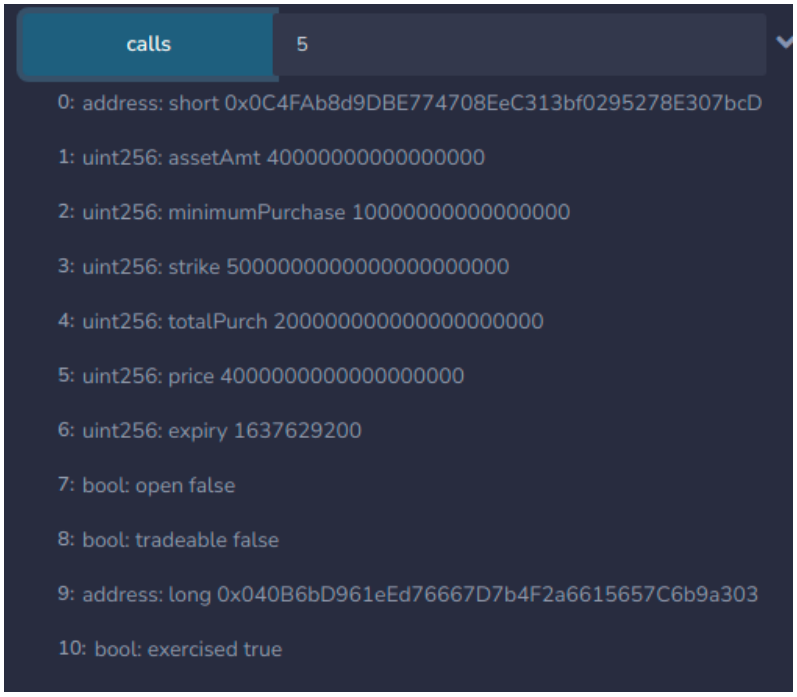
Parameters:

index (_c): 5

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x39c01634be318ce5daa46f8fc65c07534c776106e427db4e402375f1248f217d

Screenshot:



Result: SUCCESS

13. **Account D cashClose exercises an option**

Parameters:

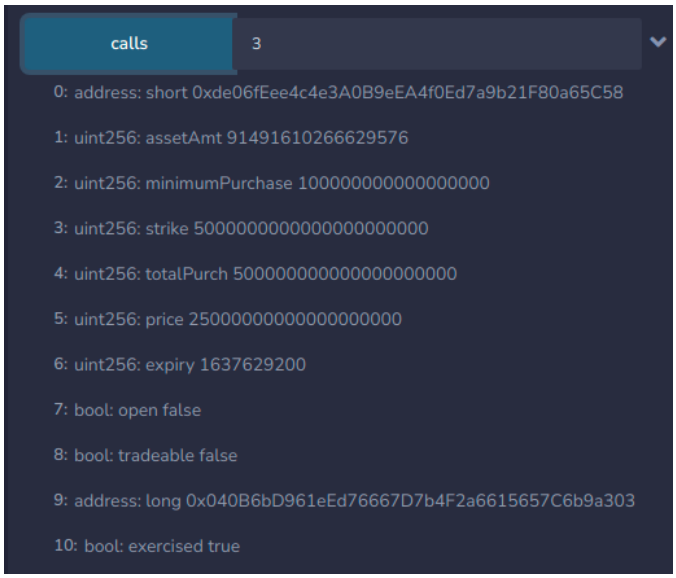
index (_c): 3

cashBack: true

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x685c2530c4b802e57a2a96de0a00748cde61b756a977502fd6080f138a35f854

Screenshot:



Result: SUCCESS

14. **Account C returns 2 expired calls**

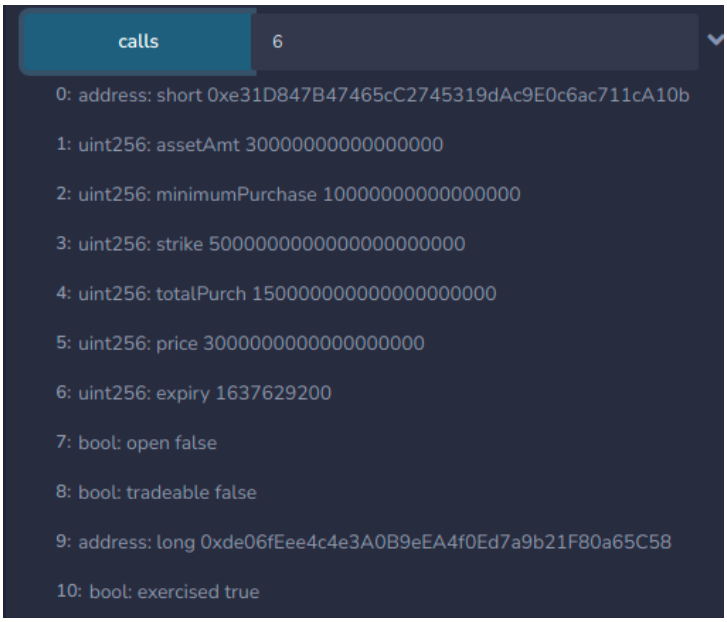
Parameters:

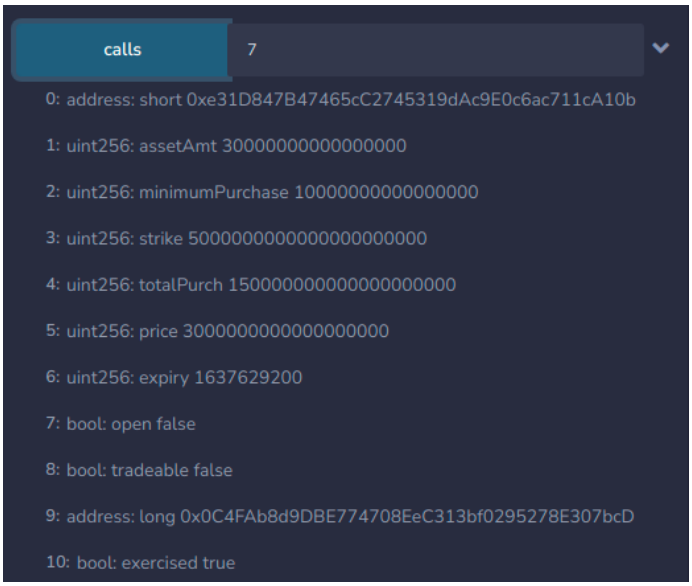
index (_c): [“6”,”7”]

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x36644fb47be1c5c302f8bd6951025dd8149051923a90b1488ea2a2ba6559046f

Screenshot:





Result: SUCCESS

15. **Account C rolls multiple expired calls (indexes 9, 10, 11, 12, 13) into one call**

Parameters:

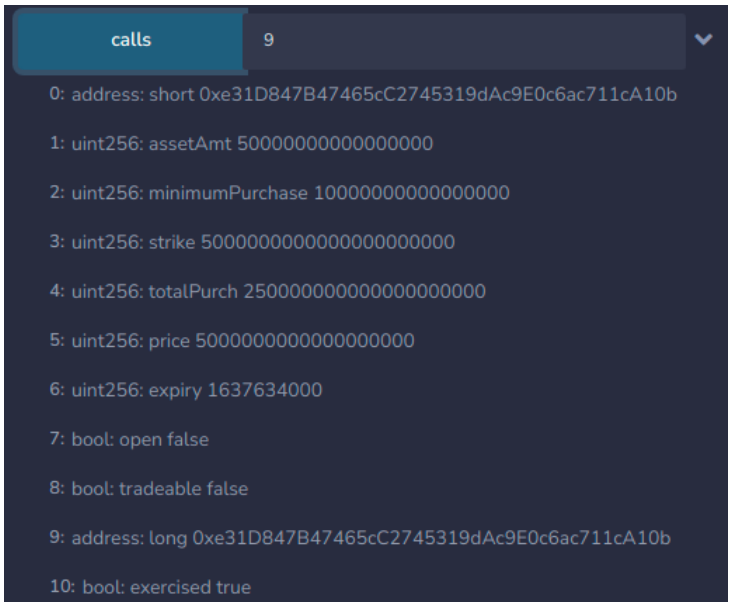
index (_c): ["9","10","11","12","13"]

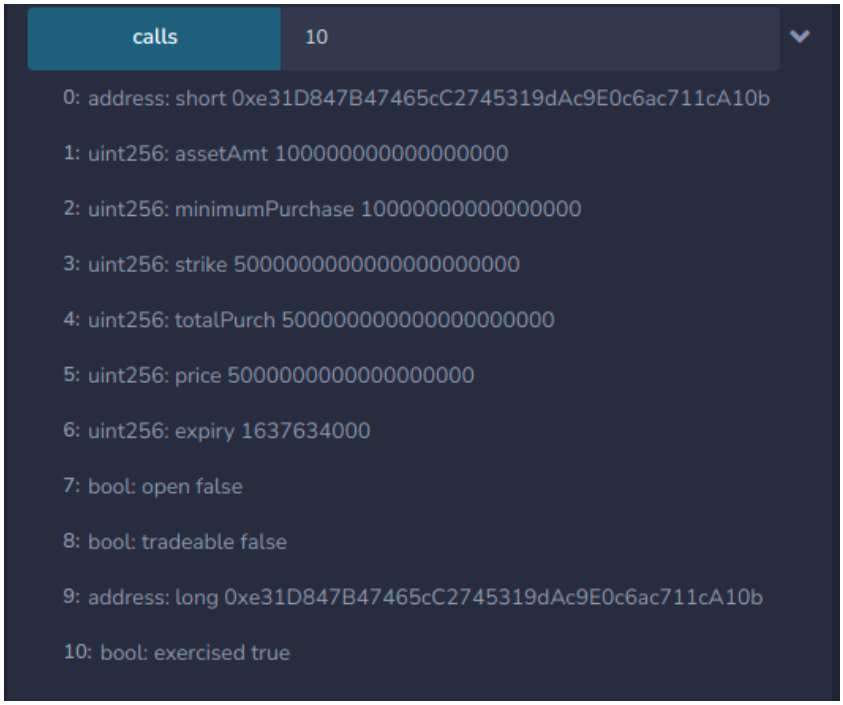
assetAmount: 0.5

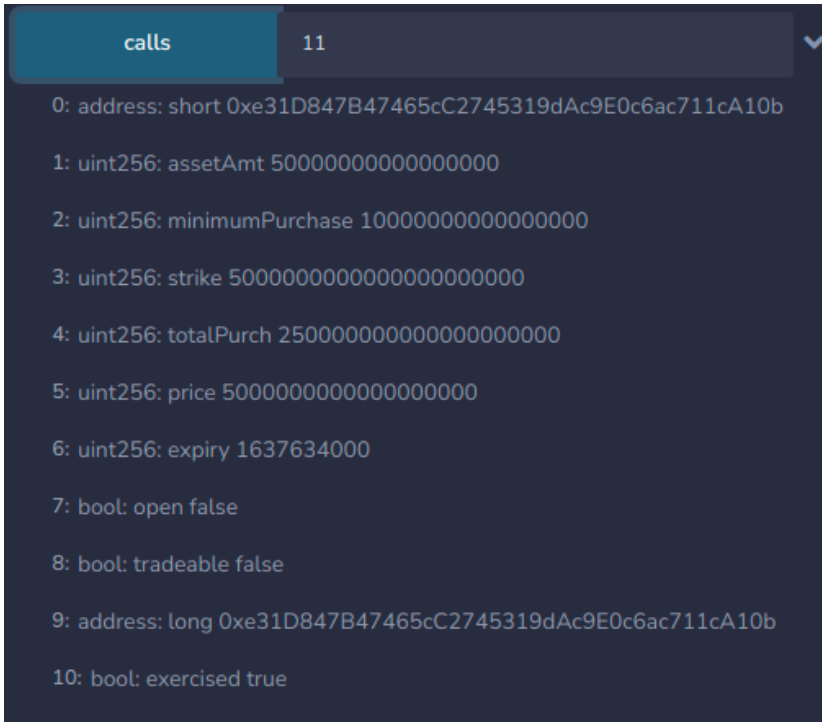
mininumPurchase: .02
newStrike: 10,000
newPrice: 77 dai
newExpiry: 1637634600
Tx value: .05 eth

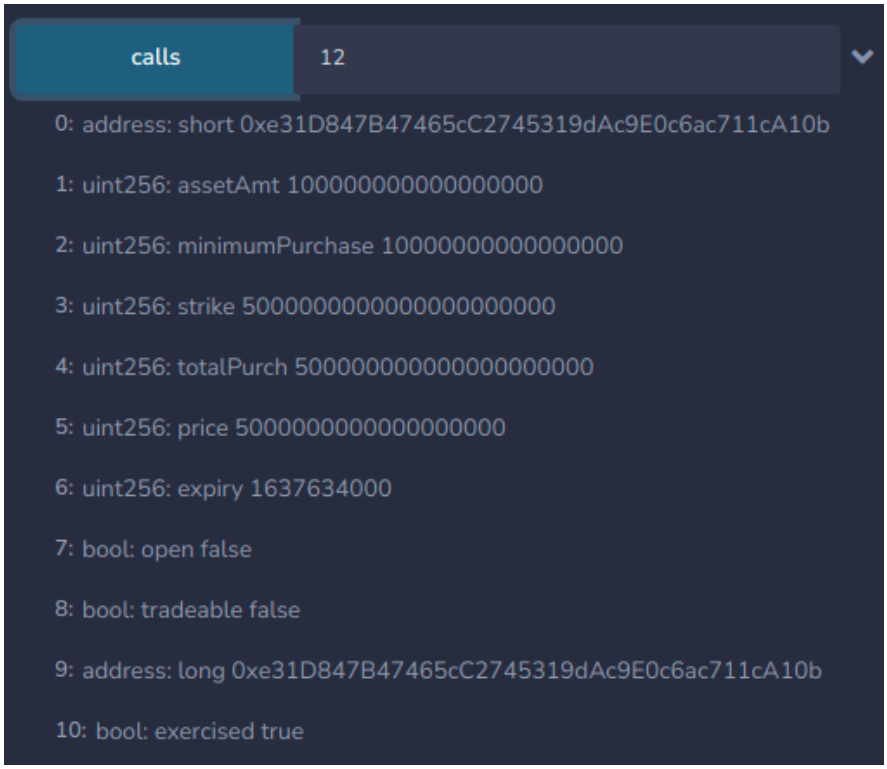
Transaction Hash:
https://rinkeby.etherscan.io/tx/0xe70a52c890668808a1a33909395bdc405fc95db45cb6bedbe37fe15118112f6e

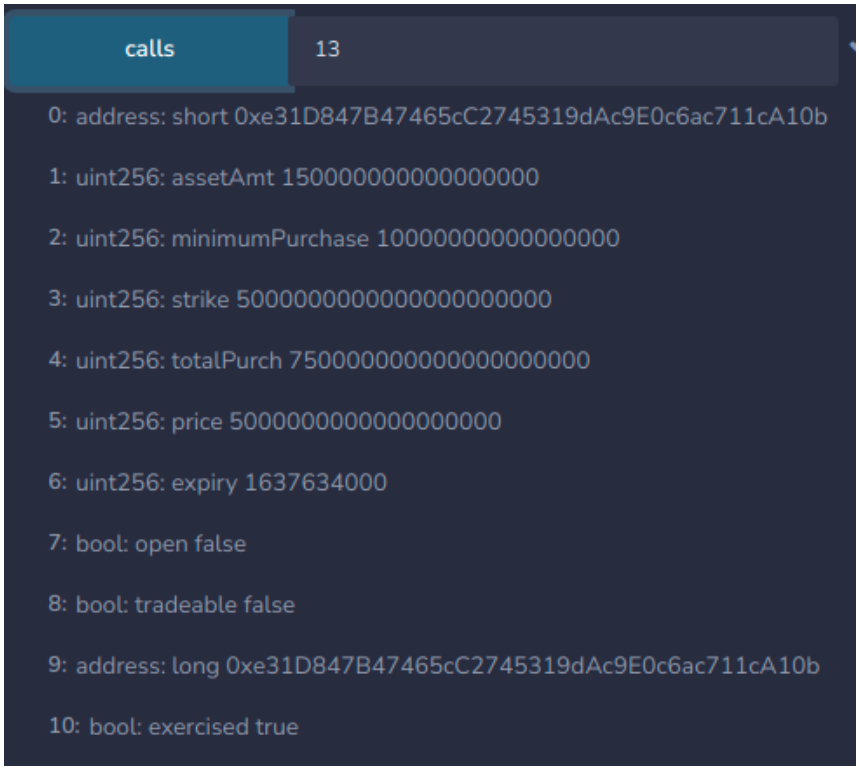
Screenshot:



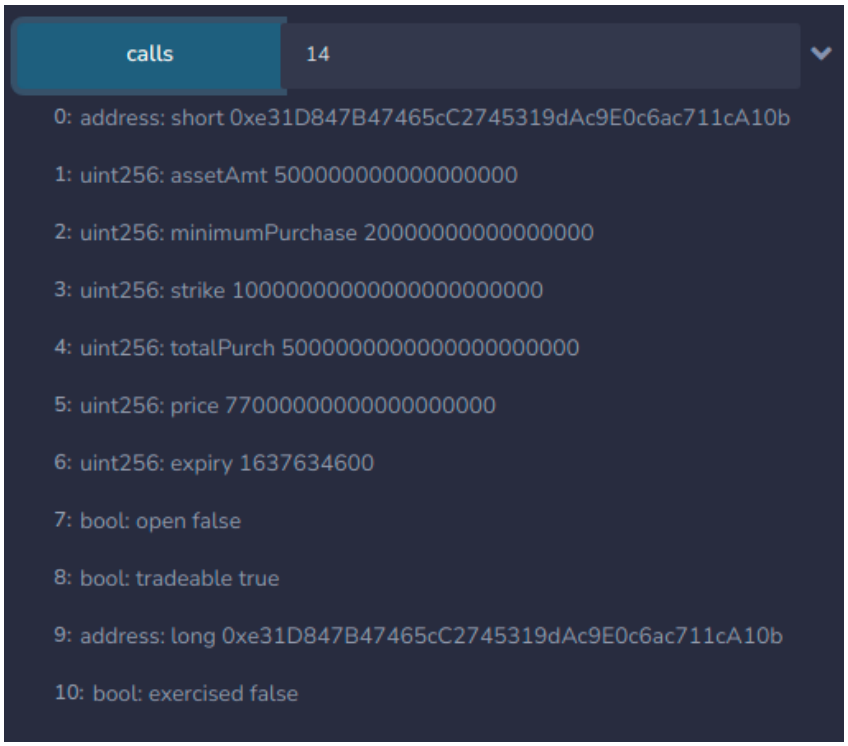



calls
11
0: address: short 0xe31D847B47465cC2745319dAc9E0c6ac711cA10b
1: uint256: assetAmt 50000000000000000
2: uint256: minimumPurchase 10000000000000000
3: uint256: strike 500000000000000000000
4: uint256: totalPurch 25000000000000000000
5: uint256: price 5000000000000000000
6: uint256: expiry 1637634000
7: bool: open false
8: bool: tradeable false
9: address: long 0xe31D847B47465cC2745319dAc9E0c6ac711cA10b
10: bool: exercised true


calls
12
0: address: short 0xe31D847B47465cC2745319dAc9E0c6ac711cA10b
1: uint256: assetAmt 100000000000000000
2: uint256: minimumPurchase 10000000000000000
3: uint256: strike 5000000000000000000000
4: uint256: totalPurch 50000000000000000000
5: uint256: price 5000000000000000000
6: uint256: expiry 1637634000
7: bool: open false
8: bool: tradeable false
9: address: long 0xe31D847B47465cC2745319dAc9E0c6ac711cA10b
10: bool: exercised true



And the new call to replace all of them:



Result: SUCCESS

16. **Account A owns a long call and multiswap cash closes an option**

Parameters:

Index (_c): 15

newOwner: 0x049a8A4CBf1f8a1a160E0ee8dE94B1775204eB8B (this is also the deployment of HedgeyAnySwap.sol)

Path: ["0xc778417e063141139fce010982780140aa0cd5ab","0xbf7a7169562078c96f0ec1a8afd6ae50f12e5a99","0x5592ec0cfb4dbc12d3ab100b257153436a1f0fea"] *WETH -> BAT -> DAI*

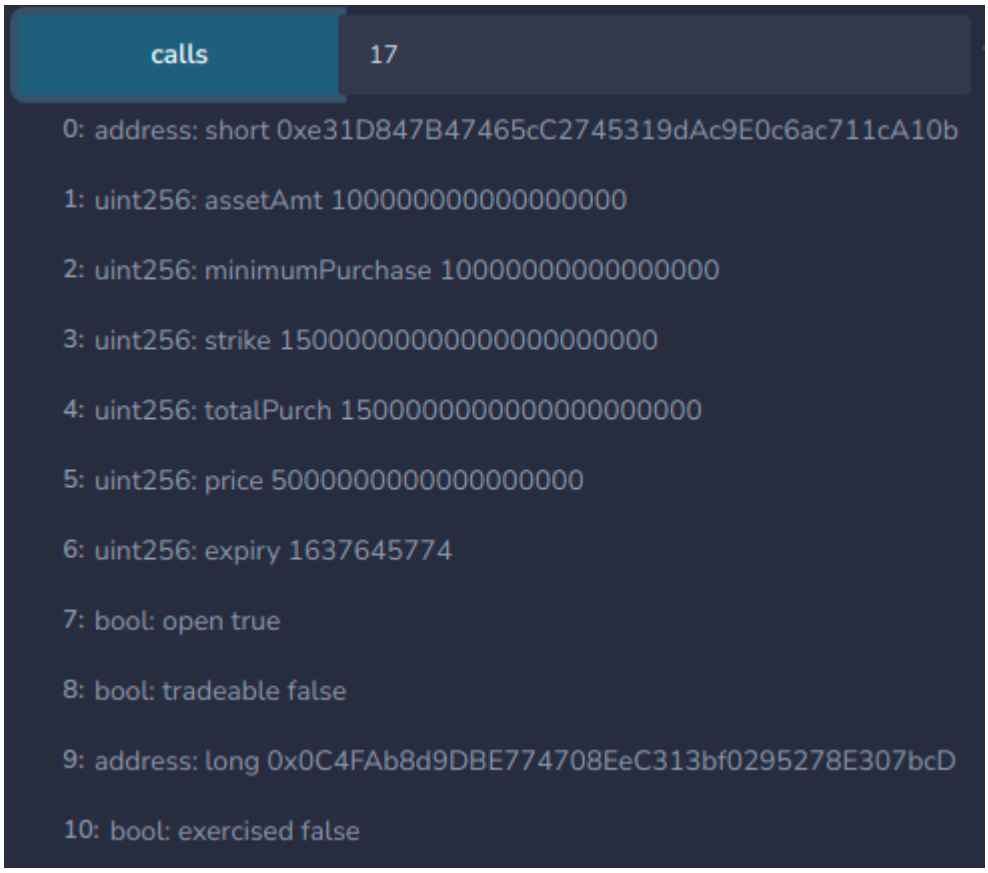
Cashback: true

Transaction Hash:

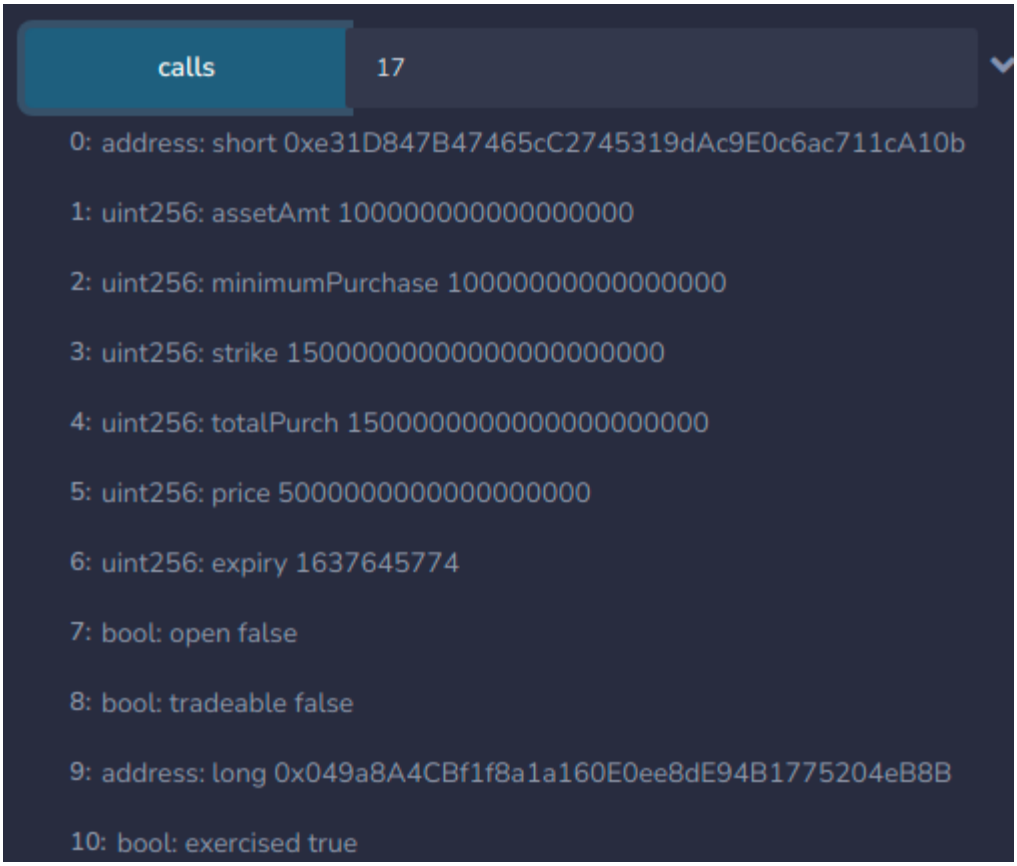
https://rinkeby.etherscan.io/tx/0x69b179f32a221671ab9885cbb5ad50c65079bfc690d7dffb90d0b962a9cbcec7

Screenshot:

Before:



After:

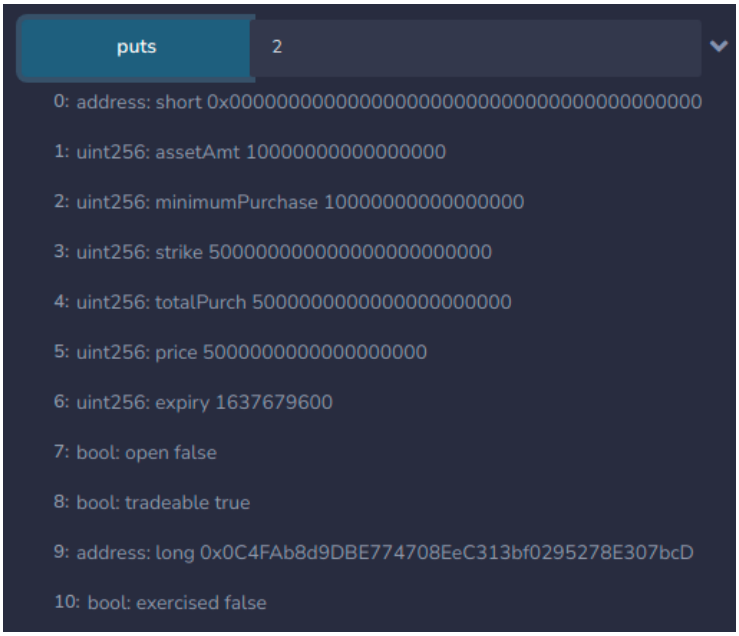


Hedgey PUTS Testing

Rinkeby Address: [0xa9e142138718207BA456AD78b6711142881545F9]
Asset: WETH
Payment Currency: DAI

1. **Account A creates a newBid**
   Parameters:
   Asset Amount: .01
   Strike: 500,000
   Price: 5
   Expiry: 1637679600
   Transaction Hash:
   https://rinkeby.etherscan.io/tx/0xb95ee0e06b1e9b9edffb1e6d34dfd89f87e768cc43d677e4416eaedb7c7c80cd

Screenshot:



Result: SUCCESS

2. **Account A Cancels newBid**

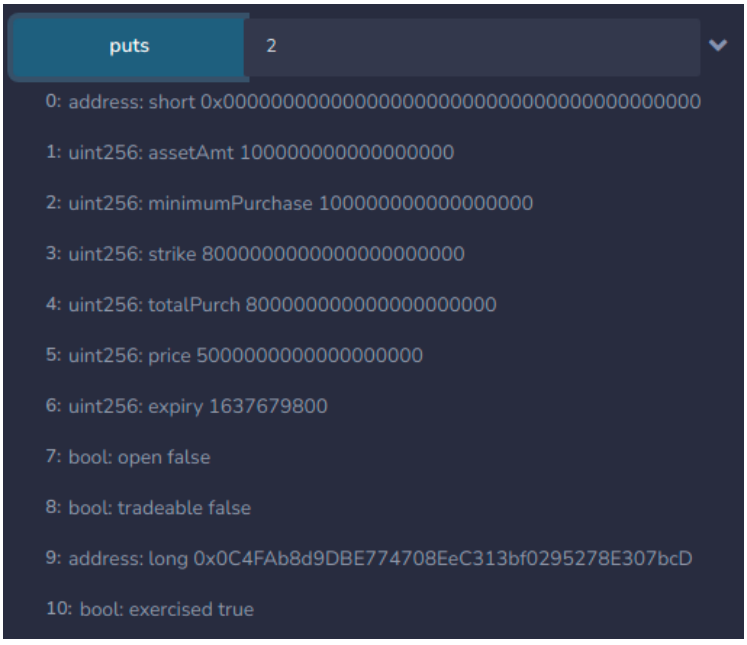
Parameters:

Call Index (_p): 2

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x22cc70c4176d3545824ff4ebf00168183993549c591a0267553be87cc3af2c56

Screenshot:



Result: SUCCESS

3. **Account A sells an existing option to a new bid owned by Account C**
    a. *(First Account A (long) & B (short) have an open option index 8, with following params:*

        AssetAmount: 0.01
        Strike: 500,000
        Price: 10 dai
        Expiry: 1637686800

    b. **Account C creates a newBid**

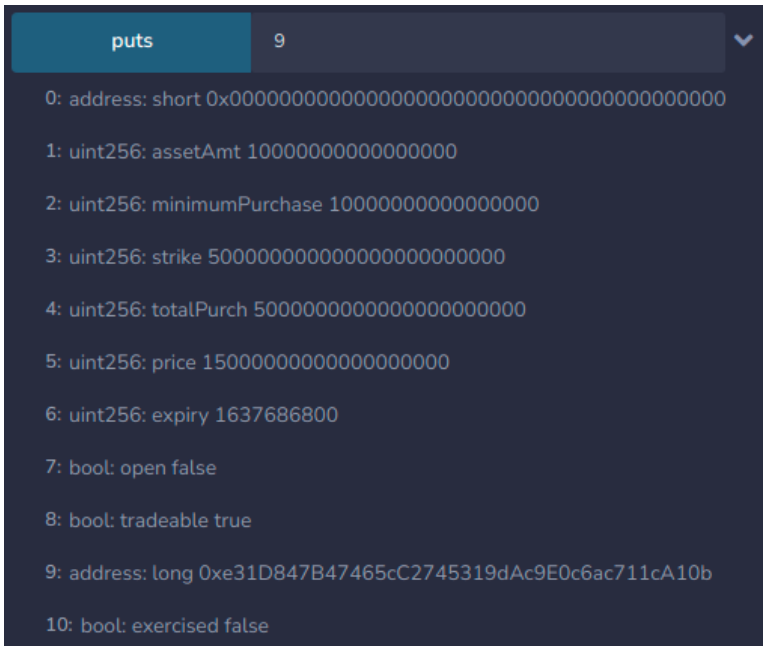
        Parameters:

        AssetAmount: 0.1
        Strike: 5,000
        Price: 20 dai
        Expiry:

        Transaction Hash:

        https://rinkeby.etherscan.io/tx/0x8d796c6770bf0b22cfbebdc70cc63826d64f71307df06bcb32a08eafb84175d0

        Screenshot:



    c. **Account A calls the function to sell its option to account C**

        Parameters:

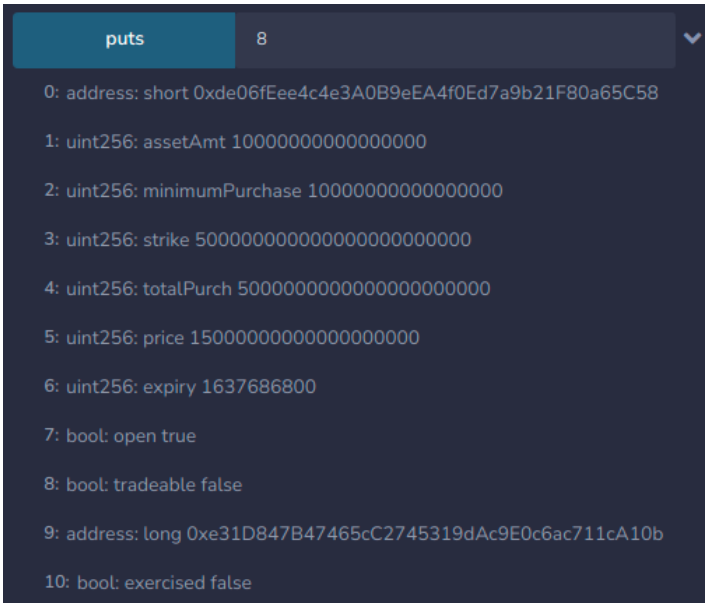
        openCall index (_p): 8
        newBid index (_q): 9
        Price: 15 dai

        Transaction Hash:

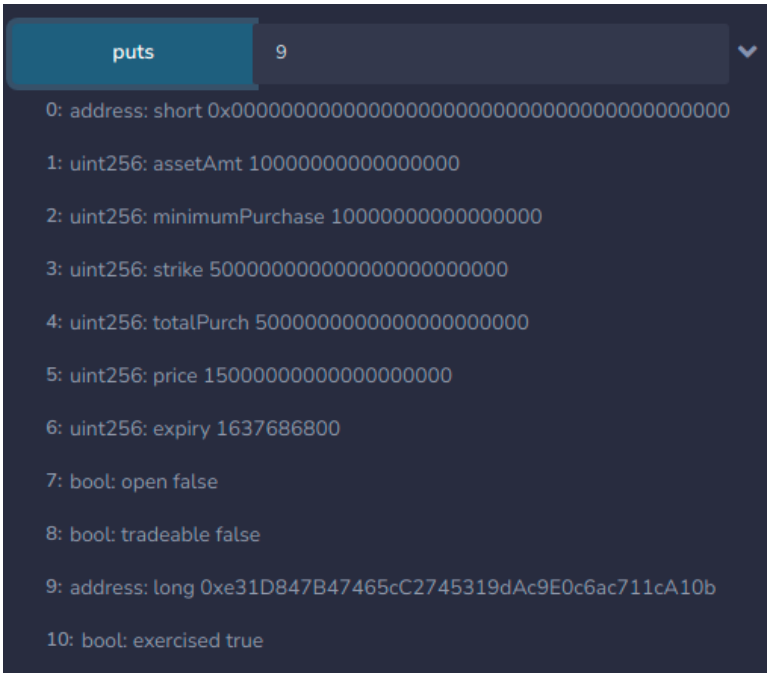
        https://rinkeby.etherscan.io/tx/0xd88bab8e172dd0aca7959a88ac96c4aed629f9e5a31ac58ceac4982327036fa2

        Screenshots:

Open option index (with now account C as the long)



Closed option index



Result: SUCCESS

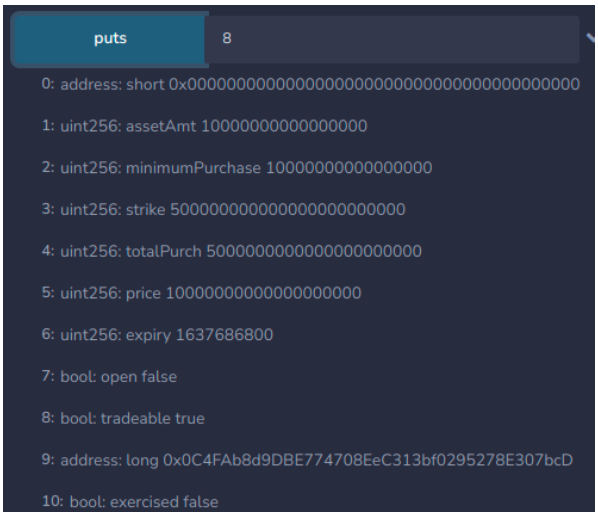
4. **Account B sells an option to Account A**
    a. Account A creates a new Bid

Parameters:

AssetAmount: 0.01
Strike: 500,000

Price: 10 dai
Expiry: 1637686800

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x9a76d930c44e150c93e2009225f372d58877e348751cbd44dfdfdea2705dda7



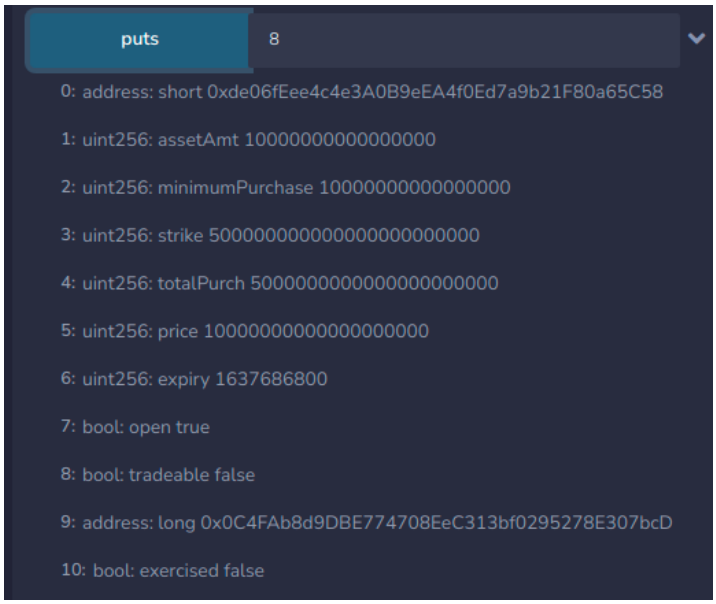
b. Account B sells the option

Parameters:

Index (_p): 8
AssetAmount: 0.01
Strike: 500,000
Price: 10 dai
Expiry: 1637686800

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xe103e3fabe65ad2445592acd6972510e7e1f72c36da0890bccf330a316b137ba

Screenshot:



Result: SUCCESS

5. **Account A changes a newBid and a newAsk**
    a. **Account A changes a newBid**
        i. **Account A creates a newBid (using index 2)**

Parameters:

Asset Amount: .01
Strike: 500,000
Price: 5 dai
Expiry: 1637679600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xb95ee0e06b1e9b9edffb1e6d34dfd89f87e768cc43d677e4416eaedb7c7c80cd

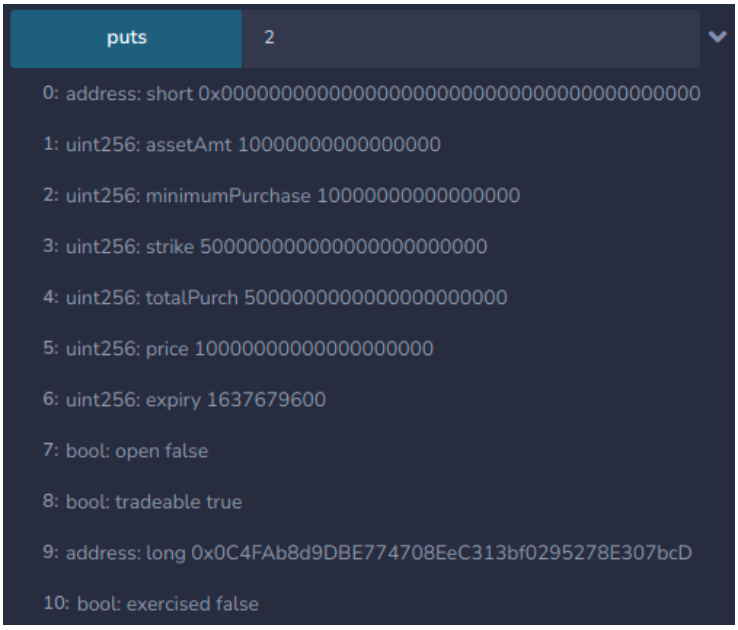
        ii. **Account A increase the price by 5 DAI (deliver 5 DAI into escrow)**

Parameters:

index(_p): 2
AssetAmount: .01
Minimum: .001
Strike: 500,000
Price: 10
Expiry: 1637679600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x5e4a1d635e0553ff5ea336ce068d70893b72305aadbed9e0c5bc07e9997ced02

Screenshot:



iii. Account A decreases the price by 5 DAI

Parameters:

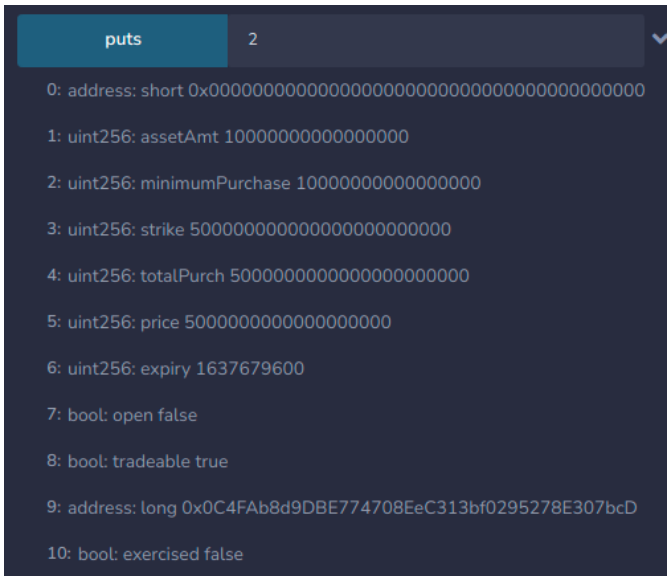
index(_p): 2

AssetAmount: .01
Minimum: .01
Strike: 500,000
Price: 5
Expiry: 1637679600

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x469aed9e6147bb76537f09c4837199dd72d877c0115026dc38dc8b6aa874e59f

Screenshot:



iv. **Account A changes the asset amount, strike**

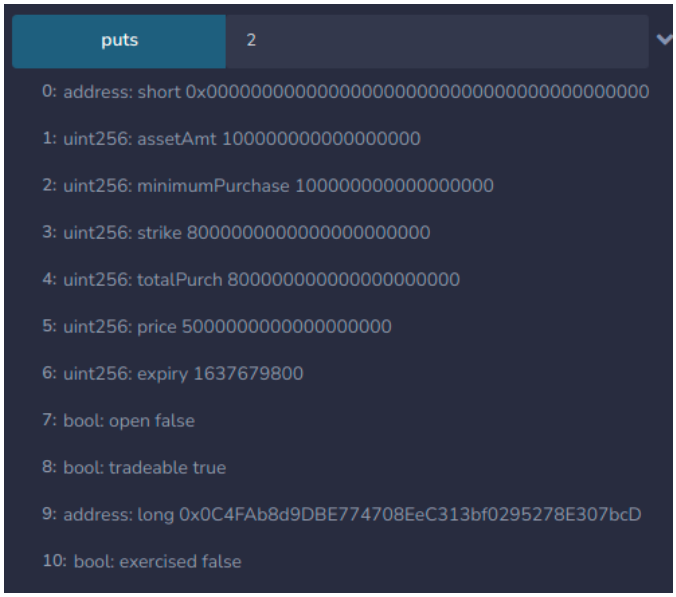
Parameters:

index(_p): 2
AssetAmount: .1
Minimum: .01
Strike: 8,000
Price: 5
Expiry: 1637679800

Transaction Hash:

https://rinkeby.etherscan.io/tx/0xd2343fa23f0ca0ad882d37938c6e61ac669ba1d678f9aa6e6b83352350b9ba49

Screenshot:



b. **Account A changes a newAsk**

i. **Account A creates a newAsk (using index 3)**

Parameters:

AssetAmount: 0.1
Minimum: 0.01
Strike: 10,000
Price: 10
Expiry: 1637679600

https://rinkeby.etherscan.io/tx/0x3fc4bf8d0d87be0fa61b76d40aefb167cdeaa3ccca22d79d1b57bd58a165acbb

ii. **Account A increases the Asset Amount to .15**

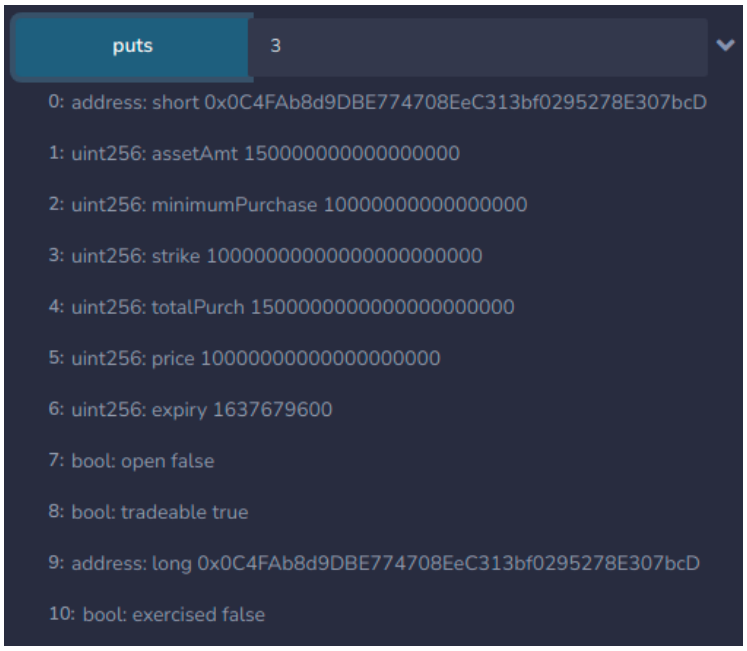
Parameters:

index (_p): 3
AssetAmount: .15
Minimum: 0.01
Strike: 10,000
Price: 10

Expiry: 1637679600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x2d42f93a9e9c04b4abfd8fb9c18c4a437ea1e6a09bb333d20ae75958d0514bd2

Screenshot:



iii. **Account A decreases the Asset Amount**

Parameters:

index (_p): 3
AssetAmount: .1
Minimum: 0.01
Strike: 10,000
Price: 10
Expiry: 1637679600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x4fc37040b282aa9fc4c9359811b10857f833621268630c71ac4c3d2087693a84

Screenshot:

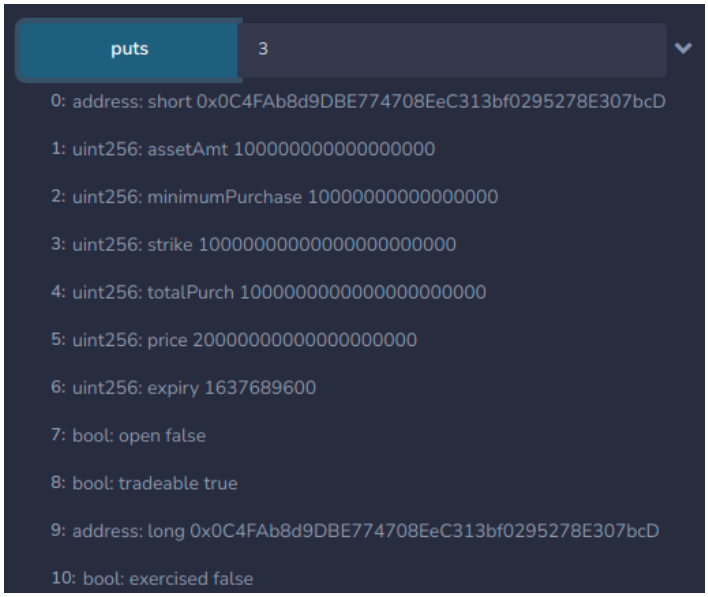
iv. **Account A changes the Price, Expiry**

Parameters:

index (_p): 3
AssetAmount: .1
Minimum: 0.01
Strike: 10,000
Price: 20
Expiry: 1637689600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xd5a4ce06bcf13884b7fefcfdd86da3023a493d28d74f3a60dfa3882690ece149

Screenshot:



Result: SUCCESS

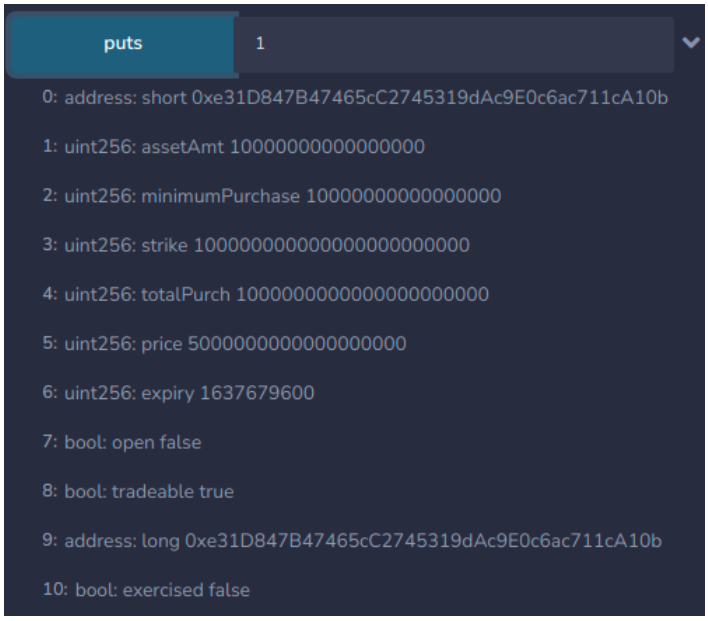
6. **Account C writes a newAsk**

Parameters:

AssetAmount: 0.01
Minimum: 0.01
Strike: 100,000
Price: 5
Expiry: 1637679600

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xc3ac71cd92446516c9094a227e159ccf07e8ed9a7b209364c927c1182e73b807

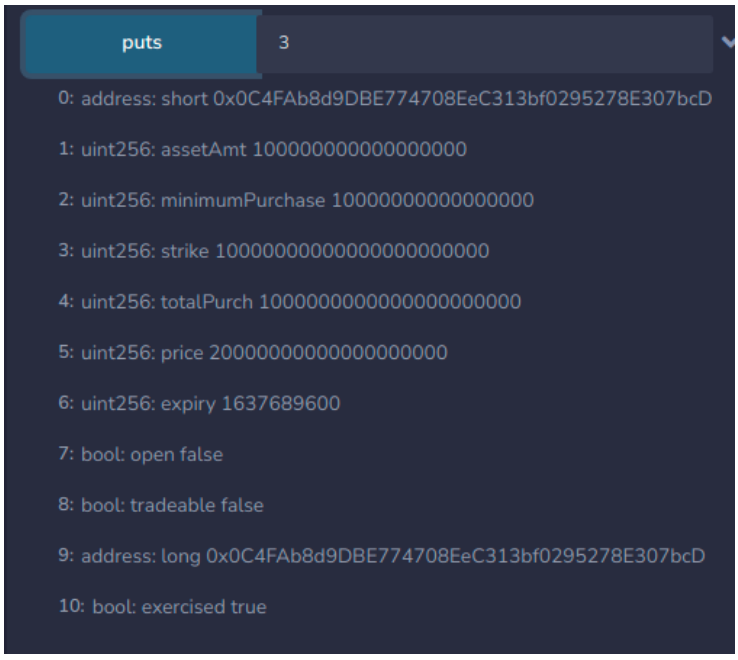
Screenshot:



Result: SUCCESS

7. **Account A cancels a newAsk**
   Parameters: put index (_p): 3
   Transaction Hash:
   https://rinkeby.etherscan.io/tx/0x08cf72a24487abf5c2082adaeb0881595e6e35292f9023d139046fb0a6d764c5

Screenshot:



Result: SUCCESS

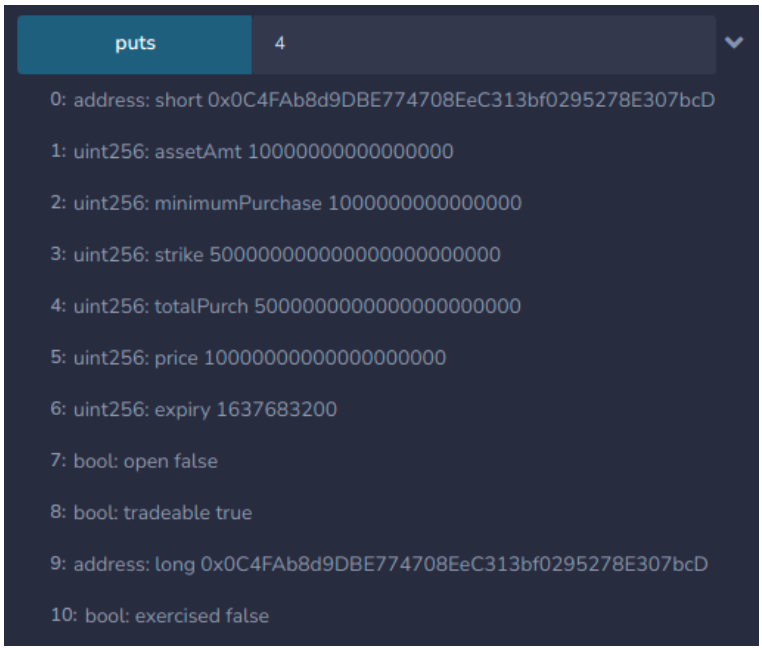
8. **Account B & C & D purchase options from account A**
    a. **Account A creates a newAsk**

Parameters:

AssetAmount: .01
Minimum: .001
Strike: 500,000
Price: 10
Expiry: 1637683200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x08cf72a24487abf5c2082adaeb0881595e6e35292f9023d139046fb0a6d764c5

Screenshot:



    b. **Account B purchases 30% of the newAsk**

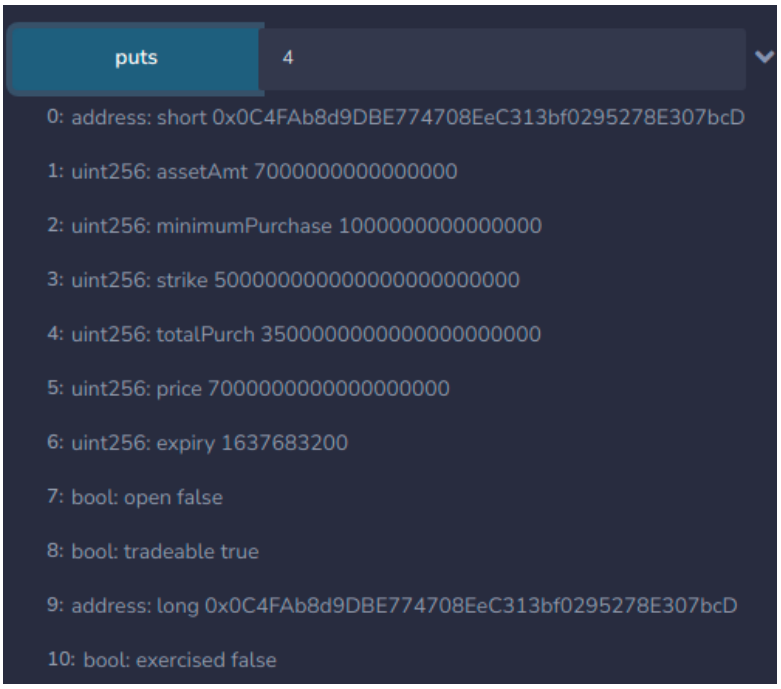
Parameters:

Index (_p): 4
assetAmount: .003
Strike: 500,000
Price: 3
Expiry: 1637683200

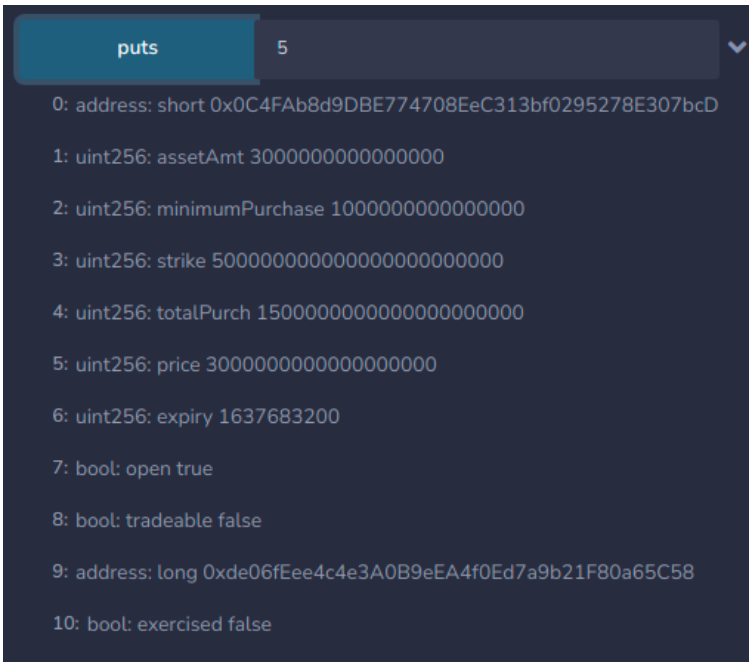
Transaction Hash:
https://rinkeby.etherscan.io/tx/0xb50f0eb5ce1b94890dd43f34eacaa10b29871834d02a728e032103c7b81cf049

Screenshot:

Remaining newAsk with the updated asset amount, price, total purchase



New call created with the long and short positions matching what account B purchased:



c. Account C purchases the same 30% of the newAsk

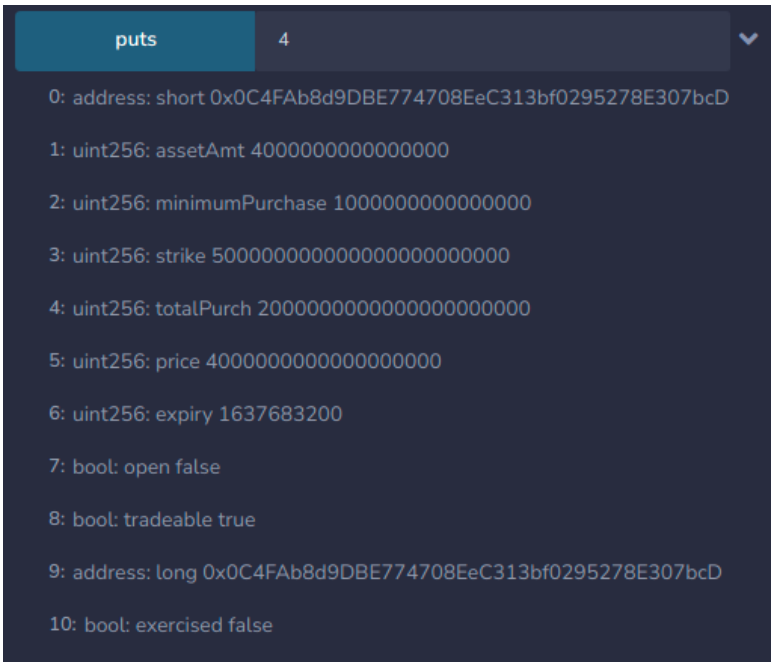
Parameters:

Index (_p): 4
assetAmount: .003
Strike: 500,000
Price: 3
Expiry: 1637683200

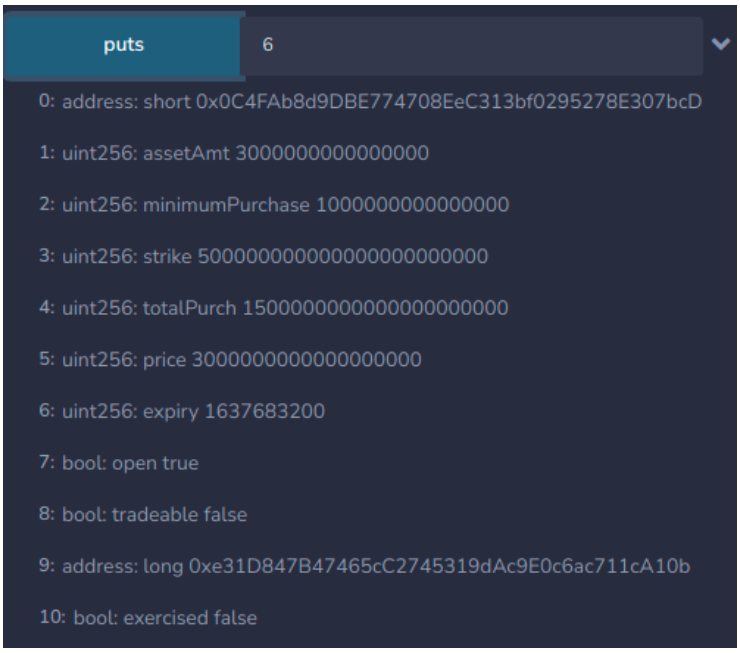
Transaction Hash:
https://rinkeby.etherscan.io/tx/0xa504788923fc50bdb19013260fd2d88c014583146a0c932c5d80fbb95c67ebe7

Screenshot:

Updated newAsk with asset amount remaining, price and total purchase re calculated



New open call option with all parameters calculated for index



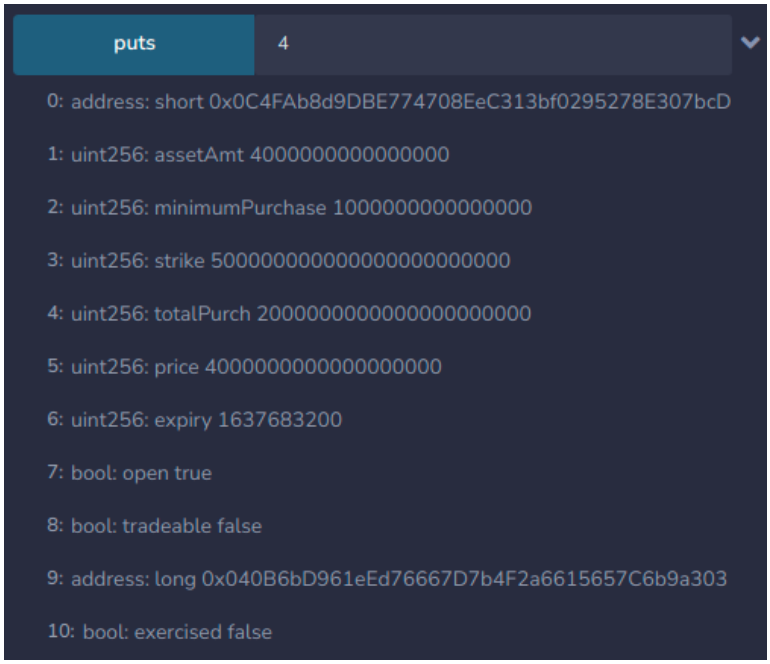
d. Account D purchases 40% of the newAsk

Parameters:

Index (_p): 4
assetAmount: .004
Strike: 500,000
Price: 4
Expiry: 1637683200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x54fd68a99694ad35bc3fe33d79c75fd0047f8d81923d2fadeffc3d3a17fd6349

Screenshot:



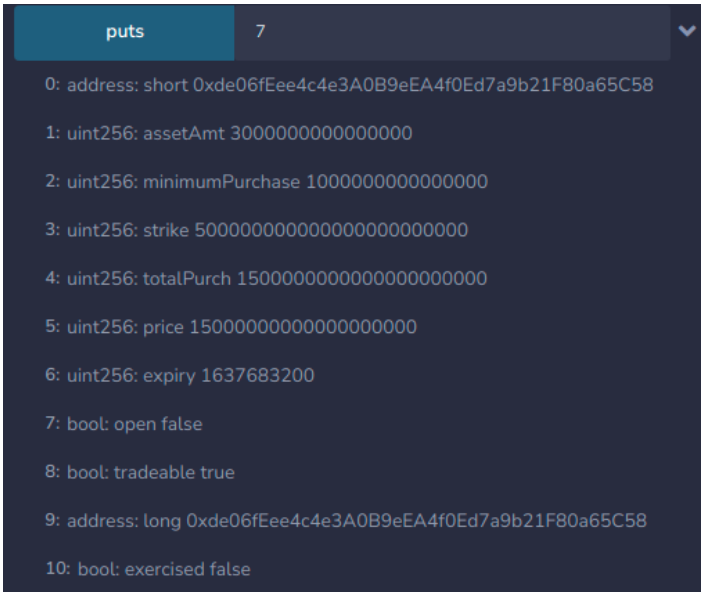
Result: SUCCESS

9. **Account B writes a newAsk and account A buys from account B**
    a. Using Index from previous section sold by A (index 5)
    b. **Account B writes a new Ask (index 7)**

Parameters:

assetAmount: 0.003
Minimum amount: 0.001
Strike: 500,000
Price: 15
Expiry: 1637683200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xe52be532004f7d62451684e51b08563cc39afa9676fd0551e74dba9452634700



c. **Account A sells put option to B new ask index (7)**

Parameters:

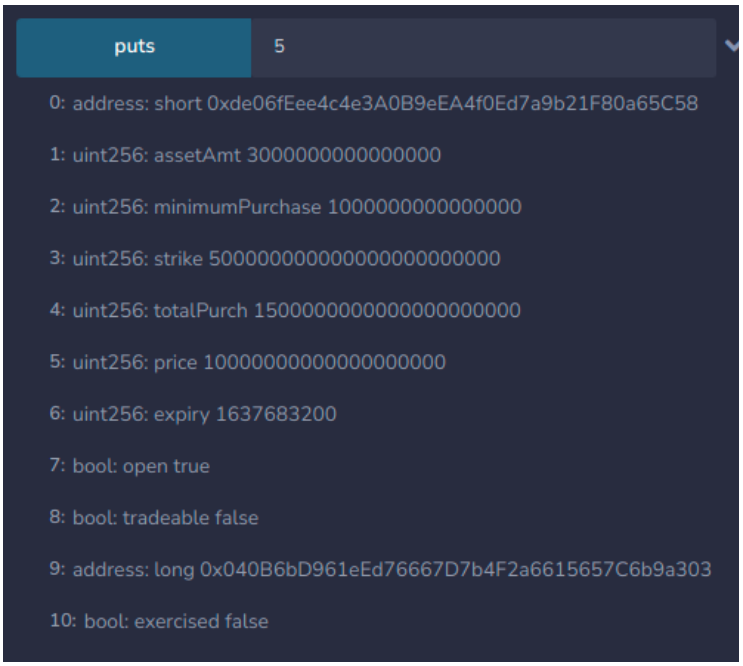
Openput index (_p): 5
newAsk index (_q): 7
Price: 15

Transaction Hash:

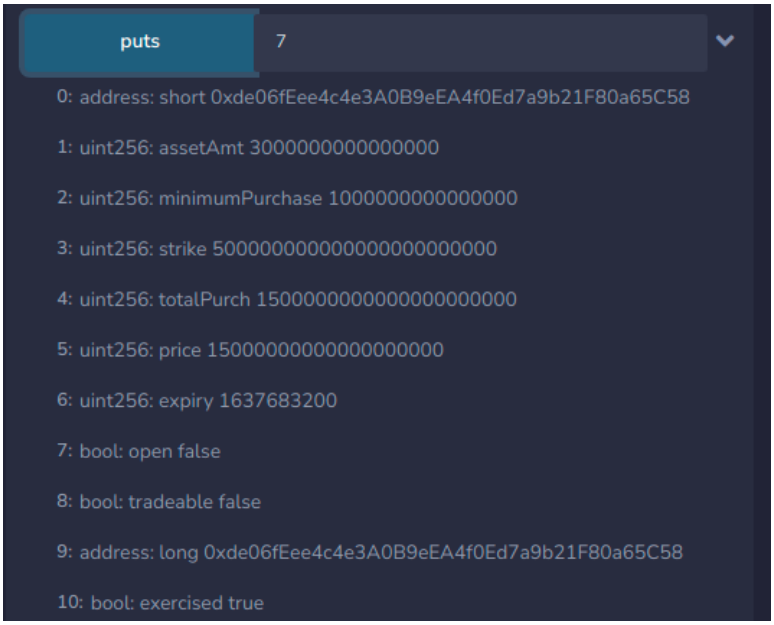
https://rinkeby.etherscan.io/tx/0x821d5375b35d5dc749c702a1d5196acd01cb4e6219fcb0f637d1c872e3004078

Screenshot:

Call update showing that Account B is now the short



And the newAsk is closed out by being exercised == true



Result: SUCCESS

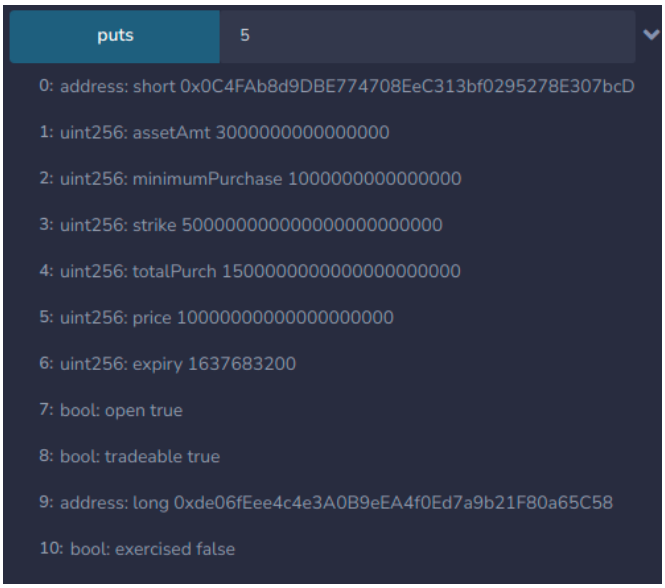
10. **Account B sends option index 5 up for sale**

Parameters:

index (_p): 5
Price: 10
Tradeable: true

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xad57f8faf5fb04d10c167962e73c374c6a3719e8a8c47b10e034f958fa083e1e

Screenshot:



Result: SUCCESS

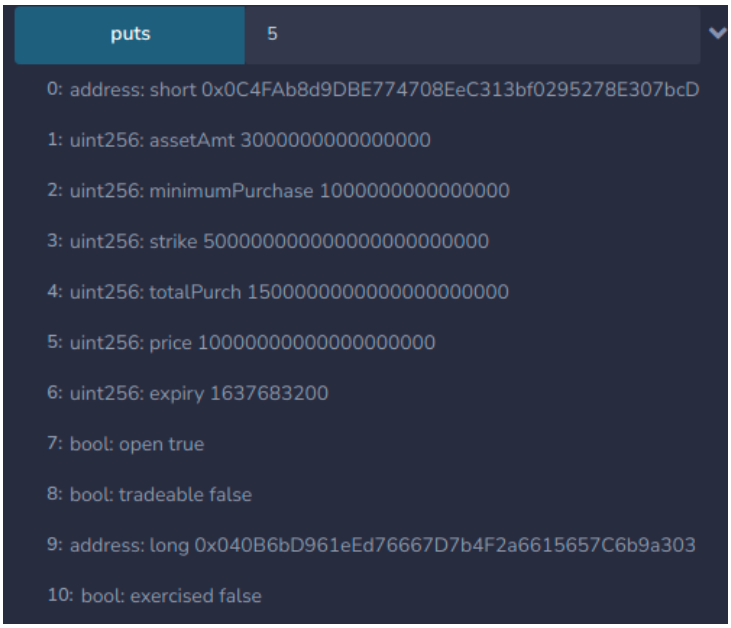
11. **Account D buys an open option from Account B**

Parameters:

index (_p): 5
assetAmt: 0.003
Strike: 500,000
Price: 10
Expiry: 1637683200

Transaction Hash:
https://rinkeby.etherscan.io/tx/0x5e17948e3ffba058fee80ce7ee4ce072cdc23de5a9ee68215aecf95e43af0ba3

Screenshot:



Result: SUCCESS

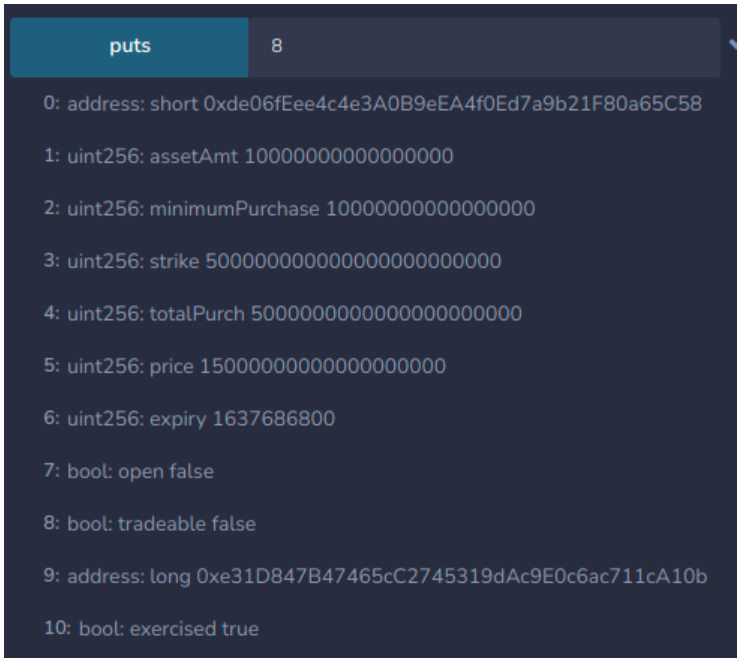
12. **Account C exercises an option (index 6)**

Parameters:

index (_p): 8

Transaction Hash:
https://rinkeby.etherscan.io/tx/0xe3f48dc43f8ebef54619a44372da13db9cbc0a1b1e48a36c08b3c3cf73df3e07

Screenshot:



Result: SUCCESS

13. **Account A cashClose exercises an option**

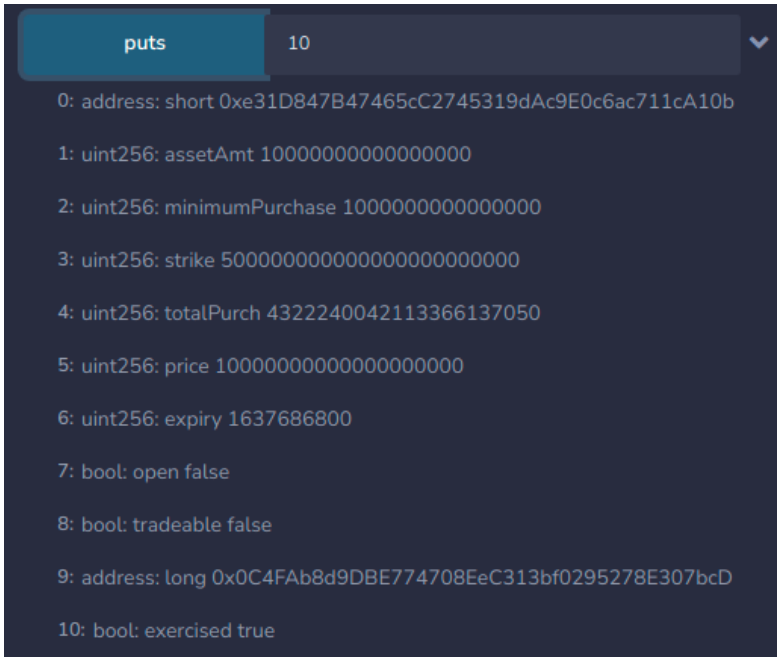
Parameters:

index (_p): 10

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x093835a6af7d7475c4dbc7f8cc49cf01b5a9a95457c94e39c98360c447907a00

Screenshot:



Result: SUCCESS

14. **Account C returns 2 expired calls**

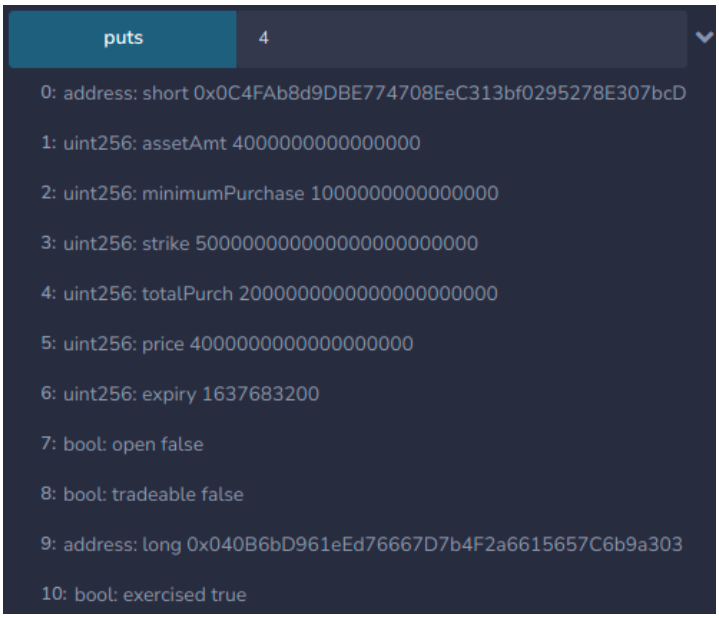
Parameters:

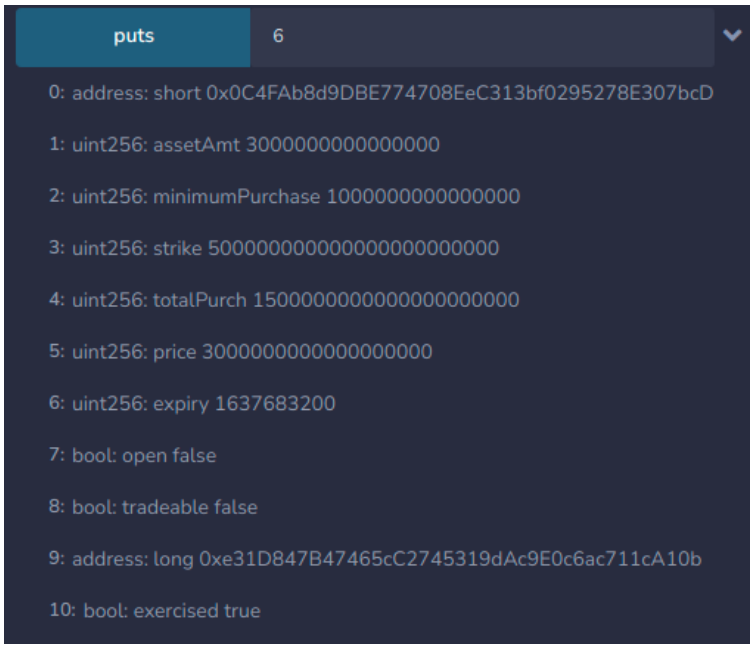
index (_p): [4,6]

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x1fa6bb96c7795701799eeb4801db55adbffef018213dc2ca8807e365d0bcea41

Screenshot:





Result: SUCCESS

15. **Account B rolls multiple expired calls (indexes 12, 13) into one put**

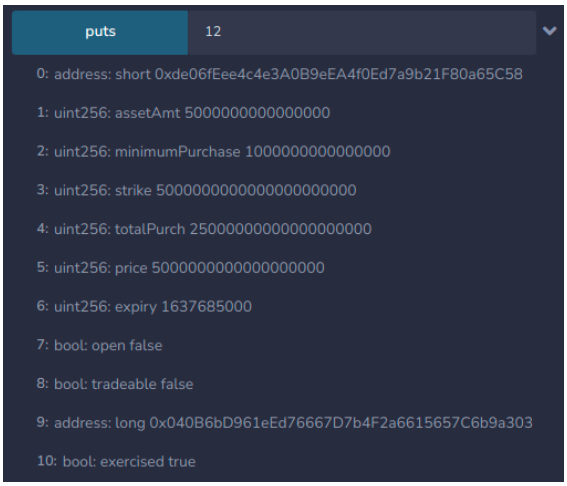
Parameters:

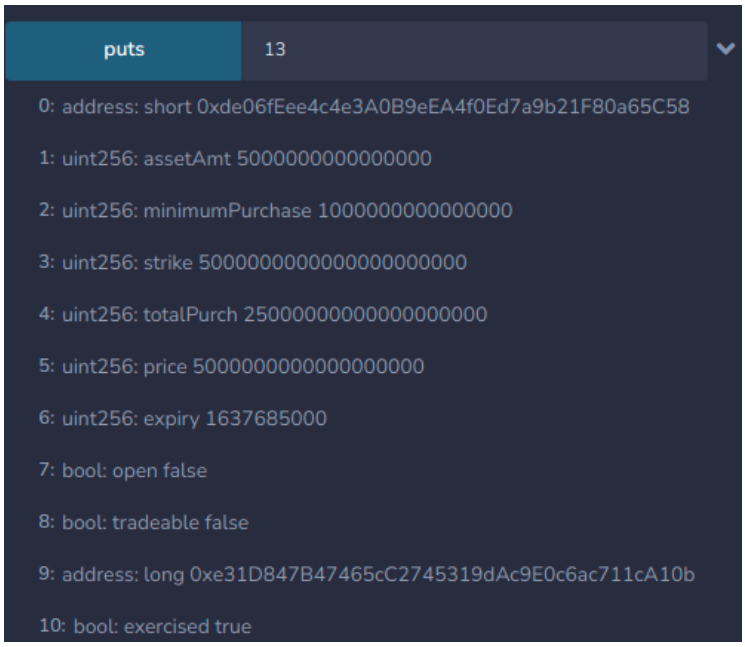
index (_p): [12, 13]

Transaction Hash:

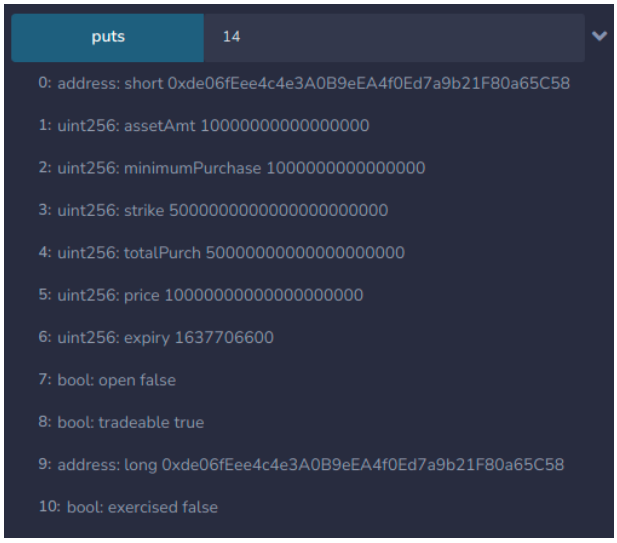
https://rinkeby.etherscan.io/tx/0x097f20624f90cea123a7205327a1130fbce54355cca92ccc6066169aa730a00b

Screenshot:





And the new call to replace all of them:



Result: SUCCESS

16. **Account A owns a long call and multiswap cash closes an option**

Parameters:

Index (_p): 11

newOwner: 0x049a8A4CBf1f8a1a160E0ee8dE94B1775204eB8B (this is also the deployment of HedgeyAnySwap.sol)

Path: ["0x5592ec0cfb4dbc12d3ab100b257153436a1f0fea","0xbf7a7169562078c96f0ec1a8afd6ae50f12e5a99","0xc778417e063141139fce010982780140aa0cd5ab"]

Transaction Hash:

https://rinkeby.etherscan.io/tx/0x1fab4e4d0761a18de71e8b1c1bd6bceecd28cfb0b294adc9ac70ddbc479bc19d

Screenshot: