



EasyAuction.json

```
{ "address": "0x0b7fFc1f4AD541A4Ed16b40D8c37f0929158D101", "abi": [ { "inputs": [],  
"stateMutability": "nonpayable", "type": "constructor" }, { "anonymous": false, "inputs": [ {  
"indexed": true, "internalType": "uint256", "name": "auctionId", "type": "uint256" }, { "indexed":  
false, "internalType": "uint96", "name": "soldAuctioningTokens", "type": "uint96" }, { "indexed":  
false, "internalType": "uint96", "name": "soldBiddingTokens", "type": "uint96" }, { "indexed":  
false, "internalType": "bytes32", "name": "clearingPriceOrder", "type": "bytes32" } ], "name":  
"AuctionCleared", "type": "event" }, { "anonymous": false, "inputs": [ { "indexed": true,  
"internalType": "uint256", "name": "auctionId", "type": "uint256" }, { "indexed": true,  
"internalType": "uint64", "name": "userId", "type": "uint64" }, { "indexed": false, "internalType":  
"uint96", "name": "buyAmount", "type": "uint96" }, { "indexed": false, "internalType": "uint96",  
"name": "sellAmount", "type": "uint96" } ], "name": "CancellationSellOrder", "type": "event" }, {  
"anonymous": false, "inputs": [ { "indexed": true, "internalType": "uint256", "name": "auctionId",  
"type": "uint256" }, { "indexed": true, "internalType": "uint64", "name": "userId", "type": "uint64"  
}, { "indexed": false, "internalType": "uint96", "name": "buyAmount", "type": "uint96" }, {  
"indexed": false, "internalType": "uint96", "name": "sellAmount", "type": "uint96" } ], "name":  
"ClaimedFromOrder", "type": "event" }, { "anonymous": false, "inputs": [ { "indexed": true,  
"internalType": "uint256", "name": "auctionId", "type": "uint256" }, { "indexed": true,  
"internalType": "contract IERC20", "name": "_auctioningToken", "type": "address" }, { "indexed":  
true, "internalType": "contract IERC20", "name": "_biddingToken", "type": "address" }, {  
"indexed": false, "internalType": "uint256", "name": "orderCancellationEndDate", "type":  
"uint256" }, { "indexed": false, "internalType": "uint256", "name": "auctionEndDate", "type":  
"uint256" }, { "indexed": false, "internalType": "uint64", "name": "userId", "type": "uint64" }, {  
"indexed": false, "internalType": "uint96", "name": "_auctionedSellAmount", "type": "uint96" }, {  
"indexed": false, "internalType": "uint96", "name": "_minBuyAmount", "type": "uint96" }, {  
"indexed": false, "internalType": "uint256", "name": "minimumBiddingAmountPerOrder", "type":  
"uint256" }, { "indexed": false, "internalType": "uint256", "name": "minFundingThreshold",  
"type": "uint256" }, { "indexed": false, "internalType": "address", "name": "allowListContract",  
"type": "address" }, { "indexed": false, "internalType": "bytes", "name": "allowListData", "type":  
"bytes" } ], "name": "NewAuction", "type": "event" }, { "anonymous": false, "inputs": [ {  
"indexed": true, "internalType": "uint256", "name": "auctionId", "type": "uint256" }, { "indexed":  
true, "internalType": "uint64", "name": "userId", "type": "uint64" }, { "indexed": false,  
"internalType": "uint96", "name": "buyAmount", "type": "uint96" }, { "indexed": false,  
"internalType": "uint96", "name": "sellAmount", "type": "uint96" } ], "name": "NewSellOrder",
```

```
"type": "event" }, { "anonymous": false, "inputs": [ { "indexed": true, "internalType": "uint64",
"name": "userId", "type": "uint64" }, { "indexed": true, "internalType": "address", "name":
"userAddress", "type": "address" } ], "name": "NewUser", "type": "event" }, { "anonymous":
false, "inputs": [ { "indexed": true, "internalType": "address", "name": "previousOwner", "type":
"address" }, { "indexed": true, "internalType": "address", "name": "newOwner", "type":
"address" } ], "name": "OwnershipTransferred", "type": "event" }, { "anonymous": false, "inputs":
[ { "indexed": true, "internalType": "address", "name": "user", "type": "address" }, { "indexed":
false, "internalType": "uint64", "name": "userId", "type": "uint64" } ], "name": "UserRegistration",
"type": "event" }, { "inputs": [], "name": "FEE_DENOMINATOR", "outputs": [ { "internalType":
"uint256", "name": "", "type": "uint256" } ], "stateMutability": "view", "type": "function" }, {
"inputs": [ { "internalType": "uint256", "name": "", "type": "uint256" } ], "name":
"auctionAccessData", "outputs": [ { "internalType": "bytes", "name": "", "type": "bytes" } ],
"stateMutability": "view", "type": "function" }, { "inputs": [ { "internalType": "uint256", "name": "",
"type": "uint256" } ], "name": "auctionAccessManager", "outputs": [ { "internalType": "address",
"name": "", "type": "address" } ], "stateMutability": "view", "type": "function" }, { "inputs": [],
"name": "auctionCounter", "outputs": [ { "internalType": "uint256", "name": "", "type": "uint256" }
], "stateMutability": "view", "type": "function" }, { "inputs": [ { "internalType": "uint256", "name":
"", "type": "uint256" } ], "name": "auctionData", "outputs": [ { "internalType": "contract IERC20",
"name": "auctioningToken", "type": "address" }, { "internalType": "contract IERC20", "name":
"biddingToken", "type": "address" }, { "internalType": "uint256", "name":
"orderCancellationEndDate", "type": "uint256" }, { "internalType": "uint256", "name":
"auctionEndDate", "type": "uint256" }, { "internalType": "bytes32", "name": "initialAuctionOrder",
"type": "bytes32" }, { "internalType": "uint256", "name": "minimumBiddingAmountPerOrder",
"type": "uint256" }, { "internalType": "uint256", "name": "interimSumBidAmount", "type":
"uint256" }, { "internalType": "bytes32", "name": "interimOrder", "type": "bytes32" }, {
"internalType": "bytes32", "name": "clearingPriceOrder", "type": "bytes32" }, { "internalType":
"uint96", "name": "volumeClearingPriceOrder", "type": "uint96" }, { "internalType": "bool",
"name": "minFundingThresholdNotReached", "type": "bool" }, { "internalType": "bool", "name":
"isAtomicClosureAllowed", "type": "bool" }, { "internalType": "uint256", "name": "feeNumerator",
"type": "uint256" }, { "internalType": "uint256", "name": "minFundingThreshold", "type":
"uint256" } ], "stateMutability": "view", "type": "function" }, { "inputs": [ { "internalType": "uint256",
"name": "auctionId", "type": "uint256" }, { "internalType": "bytes32[]", "name": "_sellOrders",
"type": "bytes32[]" } ], "name": "cancelSellOrders", "outputs": [], "stateMutability": "nonpayable",
"type": "function" }, { "inputs": [ { "internalType": "uint256", "name": "auctionId", "type": "uint256"
}, { "internalType": "bytes32[]", "name": "orders", "type": "bytes32[]" } ], "name":
"claimFromParticipantOrder", "outputs": [ { "internalType": "uint256", "name":
"sumAuctioningTokenAmount", "type": "uint256" }, { "internalType": "uint256", "name":
"sumBiddingTokenAmount", "type": "uint256" } ], "stateMutability": "nonpayable", "type":
"function" }, { "inputs": [ { "internalType": "uint256", "name": "auctionId", "type": "uint256" }, {
```

```
"internalType": "bytes32", "name": "order", "type": "bytes32" } ], "name": "containsOrder",
"outputs": [ { "internalType": "bool", "name": "", "type": "bool" } ], "stateMutability": "view",
"type": "function" }, { "inputs": [], "name": "feeNumerator", "outputs": [ { "internalType":
"uint256", "name": "", "type": "uint256" } ], "stateMutability": "view", "type": "function" }, {
"inputs": [], "name": "feeReceiverUserId", "outputs": [ { "internalType": "uint64", "name": "",
"type": "uint64" } ], "stateMutability": "view", "type": "function" }, { "inputs": [ { "internalType":
"uint256", "name": "auctionId", "type": "uint256" } ], "name": "getSecondsRemainingInBatch",
"outputs": [ { "internalType": "uint256", "name": "", "type": "uint256" } ], "stateMutability": "view",
"type": "function" }, { "inputs": [ { "internalType": "address", "name": "user", "type": "address" } ],
"name": "getUserId", "outputs": [ { "internalType": "uint64", "name": "userId", "type": "uint64" } ],
"stateMutability": "nonpayable", "type": "function" }, { "inputs": [ { "internalType": "contract
IERC20", "name": "_auctioningToken", "type": "address" }, { "internalType": "contract IERC20",
"name": "_biddingToken", "type": "address" }, { "internalType": "uint256", "name":
"orderCancellationEndDate", "type": "uint256" }, { "internalType": "uint256", "name":
"auctionEndDate", "type": "uint256" }, { "internalType": "uint96", "name":
"_auctionedSellAmount", "type": "uint96" }, { "internalType": "uint96", "name":
"_minBuyAmount", "type": "uint96" }, { "internalType": "uint256", "name":
"minimumBiddingAmountPerOrder", "type": "uint256" }, { "internalType": "uint256", "name":
"minFundingThreshold", "type": "uint256" }, { "internalType": "bool", "name":
"isAtomicClosureAllowed", "type": "bool" }, { "internalType": "address", "name":
"accessManagerContract", "type": "address" }, { "internalType": "bytes", "name":
"accessManagerContractData", "type": "bytes" } ], "name": "initiateAuction", "outputs": [ {
"internalType": "uint256", "name": "", "type": "uint256" } ], "stateMutability": "nonpayable",
"type": "function" }, { "inputs": [], "name": "numUsers", "outputs": [ { "internalType": "uint64",
"name": "", "type": "uint64" } ], "stateMutability": "view", "type": "function" }, { "inputs": [],
"name": "owner", "outputs": [ { "internalType": "address", "name": "", "type": "address" } ],
"stateMutability": "view", "type": "function" }, { "inputs": [ { "internalType": "uint256", "name":
"auctionId", "type": "uint256" }, { "internalType": "uint96[]", "name": "_minBuyAmounts", "type":
"uint96[]" }, { "internalType": "uint96[]", "name": "_sellAmounts", "type": "uint96[]" }, {
"internalType": "bytes32[]", "name": "_prevSellOrders", "type": "bytes32[]" }, { "internalType":
"bytes", "name": "allowListCallData", "type": "bytes" } ], "name": "placeSellOrders", "outputs": [
{ "internalType": "uint64", "name": "userId", "type": "uint64" } ], "stateMutability": "nonpayable",
"type": "function" }, { "inputs": [ { "internalType": "uint256", "name": "auctionId", "type": "uint256"
}, { "internalType": "uint96[]", "name": "_minBuyAmounts", "type": "uint96[]" }, { "internalType":
"uint96[]", "name": "_sellAmounts", "type": "uint96[]" }, { "internalType": "bytes32[]", "name":
"_prevSellOrders", "type": "bytes32[]" }, { "internalType": "bytes", "name": "allowListCallData",
"type": "bytes" }, { "internalType": "address", "name": "orderSubmitter", "type": "address" } ],
"name": "placeSellOrdersOnBehalf", "outputs": [ { "internalType": "uint64", "name": "userId",
"type": "uint64" } ], "stateMutability": "nonpayable", "type": "function" }, { "inputs": [ {
```

```
"internalType": "uint256", "name": "auctionId", "type": "uint256" }, { "internalType": "uint256",  
"name": "iterationSteps", "type": "uint256" } ], "name": "precalculateSellAmountSum",  
"outputs": [], "stateMutability": "nonpayable", "type": "function" }, { "inputs": [ { "internalType":  
"address", "name": "user", "type": "address" } ], "name": "registerUser", "outputs": [ {  
"internalType": "uint64", "name": "userId", "type": "uint64" } ], "stateMutability": "nonpayable",  
"type": "function" }, { "inputs": [], "name": "renounceOwnership", "outputs": [], "stateMutability":  
"nonpayable", "type": "function" }, { "inputs": [ { "internalType": "uint256", "name":  
"newFeeNumerator", "type": "uint256" }, { "internalType": "address", "name":  
"newfeeReceiverAddress", "type": "address" } ], "name": "setFeeParameters", "outputs": [],  
"stateMutability": "nonpayable", "type": "function" }, { "inputs": [ { "internalType": "uint256",  
"name": "auctionId", "type": "uint256" } ], "name": "settleAuction", "outputs": [ { "internalType":  
"bytes32", "name": "clearingOrder", "type": "bytes32" } ], "stateMutability": "nonpayable",  
"type": "function" }, { "inputs": [ { "internalType": "uint256", "name": "auctionId", "type": "uint256"  
}, { "internalType": "uint96[]", "name": "_minBuyAmount", "type": "uint96[]" }, { "internalType":  
"uint96[]", "name": "_sellAmount", "type": "uint96[]" }, { "internalType": "bytes32[]", "name":  
"_prevSellOrder", "type": "bytes32[]" }, { "internalType": "bytes", "name": "allowListCallData",  
"type": "bytes" } ], "name": "settleAuctionAtomically", "outputs": [], "stateMutability":  
"nonpayable", "type": "function" }, { "inputs": [ { "internalType": "address", "name": "newOwner",  
"type": "address" } ], "name": "transferOwnership", "outputs": [], "stateMutability": "nonpayable",  
"type": "function" } ], "transactionHash":  
"0xa7ad659a9762720bd86a30b49a3e139928cc2a27d0863ab78110e19d2bef8a51", "receipt":  
{ "to": null, "from": "0x8AB8530AEe3E5eD7EcF5D7CA69c903Ed04595094",  
"contractAddress": "0x0b7fFc1f4AD541A4Ed16b40D8c37f0929158D101", "transactionIndex":  
2, "gasUsed": "5135197", "logsBloom":  
"0x0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
4000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000  
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000",  
"blockHash":  
"0x51256d75e095f5a7443ae66ecc713d815ba6528e8466532be59220a8f70f9164",  
"transactionHash":  
"0xa7ad659a9762720bd86a30b49a3e139928cc2a27d0863ab78110e19d2bef8a51", "logs": [ {  
"transactionIndex": 2, "blockNumber": 12135186, "transactionHash":  
"0xa7ad659a9762720bd86a30b49a3e139928cc2a27d0863ab78110e19d2bef8a51",  
"address": "0x0b7fFc1f4AD541A4Ed16b40D8c37f0929158D101", "topics": [  
"0x8be0079c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e0".
```

```
"0x0000000000000000000000000000000000000000000000000000000000000000",
"0x00000000000000000000000000000000000000000000000000000000000000008ab8530aee3e5ed7ecf5d7ca69c903ed04595094" ], "data":
"0x", "logIndex": 5, "blockHash":
"0x51256d75e095f5a7443ae66ecc713d815ba6528e8466532be59220a8f70f9164" } ],
"blockNumber": 12135186, "cumulativeGasUsed": "5282924", "status": 1, "byzantium": true },
"args": [], "solcInputHash": "ad868d512eafc63430765fc273a2c47b", "metadata": "{\n  \"compiler\":\n  {\n    \"version\": \"0.6.12+commit.27d51765\", \"language\": \"Solidity\", \"output\": {\n      \"abi\": [\n        {\n          \"inputs\":\n          [\n            {\n              \"indexed\": true, \"internalType\": \"uint256\", \"name\": \"auctionId\", \"type\": \"uint256\"},\n            {\n              \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"soldAuctioningTokens\", \"type\": \"uint96\"},\n            {\n              \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"soldBiddingTokens\", \"type\": \"uint96\"},\n            {\n              \"indexed\": false, \"internalType\": \"bytes32\", \"name\": \"clearingPriceOrder\", \"type\": \"bytes32\"},\n            {\n              \"name\": \"AuctionCleared\", \"type\": \"event\", {\n                \"anonymous\": false, \"inputs\":\n                [\n                  {\n                    \"indexed\": true, \"internalType\": \"uint256\", \"name\": \"auctionId\", \"type\": \"uint256\"},\n                  {\n                    \"indexed\": true, \"internalType\": \"uint64\", \"name\": \"userId\", \"type\": \"uint64\"},\n                  {\n                    \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"buyAmount\", \"type\": \"uint96\"},\n                  {\n                    \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"sellAmount\", \"type\": \"uint96\"},\n                  {\n                    \"name\": \"CancellationSellOrder\", \"type\": \"event\", {\n                      \"anonymous\": false, \"inputs\":\n                      [\n                        {\n                          \"indexed\": true, \"internalType\": \"uint256\", \"name\": \"auctionId\", \"type\": \"uint256\"},\n                        {\n                          \"indexed\": true, \"internalType\": \"uint64\", \"name\": \"userId\", \"type\": \"uint64\"},\n                        {\n                          \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"buyAmount\", \"type\": \"uint96\"},\n                        {\n                          \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"sellAmount\", \"type\": \"uint96\"},\n                        {\n                          \"name\": \"ClaimedFromOrder\", \"type\": \"event\", {\n                            \"anonymous\": false, \"inputs\":\n                            [\n                              {\n                                \"indexed\": true, \"internalType\": \"uint256\", \"name\": \"auctionId\", \"type\": \"uint256\"},\n                              {\n                                \"indexed\": true, \"internalType\": \"contract\nIERC20\", \"name\": \"_auctioningToken\", \"type\": \"address\"},\n                              {\n                                \"indexed\": true, \"internalType\": \"contract\nIERC20\", \"name\": \"_biddingToken\", \"type\": \"address\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"uint256\", \"name\": \"orderCancellationEndDate\", \"type\": \"uint256\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"uint256\", \"name\": \"auctionEndDate\", \"type\": \"uint256\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"uint64\", \"name\": \"userId\", \"type\": \"uint64\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"_auctionedSellAmount\", \"type\": \"uint96\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"uint96\", \"name\": \"_minBuyAmount\", \"type\": \"uint96\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"uint256\", \"name\": \"minimumBiddingAmountPerOrder\", \"type\": \"uint256\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"uint256\", \"name\": \"minFundingThreshold\", \"type\": \"uint256\"},\n                              {\n                                \"indexed\": false, \"internalType\": \"address\", \"name\": \"allowListContract\", \"type\": \"address\"}\n                            ]\n                          }\n                        }\n                      ]\n                    }\n                  }\n                ]\n              }\n            }\n          ]\n        }\n      ]\n    }\n  }\n}"
```

```
,
{"indexed":false,"internalType":"bytes","name":"allowListData","type":"bytes"},"name":
"NewAuction","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"internalType":"uint256","name":"auctionId","type":"uint256"},
{"indexed":true,"internalType":"uint64","name":"userId","type":"uint64"},
{"indexed":false,"internalType":"uint96","name":"buyAmount","type":"uint96"},
{"indexed":false,"internalType":"uint96","name":"sellAmount","type":"uint96"}],"name":
"NewSellOrder","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"internalType":"uint64","name":"userId","type":"uint64"},
{"indexed":true,"internalType":"address","name":"userAddress","type":"address"}],"name":
"NewUser","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"internalType":"address","name":"previousOwner","type":"address"},
{"indexed":true,"internalType":"address","name":"newOwner","type":"address"}],"name":
"OwnershipTransferred","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"internalType":"address","name":"user","type":"address"},
{"indexed":false,"internalType":"uint64","name":"userId","type":"uint64"}],"name":
"UserRegistration","type":"event"},{"inputs":[],"name":"FEE_DENOMINATOR","outputs":
[{"internalType":"uint256","name":"","type":"uint256"}],"stateMutability":"view","type":
"function"},{"inputs":
[{"internalType":"uint256","name":"","type":"uint256"}],"name":"auctionAccessData","
outputs":
[{"internalType":"bytes","name":"","type":"bytes"}],"stateMutability":"view","type":
"function"},{"inputs":
[{"internalType":"uint256","name":"","type":"uint256"}],"name":"auctionAccessManage
r","outputs":
[{"internalType":"address","name":"","type":"address"}],"stateMutability":"view","type":
"function"},{"inputs":[],"name":"auctionCounter","outputs":
[{"internalType":"uint256","name":"","type":"uint256"}],"stateMutability":"view","type":
"function"},{"inputs":
[{"internalType":"uint256","name":"","type":"uint256"}],"name":"auctionData","outputs":
[{"internalType":"contract IERC20","name":"auctioningToken","type":"address"},
{"internalType":"contract IERC20","name":"biddingToken","type":"address"},
{"internalType":"uint256","name":"orderCancellationEndDate","type":"uint256"},
{"internalType":"uint256","name":"auctionEndDate","type":"uint256"},
{"internalType":"bytes32","name":"initialAuctionOrder","type":"bytes32"},
{"internalType":"uint256","name":"minimumBiddingAmountPerOrder","type":"uint256"},
{"internalType":"uint256","name":"interimSumBidAmount","type":"uint256"},
{"internalType":"bytes32","name":"interimOrder","type":"bytes32"},
{"internalType":"bytes32","name":"clearingPriceOrder","type":"bytes32"},
```

```

{"internalType":"uint96","name":"volumeClearingPriceOrder","type":"uint96"},
{"internalType":"bool","name":"minFundingThresholdNotReached","type":"bool"},
{"internalType":"bool","name":"isAtomicClosureAllowed","type":"bool"},
{"internalType":"uint256","name":"feeNumerator","type":"uint256"},
{"internalType":"uint256","name":"minFundingThreshold","type":"uint256"}], "stateMutability":"view", "type":"function"}, {"inputs":
[{"internalType":"uint256","name":"auctionId","type":"uint256"},
{"internalType":"bytes32[]","name":"_sellOrders","type":"bytes32[]"}], "name":"cancelSellOrders", "outputs": [], "stateMutability":"nonpayable", "type":"function"}, {"inputs":
[{"internalType":"uint256","name":"auctionId","type":"uint256"},
{"internalType":"bytes32[]","name":"orders","type":"bytes32[]"}], "name":"claimFromParticipantOrder", "outputs":
[{"internalType":"uint256","name":"sumAuctioningTokenAmount","type":"uint256"},
{"internalType":"uint256","name":"sumBiddingTokenAmount","type":"uint256"}], "stateMutability":"nonpayable", "type":"function"}, {"inputs":
[{"internalType":"uint256","name":"auctionId","type":"uint256"},
{"internalType":"bytes32","name":"order","type":"bytes32"}], "name":"containsOrder", "outputs":
[{"internalType":"bool","name":"","type":"bool"}], "stateMutability":"view", "type":"function"}, {"inputs": [], "name":"feeNumerator", "outputs":
[{"internalType":"uint256","name":"","type":"uint256"}], "stateMutability":"view", "type":"function"}, {"inputs": [], "name":"feeReceiverUserId", "outputs":
[{"internalType":"uint64","name":"","type":"uint64"}], "stateMutability":"view", "type":"function"}, {"inputs":
[{"internalType":"uint256","name":"auctionId","type":"uint256"}], "name":"getSecondsRemainingInBatch", "outputs":
[{"internalType":"uint256","name":"","type":"uint256"}], "stateMutability":"view", "type":"function"}, {"inputs":
[{"internalType":"address","name":"user","type":"address"}], "name":"getUserId", "outputs":
[{"internalType":"uint64","name":"userId","type":"uint64"}], "stateMutability":"nonpayable", "type":"function"}, {"inputs": [{"internalType":"contractIERC20","name":"_auctioningToken","type":"address"}, {"internalType":"contractIERC20","name":"_biddingToken","type":"address"},
{"internalType":"uint256","name":"orderCancellationEndDate","type":"uint256"},
{"internalType":"uint256","name":"auctionEndDate","type":"uint256"},
{"internalType":"uint96","name":"_auctionedSellAmount","type":"uint96"},
{"internalType":"uint96","name":"_minBuyAmount","type":"uint96"},
{"internalType":"uint256","name":"minimumBiddingAmountPerOrder","type":"uint256"},

```

```
{\"internalType\":\"uint256\",\"name\":\"minFundingThreshold\",\"type\":\"uint256\"},
{\"internalType\":\"bool\",\"name\":\"isAtomicClosureAllowed\",\"type\":\"bool\"},
{\"internalType\":\"address\",\"name\":\"accessManagerContract\",\"type\":\"address\"},
{\"internalType\":\"bytes\",\"name\":\"accessManagerContractData\",\"type\":\"bytes\"}],\"name\":
\"initiateAuction\",\"outputs\":
[{\"internalType\":\"uint256\",\"name\":\"\",\"type\":\"uint256\"}],\"stateMutability\":\"nonpayable\",
\"type\":\"function\"},{\"inputs\":[],\"name\":\"numUsers\",\"outputs\":
[{\"internalType\":\"uint64\",\"name\":\"\",\"type\":\"uint64\"}],\"stateMutability\":\"view\", \"type\":\"f
unction\"},{\"inputs\":[],\"name\":\"owner\",\"outputs\":
[{\"internalType\":\"address\",\"name\":\"\",\"type\":\"address\"}],\"stateMutability\":\"view\", \"type\"
\":\"function\"},{\"inputs\":[{\"internalType\":\"uint256\",\"name\":\"auctionId\",\"type\":\"uint256\"},
{\"internalType\":\"uint96[]\",\"name\":\"_minBuyAmounts\",\"type\":\"uint96[]\"},
{\"internalType\":\"uint96[]\",\"name\":\"_sellAmounts\",\"type\":\"uint96[]\"},
{\"internalType\":\"bytes32[]\",\"name\":\"_prevSellOrders\",\"type\":\"bytes32[]\"},
{\"internalType\":\"bytes\",\"name\":\"allowListCallData\",\"type\":\"bytes\"}],\"name\":\"placeSell
Orders\",\"outputs\":
[{\"internalType\":\"uint64\",\"name\":\"userId\",\"type\":\"uint64\"}],\"stateMutability\":\"nonpayabl
e\", \"type\":\"function\"},{\"inputs\":
[{\"internalType\":\"uint256\",\"name\":\"auctionId\",\"type\":\"uint256\"},
{\"internalType\":\"uint96[]\",\"name\":\"_minBuyAmounts\",\"type\":\"uint96[]\"},
{\"internalType\":\"uint96[]\",\"name\":\"_sellAmounts\",\"type\":\"uint96[]\"},
{\"internalType\":\"bytes32[]\",\"name\":\"_prevSellOrders\",\"type\":\"bytes32[]\"},
{\"internalType\":\"bytes\",\"name\":\"allowListCallData\",\"type\":\"bytes\"},
{\"internalType\":\"address\",\"name\":\"orderSubmitter\",\"type\":\"address\"}],\"name\":\"placeS
ellOrdersOnBehalf\",\"outputs\":
[{\"internalType\":\"uint64\",\"name\":\"userId\",\"type\":\"uint64\"}],\"stateMutability\":\"nonpayabl
e\", \"type\":\"function\"},{\"inputs\":
[{\"internalType\":\"uint256\",\"name\":\"auctionId\",\"type\":\"uint256\"},
{\"internalType\":\"uint256\",\"name\":\"iterationSteps\",\"type\":\"uint256\"}],\"name\":\"precalcul
ateSellAmountSum\",\"outputs\":[],\"stateMutability\":\"nonpayable\", \"type\":\"function\"},
{\"inputs\":
[{\"internalType\":\"address\",\"name\":\"user\",\"type\":\"address\"}],\"name\":\"registerUser\", \"o
utputs\":
[{\"internalType\":\"uint64\",\"name\":\"userId\",\"type\":\"uint64\"}],\"stateMutability\":\"nonpayabl
e\", \"type\":\"function\"},{\"inputs\":[],\"name\":\"renounceOwnership\",\"outputs\":
[],\"stateMutability\":\"nonpayable\", \"type\":\"function\"},{\"inputs\":
[{\"internalType\":\"uint256\",\"name\":\"newFeeNumerator\",\"type\":\"uint256\"},
{\"internalType\":\"address\",\"name\":\"newfeeReceiverAddress\",\"type\":\"address\"}],\"name\"
\":\"setFeeParameters\",\"outputs\":[],\"stateMutability\":\"nonpayable\", \"type\":\"function\"},
```

```

{"inputs\":
[{"internalType\":"uint256\","name\":"auctionId\","type\":"uint256\"}],\"name\":"settleAuction\","outputs\":
[{"internalType\":"bytes32\","name\":"clearingOrder\","type\":"bytes32\"}],\"stateMutability\":"nonpayable\","type\":"function\"},{\"inputs\":
[{"internalType\":"uint256\","name\":"auctionId\","type\":"uint256\"},
{"internalType\":"uint96[]\","name\":"_minBuyAmount\","type\":"uint96[]\"},
{"internalType\":"uint96[]\","name\":"_sellAmount\","type\":"uint96[]\"},
{"internalType\":"bytes32[]\","name\":"_prevSellOrder\","type\":"bytes32[]\"},
{"internalType\":"bytes\","name\":"allowListCallData\","type\":"bytes\"}],\"name\":"settleAuctionAtomically\", \"outputs\":[], \"stateMutability\":"nonpayable\", \"type\":"function\"},{\"inputs\":
[{"internalType\":"address\","name\":"newOwner\","type\":"address\"}],\"name\":"transferOwnership\", \"outputs\":[], \"stateMutability\":"nonpayable\", \"type\":"function\"}],\"devdoc\":
{\"kind\":"dev\", \"methods\": {\"owner()\": {\"details\":"Returns the address of the current owner.\"}, \"renounceOwnership()\": {\"details\":"Leaves the contract without owner. It will not be possible to call `onlyOwner` functions anymore. Can only be called by the current owner. NOTE: Renouncing ownership will leave the contract without an owner, thereby removing any functionality that is only available to the owner.\"}, \"transferOwnership(address)\": {\"details\":"Transfers ownership of the contract to a new account (`newOwner`). Can only be called by the current owner.\"}}, \"version\":"1\"}, \"userdoc\": {\"kind\":"user\", \"methods\": {}, \"version\":"1\"}, \"settings\": {\"compilationTarget\": {\"contracts/EasyAuction.sol\":"EasyAuction\"}, \"evmVersion\":"istanbul\", \"libraries\": {}, \"metadata\": {\"bytecodeHash\":"ipfs\", \"useLiteralContent\":true}, \"optimizer\": {\"enabled\":false, \"runs\":200}, \"remappings\":[]}, \"sources\": {\"@openzeppelin/contracts/GSN/Context.sol\": {\"content\":"// SPDX-License-Identifier: MIT\\n\\npragma solidity >=0.6.0 <0.8.0;\\n\\n/*\\n * @dev Provides information about the current execution context, including the\\n * sender of the transaction and its data. While these are generally available\\n * via msg.sender and msg.data, they should not be accessed in such a direct\\n * manner, since when dealing with GSN meta-transactions the account sending and\\n * paying for execution may not be the actual sender (as far as an application\\n * is concerned).\\n *\\n * This contract is only required for intermediate, library-like contracts.\\n */\\nabstract contract Context {\\n function _msgSender() internal view virtual returns (address payable) {\\n return msg.sender;\\n }\\n\\n function _msgData() internal view virtual returns (bytes memory) {\\n this; // silence state mutability warning without generating bytecode - see https://github.com/ethereum/solidity/issues/2691\\n return msg.data;\\n }\\n}\\n\", \"keccak256\":"0x8d3cb350f04ff49cfb10aef08d87f19dcbaecc8027b0bed12f3275cd12f38cf0\", \"license\":"MIT\"}, \"@openzeppelin/contracts/access/Ownable.sol\": {\"content\":"// SPDX-License-Identifier: MIT\\n\\npragma solidity >=0.6.0 <0.8.0;\\n\\nimport \\\"../GSN/Context.sol\\\";\\n\\n/*\\n * @dev Contract module which provides a basic access

```

control mechanism, where

- * there is an account (an owner) that can be granted exclusive access to
- * specific functions.

By default, the owner account will be the one that deploys the contract. This can later be changed with `{transferOwnership}`. This module is used through inheritance. It will make available the modifier ``onlyOwner``, which can be applied to your functions to restrict their use to the owner.

abstract contract Ownable is Context

```

address private _owner;

event OwnershipTransferred(address indexed previousOwner, address indexed newOwner);

/**
 * @dev Initializes the contract setting the deployer as the initial owner.
 */
constructor () internal {
    address msgSender = _msgSender();
    _owner = msgSender;
    emit OwnershipTransferred(address(0), msgSender);
}

/**
 * @dev Returns the address of the current owner.
 */
function owner() public view returns (address) {
    return _owner;
}

/**
 * @dev Throws if called by any account other than the owner.
 */
modifier onlyOwner() {
    require(_owner == _msgSender(), "Ownable: caller is not the owner");
    _;
}

/**
 * @dev Leaves the contract without owner. It will not be possible to call
 * `onlyOwner` functions anymore. Can only be called by the current owner.
 * NOTE: Renouncing ownership will leave the contract without an owner,
 * thereby removing any functionality that is only available to the owner.
 */
function renounceOwnership() public virtual onlyOwner {
    emit OwnershipTransferred(_owner, address(0));
    _owner = address(0);
}

/**
 * @dev Transfers ownership of the contract to a new account
 * (`newOwner`). Can only be called by the current owner.
 */
function transferOwnership(address newOwner) public virtual onlyOwner {
    require(newOwner != address(0), "Ownable: new owner is the zero address");
    emit OwnershipTransferred(_owner, newOwner);
    _owner = newOwner;
}
}

", "keccak256": "0xf7c39c7e6d06ed3bda90cfefbcbf2ddc32c599c3d6721746546ad64946efccaa", "license": "MIT"}, {"@openzeppelin/contracts/math/Math.sol": {"content": "// SPDX-License-Identifier: MIT\n\npragma solidity >=0.6.0 <0.8.0;\n\n/**\n * @dev Standard math utilities missing in the Solidity language.\n */\nlibrary Math {\n    /**\n     * @dev Returns the largest of two numbers.\n     */\n    function max(uint256 a, uint256 b) internal pure returns (uint256) {\n        return a >= b ? a : b;\n    }\n\n    /**\n     * @dev Returns the smallest of two numbers.\n     */\n    function min(uint256 a, uint256 b) internal pure returns (uint256) {\n        return a < b ? a : b;\n    }\n\n    /**\n     * @dev Returns the average of two numbers. The result is rounded\n     * towards zero.\n     */\n    function average(uint256 a, uint256 b) internal pure returns (uint256) {\n        // (a + b) / 2 can overflow, so we distribute\n        return (a / 2) + (b / 2) + ((a % 2 + b % 2) / 2);\n    }\n}\n\n", "keccak256": "0x363bd3b45201f07c9b71c2edc96533468cf14a3d029fabd82fddceb1eb3ebd9c", "license": "MIT"}, {"@openzeppelin/contracts/math/SafeMath.sol": {"content": "// SPDX-License-Identifier: MIT\n\npragma solidity >=0.6.0 <0.8.0;\n\n/**\n * @dev Wrappers over Solidity's arithmetic operations with added overflow\n * checks.\n *\n * Arithmetic operations in Solidity wrap on overflow. This can easily result\n
```

programmers usually assume that an overflow raises an error, which is the standard behavior in high level programming languages. `SafeMath` restores this intuition by reverting the transaction when an operation overflows. Using this library instead of the unchecked operations eliminates an entire class of bugs, so it's recommended to use it always.

library SafeMath

@dev Returns the addition of two unsigned integers, reverting on overflow. Counterpart to Solidity's `+` operator.

Requirements:

- Addition cannot overflow.

function `add(uint256 a, uint256 b)` **internal pure** returns (uint256) {
 uint256 c = a + b;
 require(c >= a, "SafeMath: addition overflow");
 return c;
}

@dev Returns the subtraction of two unsigned integers, reverting on overflow (when the result is negative). Counterpart to Solidity's `-` operator.

Requirements:

- Subtraction cannot overflow.

function `sub(uint256 a, uint256 b)` **internal pure** returns (uint256) {
 return sub(a, b, "SafeMath: subtraction overflow");
}

@dev Returns the subtraction of two unsigned integers, reverting with custom message on overflow (when the result is negative). Counterpart to Solidity's `-` operator.

Requirements:

- Subtraction cannot overflow.

function `sub(uint256 a, uint256 b, string memory errorMessage)` **internal pure** returns (uint256) {
 require(b <= a, errorMessage);
 uint256 c = a - b;
 return c;
}

@dev Returns the multiplication of two unsigned integers, reverting on overflow. Counterpart to Solidity's `\*` operator.

Requirements:

- Multiplication cannot overflow.

function `mul(uint256 a, uint256 b)` **internal pure** returns (uint256) {
 // Gas optimization: this is cheaper than requiring 'a' not being zero, but the benefit is lost if 'b' is also tested.
 // See: <https://github.com/OpenZeppelin/openzeppelin-contracts/pull/522>
 if (a == 0) {
 return 0;
 }
 uint256 c = a * b;
 require(c / a == b, "SafeMath: multiplication overflow");
 return c;
}

@dev Returns the integer division of two unsigned integers. Reverts on division by zero. The result is rounded towards zero. Counterpart to Solidity's `/` operator. Note: this function uses a `revert` opcode (which leaves remaining gas untouched) while Solidity uses an invalid opcode to revert (consuming all remaining gas).

Requirements:

- The divisor cannot be zero.

function `div(uint256 a, uint256 b)` **internal pure** returns (uint256) {
 return div(a, b, "SafeMath: division by zero");
}

@dev Returns the integer division of two unsigned integers. Reverts with custom message on division by zero. The result is rounded towards zero. Counterpart to Solidity's `/` operator. Note: this function uses a `revert` opcode (which leaves remaining gas untouched) while Solidity uses an invalid opcode to revert (consuming all remaining gas).

Requirements:

- The divisor cannot be zero.

function `div(uint256 a, uint256 b, string memory errorMessage)` **internal pure** returns (uint256) {
 require(b > 0, errorMessage);
 uint256 c = a / b;
 // assert(a == b * c + a % b);
 // There is no case in which this doesn't hold
 return c;
}

@dev Returns the remainder of dividing two unsigned integers. (unsigned integer modulo), Reverts when

dividing by zero.

* Counterpart to Solidity's `%` operator. This function uses a `revert` opcode (which leaves remaining gas untouched) while Solidity uses an invalid opcode to revert (consuming all remaining gas).

* Requirements:

- The divisor cannot be zero.

function mod(uint256 a, uint256 b) internal pure returns (uint256) {
 return mod(a, b, "SafeMath: modulo by zero");
}

/**
 * @dev Returns the remainder of dividing two unsigned integers. (unsigned integer modulo),
 * Reverts with custom message when dividing by zero.

* Counterpart to Solidity's `%` operator. This function uses a `revert` opcode (which leaves remaining gas untouched) while Solidity uses an invalid opcode to revert (consuming all remaining gas).

* Requirements:

- The divisor cannot be zero.

function mod(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
 require(b != 0, errorMessage);
 return a % b;
}

,"keccak256":"0x3b21f2c8d626de3b9925ae33e972d8bf5c8b1bffb3f4ee94daeed7d0679036e6","license":"MIT"},"@openzeppelin/contracts/token/ERC20/IERC20.sol":
{"content":"// SPDX-License-Identifier: MIT\n\npragma solidity >=0.6.0 <0.8.0;\n\n/**
 * @dev Interface of the ERC20 standard as defined in the EIP.\n\ninterface IERC20 {
 * @dev Returns the amount of tokens in existence.\n\nfunction totalSupply() external view returns (uint256);\n\n * @dev Returns the amount of tokens owned by `account`.\n\nfunction balanceOf(address account) external view returns (uint256);\n\n * @dev Moves `amount` tokens from the caller's account to `recipient`.\n\n * Returns a boolean value indicating whether the operation succeeded.\n\n * Emits a {Transfer} event.\n\nfunction transfer(address recipient, uint256 amount) external returns (bool);\n\n * @dev Returns the remaining number of tokens that `spender` will be
 * allowed to spend on behalf of `owner` through {transferFrom}. This is
 * zero by default.\n\n * This value changes when {approve} or {transferFrom} are called.\n\nfunction allowance(address owner, address spender) external view returns (uint256);\n\n * @dev Sets `amount` as the allowance of `spender` over the caller's tokens.
 * Returns a boolean value indicating whether the operation succeeded.
 * IMPORTANT: Beware that changing an allowance with this method brings the risk
 * that someone may use both the old and the new allowance by unfortunate
 * transaction ordering. One possible solution to mitigate this race
 * condition is to first reduce the spender's allowance to 0 and set the
 * desired value afterwards:\n\n * <https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729>
 * Emits an {Approval} event.\n\nfunction approve(address spender, uint256 amount) external returns (bool);\n\n * @dev Moves `amount` tokens from `sender` to `recipient` using the
 * allowance mechanism. `amount` is then deducted from the caller's
 * allowance.
 * Returns a boolean value indicating whether the operation succeeded.
 * Emits a {Transfer} event.\n\nfunction transferFrom(address sender, address recipient, uint256 amount) external returns (bool);\n\n * @dev Emitted when `value` tokens are moved from one account (`from`) to
 * another (`to`).
 * Note that `value` may be zero.
 * event Transfer(address indexed from, address indexed to, uint256 value);\n\n * @dev

Emitted when the allowance of a `spender` for an `owner` is set by
`value` is the new allowance.
`\*` event Approval(address indexed owner, address indexed spender, uint256 value);
`\*` license: "MIT",
`\*` SPDX-License-Identifier: MIT
`\*` pragma solidity >=0.6.0 <0.8.0;
`\*` import "IERC20.sol";
`\*` import "math/SafeMath.sol";
`\*` import "utils/Address.sol";
`\*` @title SafeERC20
`\*` @dev Wrappers around ERC20 operations that throw on failure (when the token contract returns false). Tokens that return no value (and instead revert or throw on failure) are also supported, non-reverting calls are assumed to be successful.
`\*` To use this library you can add a `using SafeERC20 for IERC20;` statement to your contract, which allows you to call the safe operations as `token.safeTransfer(...)`, etc.
`\*` library SafeERC20 {
`\*` using SafeMath for uint256;
`\*` using Address for address;
`\*` function safeTransfer(IERC20 token, address to, uint256 value) internal {
`\*` _callOptionalReturn(token, abi.encodeWithSelector(token.transfer.selector, to, value));
`\*` }
`\*` function safeTransferFrom(IERC20 token, address from, address to, uint256 value) internal {
`\*` _callOptionalReturn(token, abi.encodeWithSelector(token.transferFrom.selector, from, to, value));
`\*` }
`\*` /**
`\*` @dev Deprecated. This function has issues similar to the ones found in {IERC20-approve}, and its usage is discouraged.
`\*` Whenever possible, use {safeIncreaseAllowance} and {safeDecreaseAllowance} instead.
`\*` function safeApprove(IERC20 token, address spender, uint256 value) internal {
`\*` // safeApprove should only be called when setting an initial allowance, or when resetting it to zero. To increase and decrease it, use
`\*` 'safeIncreaseAllowance' and 'safeDecreaseAllowance'
`\*` // solhint-disable-next-line max-line-length
`\*` require((value == 0) || (token.allowance(address(this), spender) == 0),
`\*` "SafeERC20: approve from non-zero to non-zero allowance");
`\*` _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender, value));
`\*` }
`\*` function safeIncreaseAllowance(IERC20 token, address spender, uint256 value) internal {
`\*` uint256 newAllowance = token.allowance(address(this), spender).add(value);
`\*` _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender, newAllowance));
`\*` }
`\*` function safeDecreaseAllowance(IERC20 token, address spender, uint256 value) internal {
`\*` uint256 newAllowance = token.allowance(address(this), spender).sub(value, "SafeERC20: decreased allowance below zero");
`\*` _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender, newAllowance));
`\*` }
`\*` /**
`\*` @dev Imitates a Solidity high-level call (i.e. a regular function call to a contract), relaxing the requirement on the return value: the return value is optional (but if data is returned, it must not be false).
`\*` @param token The token targeted by the call.
`\*` @param data The call data (encoded using abi.encode or one of its variants).
`\*` function

```

_callOptionalReturn(IERC20 token, bytes memory data) private {
    // We need to perform a low level call here, to bypass Solidity's return data size checking mechanism, since
    // we're implementing it ourselves. We use {Address.functionCall} to perform this call, which verifies
    // that the target address contains contract code and also asserts for success in the low-level call.
    bytes memory returndata = address(token).functionCall(data, "SafeERC20: low-level call failed");
    if (returndata.length > 0) { // Return data is optional
        // solhint-disable-next-line max-line-length
        require(abi.decode(returndata, (bool)), "SafeERC20: ERC20 operation did not succeed");
    }
}

}

", "keccak256":"0xf12dfbe97e6276980b83d2830bb0eb75e0cf4f3e626c2471137f82158ae6a0fc", "license":"MIT"}, "@openzeppelin/contracts/utils/Address.sol":{"content":"//
SPDX-License-Identifier: MIT
pragma solidity >=0.6.2 <0.8.0;

/**
 * @dev Collection of functions related to the address type
 */
library Address {
    /**
     * @dev Returns true if `account` is a contract.
     *
     * [IMPORTANT]
     * It is unsafe to assume that an address for which this function returns
     * false is an externally-owned account (EOA) and not a contract.
     *
     * Among others, `isContract` will return false for the following
     * types of addresses:
     *
     * - an externally-owned account
     * - a contract in construction
     * - an address where a contract will be created
     * - an address where a contract lived, but was destroyed
     *
     */
    function isContract(address account) internal view returns (bool)
    // This method relies on extcodesize, which returns 0 for contracts in
    // construction, since the code is only stored at the end of the
    // constructor execution.
    uint256 size;
    // solhint-disable-next-line no-inline-assembly
    assembly { size := extcodesize(account) }
    return size > 0;
}

/**
 * @dev Replacement for Solidity's `transfer`: sends `amount` wei to
 * `recipient`, forwarding all available gas and reverting on errors.
 *
 * https://eips.ethereum.org/EIPS/eip-1884[EIP1884] increases the gas cost
 * of certain opcodes, possibly making contracts go over the 2300 gas limit
 * imposed by `transfer`, making them unable to receive funds via
 * `transfer`. {sendValue} removes this limitation.
 *
 * https://diligence.consensys.net/posts/2019/09/stop-using-soliditys-transfer-now/[Learn more].
 *
 * IMPORTANT: because control is transferred to `recipient`, care must be
 * taken to not create reentrancy vulnerabilities. Consider using
 * {ReentrancyGuard} or the
 * https://solidity.readthedocs.io/en/v0.5.11/security-considerations.html#use-the-checks-effects-interactions-pattern[checks-effects-interactions pattern].
 */
function sendValue(address payable recipient, uint256 amount) internal {
    require(address(this).balance >= amount, "Address: insufficient balance");
    // solhint-disable-next-line avoid-low-level-calls, avoid-call-value
    (bool success, ) = recipient.call{value: amount}("");
    require(success, "Address: unable to send value, recipient may have reverted");
}

/**
 * @dev Performs a Solidity function call using a low level `call`. A
 * plain `call` is an unsafe replacement for a function call: use this
 * function instead.
 *
 * If `target` reverts with a revert reason, it is bubbled up by this
 * function (like regular Solidity function calls).
 *
 * Returns the raw returned data. To convert to the

```

```

expected return value,\n * use https://solidity.readthedocs.io/en/latest/units-and-global-
variables.html?highlight=abi.decode#abi-encoding-and-decoding-functions[abi.decode]`.\n
*\n * Requirements:\n * - `target` must be a contract.\n * - calling `target` with `data` must
not revert.\n * - _Available since v3.1._\n */\n function functionCall(address target, bytes
memory data) internal returns (bytes memory) {\n return functionCall(target, data, \\\"Address:
low-level call failed\\\");\n }\n\n /**\n * @dev Same as {xref-Address-functionCall-address-
bytes-}[functionCall], but with\n * `errorMessage` as a fallback revert reason when `target`
reverts.\n * - _Available since v3.1._\n */\n function functionCall(address target, bytes
memory data, string memory errorMessage) internal returns (bytes memory) {\n return
functionCallWithValue(target, data, 0, errorMessage);\n }\n\n /**\n * @dev Same as {xref-
Address-functionCall-address-bytes-}[functionCall],\n * but also transferring `value` wei to
`target`.\n * - _Available since v3.1._\n * - the calling contract must have an ETH balance of at
least `value`.\n * - the called Solidity function must be `payable`.\n * - _Available since
v3.1._\n */\n function functionCallWithValue(address target, bytes memory data, uint256
value) internal returns (bytes memory) {\n return functionCallWithValue(target, data, value,
\\\"Address: low-level call with value failed\\\");\n }\n\n /**\n * @dev Same as {xref-Address-
functionCallWithValue-address-bytes-uint256-}[functionCallWithValue], but\n * with
`errorMessage` as a fallback revert reason when `target` reverts.\n * - _Available since
v3.1._\n */\n function functionCallWithValue(address target, bytes memory data, uint256
value, string memory errorMessage) internal returns (bytes memory) {\n
require(address(this).balance >= value, \\\"Address: insufficient balance for call\\\");\n
require(isContract(target), \\\"Address: call to non-contract\\\");\n\n // solhint-disable-next-line
avoid-low-level-calls\n (bool success, bytes memory returndata) = target.call{ value: value }
(data);\n return _verifyCallResult(success, returndata, errorMessage);\n }\n\n /**\n * @dev
Same as {xref-Address-functionCall-address-bytes-}[functionCall],\n * but performing a static
call.\n * - _Available since v3.3._\n */\n function functionStaticCall(address target, bytes
memory data) internal view returns (bytes memory) {\n return functionStaticCall(target, data,
\\\"Address: low-level static call failed\\\");\n }\n\n /**\n * @dev Same as {xref-Address-
functionCall-address-bytes-string-}[functionCall],\n * but performing a static call.\n * -
_Available since v3.3._\n */\n function functionStaticCall(address target, bytes memory data,
string memory errorMessage) internal view returns (bytes memory) {\n
require(isContract(target), \\\"Address: static call to non-contract\\\");\n\n // solhint-disable-
next-line avoid-low-level-calls\n (bool success, bytes memory returndata) =
target.staticcall(data);\n return _verifyCallResult(success, returndata, errorMessage);\n }\n\n
function _verifyCallResult(bool success, bytes memory returndata, string memory
errorMessage) private pure returns (bytes memory) {\n if (success) {\n return returndata;\n }
else {\n // Look for revert reason and bubble it up if present\n if (returndata.length > 0) {\n //
The easiest way to bubble the revert reason is using memory via assembly\n\n // solhint-
disable-next-line no-inline-assembly\n assembly {\n let returndata_size :=

```

```

mload(returndata))\n revert(add(32, returndata), returndata_size)\n }\n } else {\n
revert(errorMessage);\n }\n }\n
}\n}\n\n","keccak256":"0xa6a15dddbcf29d2922a1e0d4151b5d2d33da24b93cc9ebc12390e
0d855532f8","license":"MIT"},"contracts/EasyAuction.sol":{"content":"pragma solidity
>=0.6.8;\n\nimport \"@openzeppelin/contracts/token/ERC20/IERC20.sol\";\n\nimport
\"@openzeppelin/contracts/token/ERC20/SafeERC20.sol\";\n\nimport
\"./libraries/IterableOrderedOrderSet.sol\";\n\nimport
\"@openzeppelin/contracts/math/Math.sol\";\n\nimport
\"@openzeppelin/contracts/math/SafeMath.sol\";\n\nimport
\"./libraries/IdToAddressBiMap.sol\";\n\nimport \"./libraries/SafeCast.sol\";\n\nimport
\"@openzeppelin/contracts/access/Ownable.sol\";\n\nimport
\"./interfaces/AllowListVerifier.sol\";\n\ncontract EasyAuction is Ownable {\n\n    using
SafeERC20 for IERC20;\n\n    using SafeMath for uint64;\n\n    using SafeMath for uint96;\n\n    using
SafeMath for uint256;\n\n    using SafeCast for uint256;\n\n    using
IterableOrderedOrderSet for IterableOrderedOrderSet.Data;\n\n    using
IterableOrderedOrderSet for bytes32;\n\n    using IdToAddressBiMap for
IdToAddressBiMap.Data;\n\n\n    modifier atStageOrderPlacement(uint256 auctionId) {\n\n        require(\n\n            block.timestamp < auctionData[auctionId].auctionEndDate,\n\n            \"no longer in
order placement phase\"\n\n        );\n\n        _;\n\n    }\n\n    modifier
atStageOrderPlacementAndCancelation(uint256 auctionId) {\n\n        require(\n\n            block.timestamp < auctionData[auctionId].orderCancellationEndDate,\n\n            \"no longer in
order placement and cancelation phase\"\n\n        );\n\n        _;\n\n    }\n\n    modifier
atStageSolutionSubmission(uint256 auctionId) {\n\n        {\n\n            uint256 auctionEndDate =
auctionData[auctionId].auctionEndDate;\n\n            require(\n\n                auctionEndDate != 0 &&\n\n                block.timestamp >= auctionEndDate &&\n\n                auctionData[auctionId].clearingPriceOrder ==
bytes32(0),\n\n                \"Auction not in solution submission phase\"\n\n            );\n\n        }\n\n        _;\n\n    }\n\n    modifier atStageFinished(uint256 auctionId) {\n\n        require(\n\n            auctionData[auctionId].clearingPriceOrder != bytes32(0),\n\n            \"Auction not yet
finished\"\n\n        );\n\n        _;\n\n    }\n\n    event NewSellOrder(\n\n        uint256 indexed
auctionId,\n\n        uint64 indexed userId,\n\n        uint96 buyAmount,\n\n        uint96 sellAmount\n\n    );\n\n    event CancellationSellOrder(\n\n        uint256 indexed auctionId,\n\n        uint64 indexed
userId,\n\n        uint96 buyAmount,\n\n        uint96 sellAmount\n\n    );\n\n    event
ClaimedFromOrder(\n\n        uint256 indexed auctionId,\n\n        uint64 indexed userId,\n\n        uint96
buyAmount,\n\n        uint96 sellAmount\n\n    );\n\n    event NewUser(uint64 indexed userId, address
indexed userAddress);\n\n    event NewAuction(\n\n        uint256 indexed auctionId,\n\n        IERC20
indexed _auctioningToken,\n\n        IERC20 indexed _biddingToken,\n\n        uint256
orderCancellationEndDate,\n\n        uint256 auctionEndDate,\n\n        uint64 userId,\n\n        uint96
_auctionedSellAmount,\n\n        uint96 _minBuyAmount,\n\n        uint256
minimumBiddingAmountPerOrder,\n\n        uint256 minFundingThreshold,\n\n        address

```

```

allowListContract,\\r\\n bytes allowListData\\r\\n );\\r\\n event AuctionCleared(\\r\\n uint256
indexed auctionId,\\r\\n uint96 soldAuctioningTokens,\\r\\n uint96 soldBiddingTokens,\\r\\n
bytes32 clearingPriceOrder\\r\\n );\\r\\n event UserRegistration(address indexed user, uint64
userId);\\r\\n\\r\\n struct AuctionData {\\r\\n IERC20 auctioningToken;\\r\\n IERC20
biddingToken;\\r\\n uint256 orderCancellationEndDate;\\r\\n uint256 auctionEndDate;\\r\\n
bytes32 initialAuctionOrder;\\r\\n uint256 minimumBiddingAmountPerOrder;\\r\\n uint256
interimSumBidAmount;\\r\\n bytes32 interimOrder;\\r\\n bytes32 clearingPriceOrder;\\r\\n uint96
volumeClearingPriceOrder;\\r\\n bool minFundingThresholdNotReached;\\r\\n bool
isAtomicClosureAllowed;\\r\\n uint256 feeNumerator;\\r\\n uint256 minFundingThreshold;\\r\\n
}\\r\\n mapping(uint256 => IterableOrderedOrderSet.Data) internal sellOrders;\\r\\n
mapping(uint256 => AuctionData) public auctionData;\\r\\n mapping(uint256 => address)
public auctionAccessManager;\\r\\n mapping(uint256 => bytes) public
auctionAccessData;\\r\\n\\r\\n IdToAddressBiMap.Data private registeredUsers;\\r\\n uint64
public numUsers;\\r\\n uint256 public auctionCounter;\\r\\n\\r\\n constructor() public Ownable()
{\\r\\n\\r\\n uint256 public feeNumerator = 0;\\r\\n uint256 public constant
FEE_DENOMINATOR = 1000;\\r\\n uint64 public feeReceiverUserId = 1;\\r\\n\\r\\n function
setFeeParameters(\\r\\n uint256 newFeeNumerator,\\r\\n address
newfeeReceiverAddress\\r\\n ) public onlyOwner() {\\r\\n require(\\r\\n newFeeNumerator <=
15,\\r\\n \\r\\n \\r\\n"Fee is not allowed to be set higher than 1.5%\\r\\n"\\r\\n );\\r\\n // caution: for currently
running auctions, the feeReceiverUserId is changing as well.\\r\\n feeReceiverUserId =
getUserId(newfeeReceiverAddress);\\r\\n feeNumerator = newFeeNumerator;\\r\\n }\\r\\n\\r\\n //
@dev: function to initiate a new auction\\r\\n // Warning: In case the auction is expected to raise
more than\\r\\n // 2^96 units of the biddingToken, don't start the auction, as\\r\\n // it will not be
setttable. This corresponds to about 79\\r\\n // billion DAI.\\r\\n //\\r\\n // Prices between
biddingToken and auctioningToken are expressed by a\\r\\n // fraction whose components are
stored as uint96.\\r\\n function initiateAuction(\\r\\n IERC20 _auctioningToken,\\r\\n IERC20
_biddingToken,\\r\\n uint256 orderCancellationEndDate,\\r\\n uint256 auctionEndDate,\\r\\n
uint96 _auctionedSellAmount,\\r\\n uint96 _minBuyAmount,\\r\\n uint256
minimumBiddingAmountPerOrder,\\r\\n uint256 minFundingThreshold,\\r\\n bool
isAtomicClosureAllowed,\\r\\n address accessManagerContract,\\r\\n bytes memory
accessManagerContractData\\r\\n ) public returns (uint256) {\\r\\n // withdraws sellAmount +
fees\\r\\n _auctioningToken.safeTransferFrom(\\r\\n msg.sender,\\r\\n address(this),\\r\\n
_auctionedSellAmount.mul(FEE_DENOMINATOR.add(feeNumerator)).div(\\r\\n
FEE_DENOMINATOR\\r\\n ) //[0]\\r\\n );\\r\\n require(_auctionedSellAmount > 0, \\r\\n"cannot
auction zero tokens\\r\\n");\\r\\n require(_minBuyAmount > 0, \\r\\n"tokens cannot be auctioned for
free\\r\\n");\\r\\n require(\\r\\n minimumBiddingAmountPerOrder > 0,\\r\\n
\\r\\n"minimumBiddingAmountPerOrder is not allowed to be zero\\r\\n"\\r\\n );\\r\\n require(\\r\\n
orderCancellationEndDate <= auctionEndDate,\\r\\n \\r\\n"time periods are not configured
correctly\\r\\n"\\r\\n );\\r\\n require(\\r\\n auctionEndDate > block.timestamp,\\r\\n \\r\\n"auction end

```

```

date must be in the future\\r\\n );\\r\\n auctionCounter = auctionCounter.add(1);\\r\\n
sellOrders[auctionCounter].initializeEmptyList();\\r\\n uint64 userId =
getUserId(msg.sender);\\r\\n auctionData[auctionCounter] = AuctionData(\\r\\n
_auctioningToken,\\r\\n _biddingToken,\\r\\n orderCancellationEndDate,\\r\\n
auctionEndDate,\\r\\n IterableOrderedOrderSet.encodeOrder(\\r\\n userId,\\r\\n
_minBuyAmount,\\r\\n _auctionedSellAmount\\r\\n ),\\r\\n
minimumBiddingAmountPerOrder,\\r\\n 0,\\r\\n IterableOrderedOrderSet.QUEUE_START,\\r\\n
bytes32(0),\\r\\n 0,\\r\\n false,\\r\\n isAtomicClosureAllowed,\\r\\n feeNumerator,\\r\\n
minFundingThreshold\\r\\n );\\r\\n auctionAccessManager[auctionCounter] =
accessManagerContract;\\r\\n auctionAccessData[auctionCounter] =
accessManagerContractData;\\r\\n emit NewAuction(\\r\\n auctionCounter,\\r\\n
_auctioningToken,\\r\\n _biddingToken,\\r\\n orderCancellationEndDate,\\r\\n
auctionEndDate,\\r\\n userId,\\r\\n _auctionedSellAmount,\\r\\n _minBuyAmount,\\r\\n
minimumBiddingAmountPerOrder,\\r\\n minFundingThreshold,\\r\\n
accessManagerContract,\\r\\n accessManagerContractData\\r\\n );\\r\\n return
auctionCounter;\\r\\n }\\r\\n\\r\\n function placeSellOrders(\\r\\n uint256 auctionId,\\r\\n uint96[]
memory _minBuyAmounts,\\r\\n uint96[] memory _sellAmounts,\\r\\n bytes32[] memory
_prevSellOrders,\\r\\n bytes calldata allowListCallData\\r\\n ) external
atStageOrderPlacement(auctionId) returns (uint64 userId) {\\r\\n return\\r\\n
_placeSellOrders(\\r\\n auctionId,\\r\\n _minBuyAmounts,\\r\\n _sellAmounts,\\r\\n
_prevSellOrders,\\r\\n allowListCallData,\\r\\n msg.sender\\r\\n );\\r\\n }\\r\\n\\r\\n function
placeSellOrdersOnBehalf(\\r\\n uint256 auctionId,\\r\\n uint96[] memory _minBuyAmounts,\\r\\n
uint96[] memory _sellAmounts,\\r\\n bytes32[] memory _prevSellOrders,\\r\\n bytes calldata
allowListCallData,\\r\\n address orderSubmitter\\r\\n ) external
atStageOrderPlacement(auctionId) returns (uint64 userId) {\\r\\n return\\r\\n
_placeSellOrders(\\r\\n auctionId,\\r\\n _minBuyAmounts,\\r\\n _sellAmounts,\\r\\n
_prevSellOrders,\\r\\n allowListCallData,\\r\\n orderSubmitter\\r\\n );\\r\\n }\\r\\n\\r\\n function
_placeSellOrders(\\r\\n uint256 auctionId,\\r\\n uint96[] memory _minBuyAmounts,\\r\\n uint96[]
memory _sellAmounts,\\r\\n bytes32[] memory _prevSellOrders,\\r\\n bytes calldata
allowListCallData,\\r\\n address orderSubmitter\\r\\n ) internal returns (uint64 userId) {\\r\\n
\\r\\n address allowListManger = auctionAccessManager[auctionId];\\r\\n if (allowListManger
!= address(0)) {\\r\\n require(\\r\\n AllowListVerifier(allowListManger).isAllowed(\\r\\n
orderSubmitter,\\r\\n auctionId,\\r\\n allowListCallData\\r\\n ) ==
AllowListVerifierHelper.MAGICVALUE,\\r\\n \\r\\n"user not allowed to place order\\r\\n"\\r\\n );\\r\\n
}\\r\\n }\\r\\n {\\r\\n (\\r\\n ,\\r\\n uint96 buyAmountOfInitialAuctionOrder,\\r\\n uint96
sellAmountOfInitialAuctionOrder\\r\\n ) =
auctionData[auctionId].initialAuctionOrder.decodeOrder();\\r\\n for (uint256 i = 0; i <
_minBuyAmounts.length; i++) {\\r\\n require(\\r\\n
_minBuyAmounts[i].mul(buyAmountOfInitialAuctionOrder) <\\r\\n

```

```
sellAmountOfInitialAuctionOrder.mul(_sellAmounts[i]),\r\n\r\n limit price not better than  
mimimal offer""\r\n\r\n );\r\n\r\n }\r\n\r\n } \r\n\r\n uint256 sumOfSellAmounts = 0;\r\n\r\n userId =  
getUserId(orderSubmitter);\r\n\r\n uint256 minimumBiddingAmountPerOrder =\r\n\r\n auctionData[auctionId].minimumBiddingAmountPerOrder;\r\n\r\n for (uint256 i = 0; i <  
_minBuyAmounts.length; i++) {\r\n\r\n require(\r\n\r\n _minBuyAmounts[i] > 0,\r\n\r\n ""_minBuyAmounts must be greater than 0""\r\n\r\n );\r\n\r\n // orders should have a minimum bid  
size in order to limit the gas\r\n\r\n // required to compute the final price of the auction.\r\n\r\n require(\r\n\r\n _sellAmounts[i] > minimumBiddingAmountPerOrder,\r\n\r\n ""order too  
small""\r\n\r\n );\r\n\r\n if (\r\n\r\n sellOrders[auctionId].insert(\r\n\r\n IterableOrderedOrderSet.encodeOrder(\r\n\r\n userId,\r\n\r\n _minBuyAmounts[i],\r\n\r\n _sellAmounts[i]\r\n\r\n ),\r\n\r\n _prevSellOrders[i]\r\n\r\n ) \r\n\r\n ) {\r\n\r\n sumOfSellAmounts =  
sumOfSellAmounts.add(_sellAmounts[i]);\r\n\r\n emit NewSellOrder(\r\n\r\n auctionId,\r\n\r\n userId,\r\n\r\n _minBuyAmounts[i],\r\n\r\n _sellAmounts[i]\r\n\r\n );\r\n\r\n }\r\n\r\n }\r\n\r\n auctionData[auctionId].biddingToken.safeTransferFrom(\r\n\r\n msg.sender,\r\n\r\n address(this),\r\n\r\n sumOfSellAmounts\r\n\r\n ); //[1]\r\n\r\n }\r\n\r\n\r\n function  
cancelSellOrders(uint256 auctionId, bytes32[] memory _sellOrders)\r\n\r\n public\r\n\r\n atStageOrderPlacementAndCancelation(auctionId)\r\n\r\n {\r\n\r\n uint64 userId =  
getUserId(msg.sender);\r\n\r\n uint256 claimableAmount = 0;\r\n\r\n for (uint256 i = 0; i <  
_sellOrders.length; i++) {\r\n\r\n // Note: we keep the back pointer of the deleted element so  
that\r\n\r\n // it can be used as a reference point to insert a new node.\r\n\r\n bool success =\r\n\r\n sellOrders[auctionId].removeKeepHistory(_sellOrders[i]);\r\n\r\n if (success) {\r\n\r\n (\r\n\r\n uint64  
userIdOfIter,\r\n\r\n uint96 buyAmountOfIter,\r\n\r\n uint96 sellAmountOfIter\r\n\r\n ) =  
_sellOrders[i].decodeOrder();\r\n\r\n require(\r\n\r\n userIdOfIter == userId,\r\n\r\n ""Only the user  
can cancel his orders""\r\n\r\n );\r\n\r\n claimableAmount =  
claimableAmount.add(sellAmountOfIter);\r\n\r\n emit CancellationSellOrder(\r\n\r\n auctionId,\r\n\r\n userId,\r\n\r\n buyAmountOfIter,\r\n\r\n sellAmountOfIter\r\n\r\n );\r\n\r\n }\r\n\r\n }\r\n\r\n auctionData[auctionId].biddingToken.safeTransfer(\r\n\r\n msg.sender,\r\n\r\n claimableAmount\r\n\r\n ); //[2]\r\n\r\n }\r\n\r\n\r\n function precalculateSellAmountSum(\r\n\r\n uint256  
auctionId,\r\n\r\n uint256 iterationSteps\r\n\r\n ) public atStageSolutionSubmission(auctionId) {\r\n\r\n ( , uint96 auctioneerSellAmount) =\r\n\r\n auctionData[auctionId].initialAuctionOrder.decodeOrder();\r\n\r\n uint256 sumBidAmount =  
auctionData[auctionId].interimSumBidAmount;\r\n\r\n bytes32 iterOrder =  
auctionData[auctionId].interimOrder;\r\n\r\n\r\n for (uint256 i = 0; i < iterationSteps; i++) {\r\n\r\n iterOrder = sellOrders[auctionId].next(iterOrder);\r\n\r\n ( , uint96 sellAmountOfIter) =  
iterOrder.decodeOrder();\r\n\r\n sumBidAmount = sumBidAmount.add(sellAmountOfIter);\r\n\r\n }\r\n\r\n\r\n require(\r\n\r\n iterOrder != IterableOrderedOrderSet.QUEUE_END,\r\n\r\n ""reached  
end of order list""\r\n\r\n );\r\n\r\n\r\n // it is checked that not too many iteration steps were  
taken:\r\n\r\n // require that the sum of SellAmounts times the price of the last order\r\n\r\n // is not  
more than initially sold amount\r\n\r\n ( , uint96 buyAmountOfIter, uint96 sellAmountOfIter) =\r\n\r\n
```

```

iterOrder.decodeOrder();\r\n\r\n require(\r\n\r\n sumBidAmount.mul(buyAmountOfilter) <\r\n\r\n
auctioneerSellAmount.mul(sellAmountOfilter),\r\n\r\n \r\n\r\n"too many orders summed up\r\n\r\n"
);\r\n\r\n\r\n\r\n auctionData[auctionId].interimSumBidAmount = sumBidAmount;\r\n\r\n
auctionData[auctionId].interimOrder = iterOrder;\r\n\r\n }\r\n\r\n\r\n\r\n function
settleAuctionAtomically(\r\n\r\n uint256 auctionId,\r\n\r\n uint96[] memory _minBuyAmount,\r\n\r\n
uint96[] memory _sellAmount,\r\n\r\n bytes32[] memory _prevSellOrder,\r\n\r\n bytes calldata
allowListCallData\r\n\r\n ) public atStageSolutionSubmission(auctionId) {\r\n\r\n require(\r\n\r\n
auctionData[auctionId].isAtomicClosureAllowed,\r\n\r\n \r\n\r\n"not allowed to settle auction
atomically\r\n\r\n" );\r\n\r\n require(\r\n\r\n _minBuyAmount.length == 1 && _sellAmount.length ==
1,\r\n\r\n \r\n\r\n"Only one order can be placed atomically\r\n\r\n" );\r\n\r\n uint64 userId =
getUserId(msg.sender);\r\n\r\n require(\r\n\r\n
auctionData[auctionId].interimOrder.smallerThan(\r\n\r\n
IterableOrderedOrderSet.encodeOrder(\r\n\r\n userId,\r\n\r\n _minBuyAmount[0],\r\n\r\n
_sellAmount[0]\r\n\r\n )\r\n\r\n ),\r\n\r\n \r\n\r\n"precalculateSellAmountSum is already too
advanced\r\n\r\n" );\r\n\r\n _placeSellOrders(\r\n\r\n auctionId,\r\n\r\n _minBuyAmount,\r\n\r\n
_sellAmount,\r\n\r\n _prevSellOrder,\r\n\r\n allowListCallData,\r\n\r\n msg.sender\r\n\r\n );\r\n\r\n
settleAuction(auctionId);\r\n\r\n }\r\n\r\n\r\n\r\n // @dev function settling the auction and calculating
the price\r\n\r\n function settleAuction(uint256 auctionId)\r\n\r\n public\r\n\r\n
atStageSolutionSubmission(auctionId)\r\n\r\n returns (bytes32 clearingOrder)\r\n\r\n {\r\n\r\n (\r\n\r\n
uint64 auctioneerId,\r\n\r\n uint96 minAuctionedBuyAmount,\r\n\r\n uint96
fullAuctionedAmount\r\n\r\n ) =
auctionData[auctionId].initialAuctionOrder.decodeOrder();\r\n\r\n\r\n\r\n uint256 currentBidSum =
auctionData[auctionId].interimSumBidAmount;\r\n\r\n bytes32 currentOrder =
auctionData[auctionId].interimOrder;\r\n\r\n uint256 buyAmountOfilter;\r\n\r\n uint256
sellAmountOfilter;\r\n\r\n uint96 fillVolumeOfAuctioneerOrder = fullAuctionedAmount;\r\n\r\n // Sum
order up, until fullAuctionedAmount is fully bought or queue end is reached\r\n\r\n do {\r\n\r\n
bytes32 nextOrder = sellOrders[auctionId].next(currentOrder);\r\n\r\n if (nextOrder ==
IterableOrderedOrderSet.QUEUE_END) {\r\n\r\n break;\r\n\r\n }\r\n\r\n currentOrder =
nextOrder;\r\n\r\n (, buyAmountOfilter, sellAmountOfilter) = currentOrder.decodeOrder();\r\n\r\n
currentBidSum = currentBidSum.add(sellAmountOfilter);\r\n\r\n } while (\r\n\r\n
currentBidSum.mul(buyAmountOfilter) <\r\n\r\n fullAuctionedAmount.mul(sellAmountOfilter)\r\n\r\n
);\r\n\r\n\r\n\r\n if (\r\n\r\n currentBidSum > 0 &&\r\n\r\n currentBidSum.mul(buyAmountOfilter) >=\r\n\r\n
fullAuctionedAmount.mul(sellAmountOfilter)\r\n\r\n ) {\r\n\r\n // All considered/summed orders are
sufficient to close the auction fully\r\n\r\n // at price between current and previous orders.\r\n\r\n
uint256 uncoveredBids =\r\n\r\n currentBidSum.sub(\r\n\r\n
fullAuctionedAmount.mul(sellAmountOfilter).div(\r\n\r\n buyAmountOfilter\r\n\r\n )\r\n\r\n );\r\n\r\n\r\n\r\n if
(sellAmountOfilter >= uncoveredBids) {\r\n\r\n // [13]\r\n\r\n // Auction fully filled via partial match of
currentOrder\r\n\r\n uint256 sellAmountClearingOrder =\r\n\r\n
sellAmountOfilter.sub(uncoveredBids);\r\n\r\n auctionData[auctionId]\r\n\r\n

```

```

.volumeClearingPriceOrder = sellAmountClearingOrder\\r\\n .toUint96());\\r\\n currentBidSum =
currentBidSum.sub(uncoveredBids);\\r\\n clearingOrder = currentOrder;\\r\\n } else {\\r\\n
//[14]\\r\\n // Auction fully filled via price strictly between currentOrder and the order\\r\\n //
immediately before. For a proof, see the security-considerations.md\\r\\n currentBidSum =
currentBidSum.sub(sellAmountOfIter);\\r\\n clearingOrder =
IterableOrderedOrderSet.encodeOrder(\\r\\n 0,\\r\\n fullAuctionedAmount,\\r\\n
currentBidSum.toUint96())\\r\\n );\\r\\n }\\r\\n } else {\\r\\n // All considered/summed orders are
not sufficient to close the auction fully at price of last order //[18]\\r\\n // Either a higher price
must be used or auction is only partially filled\\r\\n\\r\\n if (currentBidSum >
minAuctionedBuyAmount) {\\r\\n //[15]\\r\\n // Price higher than last order would fill the
auction\\r\\n clearingOrder = IterableOrderedOrderSet.encodeOrder(\\r\\n 0,\\r\\n
fullAuctionedAmount,\\r\\n currentBidSum.toUint96())\\r\\n );\\r\\n } else {\\r\\n //[16]\\r\\n // Even
at the initial auction price, the auction is partially filled\\r\\n clearingOrder =
IterableOrderedOrderSet.encodeOrder(\\r\\n 0,\\r\\n fullAuctionedAmount,\\r\\n
minAuctionedBuyAmount\\r\\n );\\r\\n fillVolumeOfAuctioneerOrder = currentBidSum\\r\\n
.mul(fullAuctionedAmount)\\r\\n .div(minAuctionedBuyAmount)\\r\\n .toUint96();\\r\\n }\\r\\n
}\\r\\n auctionData[auctionId].clearingPriceOrder = clearingOrder;\\r\\n\\r\\n if
(auctionData[auctionId].minFundingThreshold > currentBidSum) {\\r\\n
auctionData[auctionId].minFundingThresholdNotReached = true;\\r\\n }\\r\\n
processFeesAndAuctioneerFunds(\\r\\n auctionId,\\r\\n fillVolumeOfAuctioneerOrder,\\r\\n
auctioneerId,\\r\\n fullAuctionedAmount\\r\\n );\\r\\n emit AuctionCleared(\\r\\n auctionId,\\r\\n
fillVolumeOfAuctioneerOrder,\\r\\n uint96(currentBidSum),\\r\\n clearingOrder\\r\\n );\\r\\n // Gas
refunds\\r\\n auctionAccessManager[auctionId] = address(0);\\r\\n delete
auctionAccessData[auctionId];\\r\\n auctionData[auctionId].initialAuctionOrder =
bytes32(0);\\r\\n auctionData[auctionId].interimOrder = bytes32(0);\\r\\n
auctionData[auctionId].interimSumBidAmount = uint256(0);\\r\\n
auctionData[auctionId].minimumBiddingAmountPerOrder = uint256(0);\\r\\n }\\r\\n\\r\\n function
claimFromParticipantOrder(\\r\\n uint256 auctionId,\\r\\n bytes32[] memory orders\\r\\n )\\r\\n
public\\r\\n atStageFinished(auctionId)\\r\\n returns (\\r\\n uint256
sumAuctioningTokenAmount,\\r\\n uint256 sumBiddingTokenAmount\\r\\n )\\r\\n {\\r\\n for
(uint256 i = 0; i < orders.length; i++) {\\r\\n // Note: we don't need to keep any information
about the node since\\r\\n // no new elements need to be inserted.\\r\\n require(\\r\\n
sellOrders[auctionId].remove(orders[i]),\\r\\n "\\\"order is no longer claimable\\\""\\r\\n );\\r\\n }\\r\\n
AuctionData memory auction = auctionData[auctionId];\\r\\n (, uint96 priceNumerator, uint96
priceDenominator) =\\r\\n auction.clearingPriceOrder.decodeOrder();\\r\\n (uint64 userId, , ) =
orders[0].decodeOrder();\\r\\n bool minFundingThresholdNotReached =\\r\\n
auctionData[auctionId].minFundingThresholdNotReached;\\r\\n for (uint256 i = 0; i <
orders.length; i++) {\\r\\n (uint64 userIdOrder, uint96 buyAmount, uint96 sellAmount) =\\r\\n
orders[i].decodeOrder();\\r\\n require(\\r\\n userIdOrder == userId,\\r\\n "\\\"only allowed to claim

```

```

for same user\\\r\n );\\\r\n if (minFundingThresholdNotReached) {\r\n //10\r\n
sumBiddingTokenAmount = sumBiddingTokenAmount.add(sellAmount);\\\r\n } else {\r\n
//[23]\r\n if (orders[i] == auction.clearingPriceOrder) {\r\n //[25]\r\n
sumAuctioningTokenAmount = sumAuctioningTokenAmount.add(\\\r\n auction\\\r\n
.volumeClearingPriceOrder\\\r\n .mul(priceNumerator)\\\r\n .div(priceDenominator)\\\r\n );\\\r\n
sumBiddingTokenAmount = sumBiddingTokenAmount.add(\\\r\n
sellAmount.sub(auction.volumeClearingPriceOrder)\\\r\n );\\\r\n } else {\r\n if
(orders[i].smallerThan(auction.clearingPriceOrder)) {\r\n //[17]\r\n
sumAuctioningTokenAmount = sumAuctioningTokenAmount.add(\\\r\n
sellAmount.mul(priceNumerator).div(priceDenominator)\\\r\n );\\\r\n } else {\r\n //[24]\r\n
sumBiddingTokenAmount = sumBiddingTokenAmount.add(\\\r\n sellAmount\\\r\n );\\\r\n }\\\r\n
}\\\r\n emit ClaimedFromOrder(auctionId, userId, buyAmount, sellAmount);\\\r\n }\\\r\n
sendOutTokens(\\\r\n auctionId,\\\r\n sumAuctioningTokenAmount,\\\r\n
sumBiddingTokenAmount,\\\r\n userId\\\r\n ); //[3]\r\n }\\\r\n\\\r\n function
processFeesAndAuctioneerFunds(\\\r\n uint256 auctionId,\\\r\n uint256
fillVolumeOfAuctioneerOrder,\\\r\n uint64 auctioneerId,\\\r\n uint96 fullAuctionedAmount\\\r\n )
internal {\r\n uint256 feeAmount =\\\r\n
fullAuctionedAmount.mul(auctionData[auctionId].feeNumerator).div(\\\r\n
FEE_DENOMINATOR\\\r\n ); //[20]\r\n if
(auctionData[auctionId].minFundingThresholdNotReached) {\r\n sendOutTokens(\\\r\n
auctionId,\\\r\n fullAuctionedAmount.add(feeAmount),\\\r\n 0,\\\r\n auctioneerId\\\r\n ); //[4]\r\n }
else {\r\n //[11]\r\n (, uint96 priceNumerator, uint96 priceDenominator) =\\\r\n
auctionData[auctionId].clearingPriceOrder.decodeOrder();\\\r\n uint256
unsettledAuctionTokens =\\\r\n fullAuctionedAmount.sub(fillVolumeOfAuctioneerOrder);\\\r\n
uint256 auctioningTokenAmount =\\\r\n unsettledAuctionTokens.add(\\\r\n
feeAmount.mul(unsettledAuctionTokens).div(\\\r\n fullAuctionedAmount\\\r\n )\\\r\n );\\\r\n
uint256 biddingTokenAmount =\\\r\n
fillVolumeOfAuctioneerOrder.mul(priceDenominator).div(\\\r\n priceNumerator\\\r\n );\\\r\n
sendOutTokens(\\\r\n auctionId,\\\r\n auctioningTokenAmount,\\\r\n biddingTokenAmount,\\\r\n
auctioneerId\\\r\n ); //[5]\r\n sendOutTokens(\\\r\n auctionId,\\\r\n
feeAmount.mul(fillVolumeOfAuctioneerOrder).div(\\\r\n fullAuctionedAmount\\\r\n ),\\\r\n 0,\\\r\n
feeReceiverUserId\\\r\n ); //[7]\r\n }\\\r\n }\\\r\n\\\r\n function sendOutTokens(\\\r\n uint256
auctionId,\\\r\n uint256 auctioningTokenAmount,\\\r\n uint256 biddingTokenAmount,\\\r\n uint64
userId\\\r\n ) internal {\r\n address userAddress = registeredUsers.getAddressAt(userId);\\\r\n
if (auctioningTokenAmount > 0) {\r\n
auctionData[auctionId].auctioningToken.safeTransfer(\\\r\n userAddress,\\\r\n
auctioningTokenAmount\\\r\n );\\\r\n }\\\r\n if (biddingTokenAmount > 0) {\r\n
auctionData[auctionId].biddingToken.safeTransfer(\\\r\n userAddress,\\\r\n
biddingTokenAmount\\\r\n );\\\r\n }\\\r\n }\\\r\n\\\r\n function registerUser(address user) public

```

```

returns (uint64 userId) {
    numUsers = numUsers.add(1).toUint64();
    require(
        registeredUsers.insert(numUsers, user),
        "User already registered"
    );
    userId = numUsers;
    emit UserRegistration(user, userId);
}

function getUserId(address user) public returns (uint64 userId) {
    if (registeredUsers.hasAddress(user)) {
        userId = registeredUsers.getId(user);
    } else {
        userId = registerUser(user);
        emit NewUser(userId, user);
    }
}

function getSecondsRemainingInBatch(uint256 auctionId) public view returns (uint256) {
    if (
        auctionData[auctionId].auctionEndDate < block.timestamp
    ) {
        return 0;
    }
    return auctionData[auctionId].auctionEndDate.sub(block.timestamp);
}

function containsOrder(uint256 auctionId, bytes32 order) public view returns (bool) {
    return sellOrders[auctionId].contains(order);
}

}

", "keccak256": "0x8c14497bd253bed46a26f4858fffc60dbc7705475cd3ba2b952a40d7c0ad726c"}
, "contracts/interfaces/AllowListVerifier.sol": {
    "content": "
// SPDX-License-Identifier: LGPL-3.0-or-later
pragma solidity >=0.6.8;

library AllowListVerifierHelper {
    /// @dev Value returned by a call to `isAllowed` if the check was successful. The value is defined as:
    /// bytes4(keccak256("isAllowed(address,uint256,bytes)"))
    bytes4 internal constant MAGICVALUE = 0x19a05a7e;
}

/// @dev Standardized interface for an allowList manager for easyAuction
/// The interface was inspired by EIP-1271
interface AllowListVerifier {
    /// @dev Should return whether the a specific user has access to an auction
    /// by returning the magic value from AllowListVerifierHelper
    function isAllowed(
        address user,
        uint256 auctionId,
        bytes calldata callData
    ) external view returns (bytes4);
}

", "keccak256": "0xe646b09ca1a8756d2b10c4d50370cecb6a88c4c63bf56a5f83af3794c7f41a80", "license": "LGPL-3.0-or-later"},
"contracts/libraries/IdToAddressBiMap.sol": {
    "content": "
pragma solidity ^0.6.0;

Contract does not have test coverage, as it was nearly copied from:
https://github.com/gnosis/solidity-data-structures/blob/master/contracts/libraries/IdToAddressBiMap.sol

// The only change is uint16 -> uint64

library IdToAddressBiMap {
    struct Data {
        mapping(uint64 => address) idToAddress;
        mapping(address => uint64) addressToId;
    }

    function hasId(Data storage self, uint64 id) internal view returns (bool) {
        return self.idToAddress[id + 1] != address(0);
    }

    function hasAddress(Data storage self, address addr) internal view returns (bool) {
        return self.addressToId[addr] != 0;
    }

    function getAddressAt(Data storage self, uint64 id) internal view returns (address) {
        require(hasId(self, id), "Must have ID to get Address");
        return self.idToAddress[id + 1];
    }

    function getId(Data storage self, address addr) internal view

```

```

returns (uint64){
    require(hasAddress(self, addr), "Must have Address to get ID");
    return self.addressTold[addr] - 1;
}

function insert(
    Data storage self,
    uint64 id,
    address addr
) internal returns (bool) {
    require(addr != address(0), "Cannot insert zero address");
    require(id != uint64(-1), "Cannot insert max uint64");
    // Ensure bijectivity of the mappings
    if (self.addressTold[addr] != 0 || self.idToAddress[id + 1] != address(0)) {
        return false;
    }
    self.idToAddress[id + 1] = addr;
    self.addressTold[addr] = id + 1;
    return true;
}
}

", "keccak256":"0x44c23c11087ac487704504e99658ae1303d1cad5d5ba0b171a61adf04e2cd5ea", "contracts/libraries/IterableOrderedOrderSet.sol":{"content":"pragma solidity >=0.6.8;

import
"@openzeppelin/contracts/math/SafeMath.sol";

library IterableOrderedOrderSet
{
    using SafeMath for uint96;
    using IterableOrderedOrderSet for bytes32;

    // represents smallest possible value for an order under comparison of fn smallerThan()
    bytes32 internal constant QUEUE_START =
0x0000000000000000000000000000000000000000000000000000000000000001;

    // represents highest possible value for an order under comparison of fn smallerThan()
    bytes32 internal constant QUEUE_END =
0xffffffffffffffffffffffff000000000000000000000001;

    /// The struct is used to implement a modified version of a doubly linked list with sorted elements. The list starts from QUEUE_START to QUEUE_END, and each node keeps track of its predecessor and successor.
    /// Nodes can be added or removed.
    /// `next` and `prev` have a different role. The list is supposed to be traversed with `next`. If `next` is empty, the node is not part of the list. However, `prev` might be set for elements that are not in the list, which is why it should not be used for traversing. Having a `prev` set for elements not in the list is used to keep track of the history of the position in the list of a removed element.
    struct Data {
        mapping(bytes32 => bytes32) nextMap;
        mapping(bytes32 => bytes32) prevMap;
    }

    struct Order {
        uint64 owner;
        uint96 buyAmount;
        uint96 sellAmount;
    }

    function initializeEmptyList(Data storage self) internal {
        self.nextMap[QUEUE_START] = QUEUE_END;
        self.prevMap[QUEUE_END] = QUEUE_START;
    }

    function isEmpty(Data storage self) internal view returns (bool) {
        return self.nextMap[QUEUE_START] == QUEUE_END;
    }

    function insert(
        Data storage self,
        bytes32 elementToInsert,
        bytes32 elementBeforeNewOne
    ) internal returns (bool) {
        (, , uint96 denominator) = decodeOrder(elementToInsert);
        require(denominator != uint96(0), "Inserting zero is not supported");
        require(
            elementToInsert != QUEUE_START && elementToInsert != QUEUE_END,
            "Inserting element is not valid"
        );
        if (contains(self, elementToInsert)) {
            return false;
        }
        if (elementBeforeNewOne != QUEUE_START && self.prevMap[elementBeforeNewOne] == bytes32(0)) {
            return false;
        }
        if (!elementBeforeNewOne.smallerThan(elementToInsert)) {
            return

```

```

false;\r\n }\r\n\r\n // `elementBeforeNewOne` might have been removed during the time
it\r\n // took to the transaction calling this function to be mined, so\r\n // the new order cannot
be appended directly to this. We follow the\r\n // history of previous links backwards until we
find an element in\r\n // the list from which to start our search.\r\n // Note that following the
link backwards returns elements that are\r\n // before `elementBeforeNewOne` in sorted
order.\r\n while (self.nextMap[elementBeforeNewOne] == bytes32(0)) {\r\n
elementBeforeNewOne = self.prevMap[elementBeforeNewOne];\r\n }\r\n\r\n //
`elementBeforeNewOne` belongs now to the linked list. We search the\r\n // largest entry that
is smaller than the element to insert.\r\n bytes32 previous;\r\n bytes32 current =
elementBeforeNewOne;\r\n do {\r\n previous = current;\r\n current =
self.nextMap[current];\r\n } while (current.smallerThan(elementToInsert));\r\n // Note:
previous < elementToInsert < current\r\n self.nextMap[previous] = elementToInsert;\r\n
self.prevMap[current] = elementToInsert;\r\n self.prevMap[elementToInsert] = previous;\r\n
self.nextMap[elementToInsert] = current;\r\n\r\n return true;\r\n }\r\n\r\n /// The element is
removed from the linked list, but the node retains\r\n /// information on which predecessor it
had, so that a node in the chain\r\n /// can be reached by following the predecessor chain of
deleted elements.\r\n function removeKeepHistory(Data storage self, bytes32
elementToRemove)\r\n internal\r\n returns (bool)\r\n {\r\n if (!contains(self,
elementToRemove)) {\r\n return false;\r\n }\r\n bytes32 previousElement =
self.prevMap[elementToRemove];\r\n bytes32 nextElement =
self.nextMap[elementToRemove];\r\n self.nextMap[previousElement] = nextElement;\r\n
self.prevMap[nextElement] = previousElement;\r\n self.nextMap[elementToRemove] =
bytes32(0);\r\n return true;\r\n }\r\n\r\n /// Remove an element from the chain, clearing all
related storage.\r\n /// Note that no elements should be inserted using as a reference point
a\r\n /// node deleted after calling `remove`, since an element in the `prev`\r\n /// chain might
be missing.\r\n function remove(Data storage self, bytes32 elementToRemove)\r\n
internal\r\n returns (bool)\r\n {\r\n bool result = removeKeepHistory(self,
elementToRemove);\r\n if (result) {\r\n self.prevMap[elementToRemove] = bytes32(0);\r\n
}\r\n return result;\r\n }\r\n\r\n function contains(Data storage self, bytes32 value)\r\n
internal\r\n view\r\n returns (bool)\r\n {\r\n if (value == QUEUE_START) {\r\n return
false;\r\n }\r\n // Note: QUEUE_END is not contained in the list since it has no\r\n //
successor.\r\n return self.nextMap[value] != bytes32(0);\r\n }\r\n\r\n // @dev orders are
ordered by\r\n // 1. their price - buyAmount/sellAmount\r\n // 2. by the sellAmount\r\n // 3.
their userId,\r\n function smallerThan(bytes32 orderLeft, bytes32 orderRight)\r\n internal\r\n
pure\r\n returns (bool)\r\n {\r\n (\r\n uint64 userIdLeft,\r\n uint96 priceNumeratorLeft,\r\n
uint96 priceDenominatorLeft\r\n ) = decodeOrder(orderLeft);\r\n (\r\n uint64
userIdRight,\r\n uint96 priceNumeratorRight,\r\n uint96 priceDenominatorRight\r\n ) =
decodeOrder(orderRight);\r\n\r\n if (\r\n priceNumeratorLeft.mul(priceDenominatorRight)
<\r\n priceNumeratorRight.mul(priceDenominatorLeft)\r\n ) return true;\r\n if (\r\n

```

```

priceNumeratorLeft.mul(priceDenominatorRight) >\\r\\n
priceNumeratorRight.mul(priceDenominatorLeft)\\r\\n ) return false;\\r\\n\\r\\n if
(priceNumeratorLeft < priceNumeratorRight) return true;\\r\\n if (priceNumeratorLeft >
priceNumeratorRight) return false;\\r\\n require(\\r\\n userIdLeft != userIdRight,\\r\\n \\\"user is
not allowed to place same order twice\\\"\\r\\n );\\r\\n if (userIdLeft < userIdRight) {\\r\\n return
true;\\r\\n }\\r\\n return false;\\r\\n }\\r\\n\\r\\n function first(Data storage self) internal view
returns (bytes32) {\\r\\n require(!isEmpty(self), \\\"Trying to get first from empty set\\\");\\r\\n
return self.nextMap[QUEUE_START];\\r\\n }\\r\\n\\r\\n function next(Data storage self, bytes32
value)\\r\\n internal\\r\\n view\\r\\n returns (bytes32)\\r\\n {\\r\\n require(value != QUEUE_END,
\\\"Trying to get next of last element\\\");\\r\\n bytes32 nextElement = self.nextMap[value];\\r\\n
require(\\r\\n nextElement != bytes32(0),\\r\\n \\\"Trying to get next of non-existent
element\\\"\\r\\n );\\r\\n return nextElement;\\r\\n }\\r\\n\\r\\n function decodeOrder(bytes32
_orderData)\\r\\n internal\\r\\n pure\\r\\n returns (\\r\\n uint64 userId,\\r\\n uint96
buyAmount,\\r\\n uint96 sellAmount\\r\\n )\\r\\n {\\r\\n // Note: converting to uint discards the
binary digits that do not fit\\r\\n // the type.\\r\\n userId = uint64(uint256(_orderData) >>
192);\\r\\n buyAmount = uint96(uint256(_orderData) >> 96);\\r\\n sellAmount =
uint96(uint256(_orderData));\\r\\n }\\r\\n\\r\\n function encodeOrder(\\r\\n uint64 userId,\\r\\n
uint96 buyAmount,\\r\\n uint96 sellAmount\\r\\n ) internal pure returns (bytes32) {\\r\\n
return\\r\\n bytes32(\\r\\n (uint256(userId) << 192) +\\r\\n (uint256(buyAmount) << 96) +\\r\\n
uint256(sellAmount)\\r\\n );\\r\\n
}\\r\\n}\\r\\n\\\",\\\"keccak256\\\":\\\"0x9510258dca14a3327710d516a1b5c0e7147e20ac8f5a29a9804
8d97878ff1dac\\\"},\\\"contracts/libraries/SafeCast.sol\\\":{\\\"content\\\":\\\"// SPDX-License-Identifier:
MIT\\r\\n\\r\\npragma solidity >=0.6.0 <0.8.0;\\r\\n\\r\\n/**\\r\\n * @dev Wrappers over Solidity's
uintXX/intXX casting operators with added overflow\\r\\n * checks.\\r\\n *\\r\\n * Logic was
copied and modified from here: https://github.com/OpenZeppelin/openzeppelin-
contracts/blob/master/contracts/utils/SafeCast.sol\\r\\n *\\r\\nlibrary SafeCast {\\r\\n function
toUint96(uint256 value) internal pure returns (uint96) {\\r\\n require(value < 2**96, \\\"SafeCast:
value doesn't fit in 96 bits\\\");\\r\\n return uint96(value);\\r\\n }\\r\\n\\r\\n function
toUint64(uint256 value) internal pure returns (uint64) {\\r\\n require(value < 2**64, \\\"SafeCast:
value doesn't fit in 64 bits\\\");\\r\\n return uint64(value);\\r\\n
}\\r\\n}\\r\\n\\\",\\\"keccak256\\\":\\\"0xe9d43c6764bad8042246919c329bb1cf5385ba190374f64f1359
82b8d1c0bbdd\\\",\\\"license\\\":\\\"MIT\\\"}},\\\"version\\\":1}, \\\"bytecode\\\":
\\\"0x608060405260006009556001600a60006101000a81548167ffffffffffff021916908367fffffffff
fffff1602179055503480156200004057600080fd5b50600062000053620000f760201b60201c5
65b9050806000806101000a81548173ffffffffffffffffffffffffffff021916908373fffffffffffffffffffff
fffff1602179055508073fffffffffffffffffffffffffffff16600073fffffffffffffffffffffffffffff167f8be00
79c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e0604051604051809103
90a350620000ff565b600033905090565b615b6e806200010f6000396000f3fe6080604052348
01561001057600080fd5b50600436106101585760003560e01c80637882deaf116100c357806

```

3d73792a91161007c578063d73792a914610ccf578063e4a59ef414610ced578063e86dea4a1
4610d45578063ec20d0bb14610d63578063f2fde38b14610fe0578063f59c2f06146110245761
0158565b80637882deaf146106425780637ed18b701461071f5780638da5cb5b146107e15780
6391cfc1d414610815578063a7e7664414610a54578063d225269c14610a7257610158565b8
0633e12905f116101155780633e12905f1461046157806340b20b091461049957806355fc62d
2146104e75780635cefb291146105c257806363c699a4146105ea578063715018a6146106385
7610158565b80630a4cd6c91461015d57806315d37b4b146102f157806319a50f49146103335
780632199d5cd1461035b5780632b956ff7146103bd5780632e9936111461041f575b600080fd
5b6102db600480360361016081101561017457600080fd5b81019080803573ffffffffffffffff
ffffffff169060200190929190803573ffffffffffffffffffffffffffffffff169060200190929190803590602
001909291908035906020019092919080356bffffffffffffffffffffffff16906020019092919080356bfffff
ffffffff169060200190929190803590602001909291908035906020019092919080351515
9060200190929190803573ffffffffffffffffffffffffffffffff1690602001909291908035906020019064
010000000081111561025557600080fd5b82018360208201111561026757600080fd5b803590
6020019184600183028401116401000000008311171561028957600080fd5b91908080601f01
602080910402602001604051908101604052809392919081815260200183838082843760008
1840152601f19601f8201169050808301925050505050505091929192905050506110cb565b6
040518082815260200191505060405180910390f35b61031d60048036036020811015610307
57600080fd5b810190808035906020019092919050505061178a565b6040518082815260200
191505060405180910390f35b61033b6117e4565b604051808267ffffffff168152602001915
05060405180910390f35b61039d6004803603602081101561037157600080fd5b81019080803
573ffffffffffffffffffffffffffffffff1690602001909291905050506117fe565b604051808267ffffffff
16815260200191505060405180910390f35b6103ff600480360360208110156103d357600080f
d5b81019080803573ffffffffffffffffffffffffffffffff169060200190929190505050611978565b60405
1808267ffffffff16815260200191505060405180910390f35b61044b6004803603602081101
561043557600080fd5b8101908080359060200190929190505050611a0d565b604051808281
5260200191505060405180910390f35b6104976004803603604081101561047757600080fd5b
810190808035906020019092919080359060200190929190505050611fb4565b005b6104e56
00480360360408110156104af57600080fd5b8101908080359060200190929190803573ffffffff
ffffffff1690602001909291905050506122d6565b005b61051360048036036020811
0156104fd57600080fd5b8101908080359060200190929190505050612434565b604051808f7
3ffffffffffffffffffffffffffffffff1681526020018e73ffffffffffffffffffffffffffffffff1681526020018d815260
20018c81526020018b81526020018a815260200189815260200188815260200187815260200
1866bffffffffffffffff1681526020018515158152602001841515815260200183815260200182
81526020019e5050505050505050505050505050505060405180910390f35b6105ca612512565
b604051808267ffffffff16815260200191505060405180910390f35b6106206004803603604
081101561060057600080fd5b81019080803590602001909291908035906020019092919050
505061252c565b60405180821515815260200191505060405180910390f35b61064061255b5
65b005b6107026004803603604081101561065857600080fd5b810190808035906020019092

91908035906020019064010000000081111561067f57600080fd5b8201836020820111156106
9157600080fd5b803590602001918460208302840111640100000000831117156106b3576000
80fd5b919080806020026020016040519081016040528093929190818152602001838360200
280828437600081840152601f19601f820116905080830192505050505050509192919290505
0506126e1565b604051808381526020018281526020019250505060405180910390f35b6107
df6004803603604081101561073557600080fd5b810190808035906020019092919080359060
20019064010000000081111561075c57600080fd5b82018360208201111561076e57600080fd
5b8035906020019184602083028401116401000000008311171561079057600080fd5b919080
806020026020016040519081016040528093929190818152602001838360200280828437600
081840152601f19601f820116905080830192505050505050509192919290505050612d2d565
b005b6107e9612f90565b604051808273ffffffffffffffffffffffffffffffff168152602001915050604051
80910390f35b610a52600480360360a081101561082b57600080fd5b81019080803590602001
909291908035906020019064010000000081111561085257600080fd5b820183602082011115
61086457600080fd5b8035906020019184602083028401116401000000008311171561088657
600080fd5b91908080602002602001604051908101604052809392919081815260200183836
0200280828437600081840152601f19601f82011690508083019250505050505050919291929
0803590602001906401000000008111156108e657600080fd5b8201836020820111156108f85
7600080fd5b8035906020019184602083028401116401000000008311171561091a57600080f
d5b919080806020026020016040519081016040528093929190818152602001838360200280
828437600081840152601f19601f820116905080830192505050505050509192919290803590
6020019064010000000081111561097a57600080fd5b82018360208201111561098c57600080
fd5b803590602001918460208302840111640100000000831117156109ae57600080fd5b9190
808060200260200160405190810160405280939291908181526020018383602002808284376
00081840152601f19601f8201169050808301925050505050505091929192908035906020019
0640100000000811115610a0e57600080fd5b820183602082011115610a2057600080fd5b803
59060200191846001830284011164010000000083111715610a4257600080fd5b90919293919
29390505050612fb9565b005b610a5c613221565b6040518082815260200191505060405180
910390f35b610caf600480360360a0811015610a8857600080fd5b8101908080359060200190
92919080359060200190640100000000811115610aaf57600080fd5b82018360208201111561
0ac157600080fd5b80359060200191846020830284011164010000000083111715610ae35760
0080fd5b9190808060200260200160405190810160405280939291908181526020018383602
00280828437600081840152601f19601f8201169050808301925050505050505091929192908
0359060200190640100000000811115610b4357600080fd5b820183602082011115610b55576
00080fd5b80359060200191846020830284011164010000000083111715610b7757600080fd5
b91908080602002602001604051908101604052809392919081815260200183836020028082
8437600081840152601f19601f82011690508083019250505050505050919291929080359060
200190640100000000811115610bd757600080fd5b820183602082011115610be957600080fd
5b80359060200191846020830284011164010000000083111715610c0b57600080fd5b919080
806020026020016040519081016040528093929190818152602001838360200280828437600

081840152601f19601f820116905080830192505050505050509192919290803590602001906
40100000000811115610c6b57600080fd5b820183602082011115610c7d57600080fd5b803590
60200191846001830284011164010000000083111715610c9f57600080fd5b90919293919293
90505050613227565b604051808267ffffffffffffff16815260200191505060405180910390f35b61
0cd76132b4565b6040518082815260200191505060405180910390f35b610d1960048036036
020811015610d0357600080fd5b81019080803590602001909291905050506132ba565b6040
51808273ffffffffffffffffffffffffffffffffff16815260200191505060405180910390f35b610d4d6132ed5
65b6040518082815260200191505060405180910390f35b610fc0600480360360c0811015610
d7957600080fd5b810190808035906020019092919080359060200190640100000000811115
610da057600080fd5b820183602082011115610db257600080fd5b8035906020019184602083
0284011164010000000083111715610dd457600080fd5b919080806020026020016040519081
016040528093929190818152602001838360200280828437600081840152601f19601f820116
90508083019250505050505050919291929080359060200190640100000000811115610e345
7600080fd5b820183602082011115610e4657600080fd5b80359060200191846020830284011
164010000000083111715610e6857600080fd5b919080806020026020016040519081016040
528093929190818152602001838360200280828437600081840152601f19601f820116905080
83019250505050505050919291929080359060200190640100000000811115610ec85760008
0fd5b820183602082011115610eda57600080fd5b80359060200191846020830284011164010
000000083111715610efc57600080fd5b9190808060200260200160405190810160405280939
29190818152602001838360200280828437600081840152601f19601f8201169050808301925
0505050505050919291929080359060200190640100000000811115610f5c57600080fd5b820
183602082011115610f6e57600080fd5b8035906020019184600183028401116401000000008
3111715610f9057600080fd5b9091929391929390803573ffffffffffffffffffffffffffffffffff1690602001
909291905050506132f3565b604051808267ffffffffffffff16815260200191505060405180910390
f35b61102260048036036020811015610ff657600080fd5b81019080803573ffffffffffffffffffffffffffffffffff
ffffff169060200190929190505050613381565b005b6110506004803603602081101561103a57
600080fd5b810190808035906020019092919050505061358c565b604051808060200182810
3825283818151815260200191508051906020019080838360005b8381101561109057808201
5181840152602081019050611075565b50505050905090810190601f1680156110bd5780820
380516001836020036101000a031916815260200191505b509250505060405180910390f35b
6000611144333061111d6103e861110f6110f26009546103e861363c90919063ffffff16565b8e6
bffffffffffffffffffffff166136c490919063ffffff16565b61374a90919063ffffff16565b8f73ffffffffffffffffffff
ffffffffffffff16613794909392919063ffffff16565b6000886bffffffffffffffffffffff16116111c85760405
17f08c379a00081526004
0180806020018281038252601a8152602001807f63616e6e6f742061756374696f6e207a6572
6f20746f6b656e73000000000000081525060200191505060405180910390fd5b6000876bffffffff
ffffffffffffff161161122f576040517f08c379a000
0000000000000000815260040180806020018281038252602381526020018061584060239139
60400191505060405180910390fd5b60008611611288576040517f08c379a0000000000000000

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

4210613298576040517f08c379a000
0000000008152600401808060200182810382526022815260200180615b1760229139604001
91505060405180910390fd5b6132a788888888888888336146a8565b915050969550505050505
0565b6103e881565b60036020528060005260406000206000915054906101000a900473ffffff
ffffffffffffffffffffffffffff1681565b60095481565b6000876002600082815260200190815260200160
0020600301544210613364576040517f08c379a00
00000000000000000000008152600401808060200182810382526022815260200180615b1760
22913960400191505060405180910390fd5b6133738989898989898989896146a8565b915050979
65050505050505050565b6133896141c7565b73ffffffffffffffffffffffffffff1660008054906101000
a900473ffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffff1614613449576040517f08c379a
000815260040180806020
01828103825260208152602001807f4f776e61626c653a2063616c6c6572206973206e6f7420
746865206f776e657281525060200191505060405180910390fd5b600073ffffffffffffffffffffffffffff
ffff168173ffffffffffffffffffffffffffff1614156134cf576040517f08c379a00000000000000000000
008152600401808060200182810382526026815
26020018061581a6026913960400191505060405180910390fd5b8073ffffffffffffffffffffffffffff
1660008054906101000a900473ffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffff167f8be0
079c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e060405160405180910
390a3806000806101000a81548173ffffffffffffffffffffffffffff021916908373ffffffffffffffffffff
ffffff16021790555050565b60046020528060005260406000206000915090508054600181600
116156101000203166002900480601f01602080910402602001604051908101604052809291
90818152602001828054600181600116156101000203166002900480156136345780601f106
1360957610100808354040283529160200191613634565b820191906000526020600020905b
81548152906001019060200180831161361757829003601f168201915b505050505081565b6
000808284019050838110156136ba576040517f08c379a00000000000000000000000000000000
000815260040180806020018281038252601b8152602001807
f536166654d6174683a206164646974696f6e206f766572666c6f7700000000000815250602001
91505060405180910390fd5b8091505092915050565b6000808314156136d75760009050613
744565b60008284029050828482816136e857fe5b041461373f576040517f08c379a000000000
0081526004018080602001828103
825260218152602001806159dc6021913960400191505060405180910390fd5b809150505b9
2915050565b600061378c83836040518060400160405280601a81526020017f536166654d61
74683a206469766973696f6e206279207a65726f0000000000000815250614cc0565b90509291
5050565b61384f846323b872dd60e01b858585604051602401808473ffffffffffffffffffffffffff1
681526020018373ffffffffffffffffffffffffff16815260200182815260200193505050506040516
02081830303815290604052907bffffffffffffffffffffffffff19166020820180517bffffff
ffffffffffffffffff8381831617835250505050614d86565b50505050565b7fffffffffff
ffffffffffff00
0815260200160002081905550600160001b8160010160007ffffffffffffffffff00000000

0000000000000000160001b81526020019081526020016000208190555050565b6000816bf
ffffff166060846bffffffffffff16901b60c08667ffffffffffff16901b010160001b90509392
505050565b600061395d83836040518060400160405280601e81526020017f536166654d617
4683a207375627472616374696f6e206f766572666c6f770000815250614e75565b905092915
050565b60006801000000000000000082106139c8576040517f08c379a0000000000000000
000000000000000000000000000000000081526004018080602001828103825260268
152602001806159fd6026913960400191505060405180910390fd5b819050919050565b60008
073ffffffffffff168273ffffffffffff161415613a74576040517f08c379a
00815260040180806020
018281038252601a8152602001807f43616e6e6f7420696e73657274207a65726f2061646472
657373000000000000081525060200191505060405180910390fd5b7ffffffffffff
ffffff67ffffffff168367ffffffff161415613b1e576040517f08c379a00000000000000
0081526004018080602001828103825260
188152602001807f43616e6e6f7420696e73657274206d61782075696e743634000000000000
000081525060200191505060405180910390fd5b60008460010160008473ffffffffffff
ffff1673ffffffffffff16815260200190815260200160002060009054906101000a9
00467ffffffff1667ffffffff16141580613c065750600073ffffffffffff16846000
0160006001860167ffffffff1667ffffffff1681526020019081526020016000206000905490
6101000a900473ffffffffffff1673ffffffffffff1614155b15613c145760
009050613cef565b818460000160006001860167ffffffff1667ffffffff1681526020019081
5260200160002060006101000a81548173ffffffffffff021916908373ffffffffffff
ffffff160217905550600183018460010160008473ffffffffffff1673ffffffffffff
ffffff16815260200190815260200160002060006101000a81548167ffffffffffff021916
908367ffffffff160217905550600190505b9392505050565b6000808360010160008473fffff
ffffff1673ffffffffffff1681526020019081526020016000206000905
4906101000a900467ffffffff1667ffffffff161415905092915050565b6000613d6f8383613
cf6565b613de1576040517f08c379a00
0000000000815260040180806020018281038252601b8152602001807f4d75737420686176
65204164647265737320746f20676574204944000000000815250602001915050604051809
10390fd5b60018360010160008473ffffffffffff1673ffffffffffff16815
260200190815260200160002060009054906101000a900467ffffffff160390509291505056
5b600080600060c08460001c901c925060608460001c901c91508360001c905091939092505
65b60007ffffffffffff00
517f08c379a008152600
4018080602001828103825260228152602001806158ba602291396040019150506040518091
0390fd5b60008360000160008481526020019081526020016000205490506000801b8114156
13f5c576040517f08c379a000
000815260040180806020018281038252602a81526020018061597c602a9139604001915050
60405180910390fd5b8091505092915050565b60006c01000000000000000000000000082106

13fcd576040517f08c379a00
000815260040180806020018281038252602681526020018061577860269139604001915050
60405180910390fd5b819050919050565b60006140226103e86140146002600089815260200
1908152602001600020600a0154856bffffffffffffffffffff166136c490919063ffffff16565b61374a9
0919063ffffff16565b905060026000868152602001908152602001600020600901600c905490
6101000a900460ff161561407f5761407a8561407283856bffffffffffffffffffff1661363c90919063fff
ffff16565b600086614466565b6141c0565b6000806140a16002600089815260200190815260
200160002060080154613e42565b925092505060006140c987866bffffffffffffffffffff1661391b90
919063ffffff16565b90506000614112614103876bffffffffffffffffffff166140f585896136c4909190
63ffffff16565b61374a90919063ffffff16565b8361363c90919063ffffff16565b9050600061415
7856bffffffffffffffffffff16614149866bffffffffffffffffffff168c6136c490919063ffffff16565b61374a90
919063ffffff16565b90506141658a83838b614466565b6141ba8a61419c896bffffffffffffffffffff16
61418e8d8b6136c490919063ffffff16565b61374a90919063ffffff16565b6000600a600090549
06101000a900467ffffffffffffff16614466565b5050505050505b5050505050565b60003390509056
5b6000600160001b8214156141e65760009050614205565b6000801b8360000160008481526
0200190815260200160002054141590505b92915050565b600080614218848461455b565b90
50801561423e576000801b846001016000858152602001908152602001600020819055505b8
091505092915050565b60008060008061425786613e42565b92509250925060008060006142
6b88613e42565b9250925092506142a0846bffffffffffffffffffff16836bffffffffffffffffffff166136c490
919063ffffff16565b6142cf826bffffffffffffffffffff16876bffffffffffffffffffff166136c490919063ffffff16
565b10156142e45760019650505050505050614460565b614313846bffffffffffffffffffff16836bffff
ffffffffffffffffffff166136c490919063ffffff16565b614342826bffffffffffffffffffff16876bffffffffffffffffffff1
66136c490919063ffffff16565b11156143575760009650505050505050614460565b816bffffffffffff
ffffffffffff16856bffffffffffffffffffff16101561438a5760019650505050505050614460565b816bffffff
ffffffffffff16856bffffffffffffffffffff1611156143bd5760009650505050505050614460565b8267ffff
ffffffffffff168667ffffffffffffff16141561442a576040517f08c379a000000000000000000000000000
00
00
0615a6e602d913960400191505060405180910390fd5b8267ffffffffffff168667ffffffffffff16101
56144555760019650505050505050614460565b60009650505050505050505b92915050565b60
0061447c826005614f3590919063ffffff16565b905060008411156144e9576144e88185600260
0089815260200190815260200160002060000160009054906101000a900473ffffffffffffffffffff
ffffffffffff1673ffffffffffffffffffff166146069092919063ffffff16565b5b60008311156145545
761455381846002600089815260200190815260200160002060010160009054906101000a90
0473ffffffffffffffffffff1673ffffffffffffffffffff166146069092919063ffffff16565b5b
505050505050565b600061456783836141cf565b6145745760009050614600565b60008360010
160008481526020019081526020016000205490506000846000016000858152602001908152
602001600020549050808560000160008481526020019081526020016000208190555081856
001016000838152602001908152602001600020819055506000801b85600001600086815260
2001908152602001600020819055506001925050505b92915050565b6146a38363a9059ccb6

[illegible]

[illegible]

573742062652067726561746572207468616e2030547279696e6720746f20676574206e6578
74206f66206c61737420656c656d656e74416464726573733a20696e73756666696369656e7
42062616c616e636520666f722063616c6c466565206973206e6f7420616c6c6f77656420746f
2062652073657420686967686572207468616e20312e352561756374696f6e20656e6420646
17465206d75737420626520696e20746865206675747572654f6e6c79206f6e65206f7264657
22063616e20626520706c616365642061746f6d6963616c6c79547279696e6720746f2067657
4206e657874206f66206e6f6e2d6578697374656e7420656c656d656e746d696e696d756d426
96464696e67416d6f756e745065724f72646572206973206e6f7420616c6c6f77656420746f20
6265207a65726f536166654d6174683a206d756c7469706c69636174696f6e206f766572666c
6f7753616665436173743a2076616c756520646f65736e27742066697420696e203634206269
74736f6e6c7920616c6c6f77656420746f20636c61696d20666f722073616d652075736572417
56374696f6e206e6f7420696e20736f6c7574696f6e207375626d697373696f6e207068617365
75736572206973206e6f7420616c6c6f77656420746f20706c6163652073616d65206f7264657
22074776963656c696d6974207072696365206e6f7420626574746572207468616e206d696d
696d616c206f6666657274696d6520706572696f647320617265206e6f7420636f6e666967757
2656420636f72726563746c795361666545524332303a204552433230206f7065726174696f6
e20646964206e6f7420737563636565646e6f206c6f6e67657220696e206f7264657220706c61
63656d656e74207068617365a26469706673582212209d3f3585da1ffe9756c8a2e5f28991662
e7f098068faf72dd2ec014b6bd536c964736f6c634300060c0033", "deployedBytecode":
"0x608060405234801561001057600080fd5b50600436106101585760003560e01c80637882d
eaf116100c3578063d73792a91161007c578063d73792a914610ccf578063e4a59ef414610ced
578063e86dea4a14610d45578063ec20d0bb14610d63578063f2fde38b14610fe0578063f59c2
f061461102457610158565b80637882deaf146106425780637ed18b701461071f5780638da5c
b5b146107e157806391cfc1d414610815578063a7e7664414610a54578063d225269c14610a
7257610158565b80633e12905f116101155780633e12905f1461046157806340b20b09146104
9957806355fc62d2146104e75780635cefb291146105c257806363c699a4146105ea57806371
5018a61461063857610158565b80630a4cd6c91461015d57806315d37b4b146102f15780631
9a50f49146103335780632199d5cd1461035b5780632b956ff7146103bd5780632e9936111461
041f575b600080fd5b6102db600480360361016081101561017457600080fd5b810190808035
73ffffffffffffffffffffffffffffffff169060200190929190803573ffffffffffffffffffffffffffffffff16906020019
0929190803590602001909291908035906020019092919080356bffffffffffffffffffffffff1690602001
9092919080356bffffffffffffffffffffffff1690602001909291908035906020019092919080359060200
190929190803515159060200190929190803573ffffffffffffffffffffffffffffffff16906020019092919
08035906020019064010000000081111561025557600080fd5b8201836020820111156102675
7600080fd5b8035906020019184600183028401116401000000008311171561028957600080f
d5b91908080601f016020809104026020016040519081016040528093929190818152602001
838380828437600081840152601f19601f8201169050808301925050505050509192919290
5050506110cb565b6040518082815260200191505060405180910390f35b61031d600480360
3602081101561030757600080fd5b810190808035906020019092919050505061178a565b60

40518082815260200191505060405180910390f35b61033b6117e4565b604051808267ffffffffffff
ffff16815260200191505060405180910390f35b61039d6004803603602081101561037157600
080fd5b81019080803573ffffffffffffffffffffffffffffffff1690602001909291905050506117fe565b60
4051808267ffffffffffff16815260200191505060405180910390f35b6103ff60048036036020811
0156103d357600080fd5b81019080803573ffffffffffffffffffffffffffffffff1690602001909291905050
50611978565b604051808267ffffffffffff16815260200191505060405180910390f35b61044b60
04803603602081101561043557600080fd5b8101908080359060200190929190505050611a0
d565b6040518082815260200191505060405180910390f35b61049760048036036040811015
61047757600080fd5b810190808035906020019092919080359060200190929190505050611f
b4565b005b6104e5600480360360408110156104af57600080fd5b8101908080359060200190
929190803573ffffffffffffffffffffffffffffffff1690602001909291905050506122d6565b005b610513
600480360360208110156104fd57600080fd5b81019080803590602001909291905050506124
34565b604051808f73ffffffffffffffffffffffffffffffff1681526020018e73ffffffffffffffffffff
1681526020018d81526020018c81526020018b81526020018a81526020018981526020018881526
02001878152602001866bffffffffffffffff168152602001851515815260200184151581526020
018381526020018281526020019e5050505050505050505050505050505060405180910390f35
b6105ca612512565b604051808267ffffffffffff16815260200191505060405180910390f35b610
6206004803603604081101561060057600080fd5b8101908080359060200190929190803590
6020019092919050505061252c565b60405180821515815260200191505060405180910390f
35b61064061255b565b005b6107026004803603604081101561065857600080fd5b81019080
803590602001909291908035906020019064010000000081111561067f57600080fd5b820183
60208201111561069157600080fd5b8035906020019184602083028401116401000000008311
17156106b357600080fd5b91908080602002602001604051908101604052809392919081815
2602001838360200280828437600081840152601f19601f8201169050808301925050505050
05091929192905050506126e1565b6040518083815260200182815260200192505050604051
80910390f35b6107df6004803603604081101561073557600080fd5b81019080803590602001
909291908035906020019064010000000081111561075c57600080fd5b820183602082011115
61076e57600080fd5b8035906020019184602083028401116401000000008311171561079057
600080fd5b91908080602002602001604051908101604052809392919081815260200183836
0200280828437600081840152601f19601f820116905080830192505050505050919291929
0505050612d2d565b005b6107e9612f90565b604051808273ffffffffffffffffffffffffffff16815260
200191505060405180910390f35b610a52600480360360a081101561082b57600080fd5b8101
9080803590602001909291908035906020019064010000000081111561085257600080fd5b82
018360208201111561086457600080fd5b803590602001918460208302840111640100000000
8311171561088657600080fd5b9190808060200260200160405190810160405280939291908
18152602001838360200280828437600081840152601f19601f8201169050808301925050505
05050509192919290803590602001906401000000008111156108e657600080fd5b820183602
0820111156108f857600080fd5b80359060200191846020830284011164010000000083111715
61091a57600080fd5b919080806020026020016040519081016040528093929190818152602

001838360200280828437600081840152601f19601f8201169050808301925050505050509
1929192908035906020019064010000000081111561097a57600080fd5b82018360208201111
561098c57600080fd5b803590602001918460208302840111640100000000831117156109ae5
7600080fd5b9190808060200260200160405190810160405280939291908181526020018383
60200280828437600081840152601f19601f82011690508083019250505050505091929192
9080359060200190640100000000811115610a0e57600080fd5b820183602082011115610a20
57600080fd5b80359060200191846001830284011164010000000083111715610a4257600080
fd5b9091929391929390505050612fb9565b005b610a5c613221565b60405180828152602001
91505060405180910390f35b610caf600480360360a0811015610a8857600080fd5b81019080
8035906020019092919080359060200190640100000000811115610aaf57600080fd5b820183
602082011115610ac157600080fd5b8035906020019184602083028401116401000000008311
1715610ae357600080fd5b91908080602002602001604051908101604052809392919081815
2602001838360200280828437600081840152601f19601f82011690508083019250505050505
050919291929080359060200190640100000000811115610b4357600080fd5b8201836020820
11115610b5557600080fd5b80359060200191846020830284011164010000000083111715610
b7757600080fd5b919080806020026020016040519081016040528093929190818152602001
838360200280828437600081840152601f19601f8201169050808301925050505050509192
91929080359060200190640100000000811115610bd757600080fd5b82018360208201111561
0be957600080fd5b80359060200191846020830284011164010000000083111715610c0b5760
0080fd5b9190808060200260200160405190810160405280939291908181526020018383602
00280828437600081840152601f19601f82011690508083019250505050505091929192908
0359060200190640100000000811115610c6b57600080fd5b820183602082011115610c7d576
00080fd5b80359060200191846001830284011164010000000083111715610c9f57600080fd5b
9091929391929390505050613227565b604051808267ffffffffffffffff16815260200191505060405
180910390f35b610cd76132b4565b6040518082815260200191505060405180910390f35b610
d1960048036036020811015610d0357600080fd5b8101908080359060200190929190505050
6132ba565b604051808273ffffffffffffffffffffffffffffffff16815260200191505060405180910390f35
b610d4d6132ed565b6040518082815260200191505060405180910390f35b610fc0600480360
360c0811015610d7957600080fd5b8101908080359060200190929190803590602001906401
00000000811115610da057600080fd5b820183602082011115610db257600080fd5b80359060
200191846020830284011164010000000083111715610dd457600080fd5b9190808060200260
200160405190810160405280939291908181526020018383602002808284376000818401526
01f19601f8201169050808301925050505050505091929192908035906020019064010000000
0811115610e3457600080fd5b820183602082011115610e4657600080fd5b803590602001918
46020830284011164010000000083111715610e6857600080fd5b91908080602002602001604
0519081016040528093929190818152602001838360200280828437600081840152601f1960
1f82011690508083019250505050505050919291929080359060200190640100000000811115
610ec857600080fd5b820183602082011115610eda57600080fd5b8035906020019184602083
0284011164010000000083111715610efc57600080fd5b919080806020026020016040519081

[illegible]

6101000a81548160ff02191690831515021790555061016082015181600901600d6101000a81
548160ff02191690831515021790555061018082015181600a01556101a082015181600b0155
9050508360036000600854815260200190815260200160002060006101000a81548173ffffffff
ffffffffffffffffffffffff021916908373ffffffffffffffffffffffff160217905550826004600060085481
52602001908152602001600020908051906020019061162b9291906155dc565b508b73ffffffff
ffffffffffffffff168d73ffffffffffffffffffffffff166008547f728d0fed13687f1840de94f5ae640
eae49b43eda26b7ffe97e55e79a0de6e40c8e8e868f8f8f8e8e604051808a815260200189815
26020018867ffffffff168152602001876bffffffff168152602001866bffffffff16
81526020018581526020018481526020018373ffffffff168152602001806020
01828103825283818151815260200191508051906020019080838360005b838110156117335
78082015181840152602081019050611718565b50505050905090810190601f168015611760
5780820380516001836020036101000a031916815260200191505b509a50505050505050505
0505060405180910390a46008549150509b9a505050505050505050505050565b600042600260
008481526020019081526020016000206003015410156117b357600090506117df565b6117dc
42600260008581526020019081526020016000206003015461391b90919063ffffff16565b905
05b919050565b600760009054906101000a900467ffffffff1681565b600061183c611837600
1600760009054906101000a900467ffffffff1667ffffffff1661363c90919063ffffff16565b6
13965565b600760006101000a81548167ffffffff021916908367ffffffff160217905550611
890600760009054906101000a900467ffffffff168360056139d09092919063ffffff16565b611
902576040517f08c379a00
0081526004018080602001828103825260178152602001807f5573657220616c72656164792
07265676973746572656400000000000000000081525060200191505060405180910390fd5b
600760009054906101000a900467ffffffff1690508173ffffffff167f6838f67c
b358c332087b73dbe769a6c869a5f87225236c16dda5d0013a77074c82604051808267ffffff
fff16815260200191505060405180910390a2919050565b600061198e826005613cf69091906
3ffffff16565b156119ae576119a7826005613d6390919063ffffff16565b9050611a08565b6119b
7826117fe565b90508173ffffffff168167ffffffff167f969d438b19b6b5fb3c0
d3c6f16867e519fcec4233799d6be29d1c00f2045ba2660405160405180910390a35b9190505
65b600081600060026000838152602001908152602001600020600301549050600081141580
15611a3c5750804210155b8015611a6057506000801b6002600084815260200190815260200
160002060080154145b611ab5576040517f08c379a00000000000000000000000000000000
0000000000000000000000008152600401808060200182810382526028815260200180615a4
66028913960400191505060405180910390fd5b506000806000611ada600260008881526020
0190815260200160002060040154613e42565b9250925092506000600260008881526020019
081526020016000206006015490506000600260008981526020019081526020016000206007
0154905060008060008590505b6000611b4685600160008f815260200190815260200160002
0613e6890919063ffffff16565b90507ffffffff00000000000000000000000000000000160
001b811415611b795750611c02565b809450611b8585613e42565b909150816bffffffff
169150806bffffffff1690508094508195505050611bc5838761363c90919063ffffff1656

[illegible]

[illegible]

ffff1673ffffffffffffffffffffffff1681526020016001820160009054906101000a900473ffffffffffff
ffffffffffffffff1673ffffffffffffffffffffffff1673ffffffffffffffffffffffff16815260200160028
201548152602001600382015481526020016004820154815260200160058201548152602001
600682015481526020016007820154815260200160088201548152602001600982016000905
4906101000a90046bffffffffffffffff166bffffffffffffffff166bffffffffffffffff1681526020016009
8201600c9054906101000a900460ff1615151515815260200160098201600d9054906101000a
900460ff16151515158152602001600a8201548152602001600b820154815250509050600080
6129ec836101000151613e42565b92509250506000612a1088600081518110612a0357fe5b6
020026020010151613e42565b505090506000600260008b8152602001908152602001600020
600901600c9054906101000a900460ff16905060005b8951811015612d13576000806000612a
698d8581518110612a5c57fe5b6020026020010151613e42565b9250925092508567ffffffffffff
168367ffffffffffff1614612adb576040517f08c379a0000000000000000000000000000000000
0000000000000000000000008152600401808060200182810382526023815260200180615a236
023913960400191505060405180910390fd5b8415612b0957612b02816bffffffffffffffff168c6
1363c90919063ffffff16565b9a50612c9c565b8861010001518d8581518110612b1b57fe5b602
00260200101511415612bde57612b8f612b80886bffffffffffffffff16612b728b6bffffffffffffffff
168d61012001516bffffffffffffffff166136c490919063ffffff16565b61374a90919063ffffff1656
5b8d61363c90919063ffffff16565b9b50612bd7612bc88a61012001516bffffffffffffffff16836b
ffffffffffffffff1661391b90919063ffffff16565b8c61363c90919063ffffff16565b9a50612c9b56
5b612c098961010001518e8681518110612bf357fe5b602002602001015161424890919063ffff
ffff16565b15612c7657612c6f612c60886bffffffffffffffff16612c528b6bffffffffffffffff16856bfff
ffffffffffffffff166136c490919063ffffff16565b61374a90919063ffffff16565b8d61363c90919063
ffffff16565b9b50612c9a565b612c97816bffffffffffffffff168c61363c90919063ffffff16565b9a
505b5b5b8567ffffffffffff168e7f3f2c83616e48a2f8b3c9cc26499e2e5c61643714dac49027d82
cf6732f96a91a848460405180836bffffffffffffffff168152602001826bffffffffffffffff16815260
20019250505060405180910390a35050508080600101915050612a3f565b50612d208a89898
5614466565b50505050509250929050565b81600260008281526020019081526020016000
20600201544210612d9c576040517f08c379a00
0000000000000000000000081526004018080602001828103825260328152602001806158636032
913960400191505060405180910390fd5b6000612da733611978565b90506000805b8451811
015612f27576000612df0868381518110612dc757fe5b6020026020010151600160008a81526
0200190815260200160002061455b90919063ffffff16565b90508015612f1957600080600061
2e19898681518110612e0c57fe5b6020026020010151613e42565b9250925092508667ffffffffffff
ffff168367ffffffffffff1614612e8b576040517f08c379a0000000000000000000000000000000000
0000000000000000000000081526004018080602001828103825260238152602001806157f
76023913960400191505060405180910390fd5b612eac816bffffffffffffffff168761363c90919
063ffffff16565b95508667ffffffffffff168a7f7edae327fe79804b2f38bd490d874a0d188d3ae891
1b303205abe47400086ac4848460405180836bffffffffffffffff168152602001826bffffffffffffffff
fff1681526020019250505060405180910390a35050505b508080600101915050612dad565b5

0612f8933826002600089815260200190815260200160002060010160009054906101000a90
0473ffffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffffff166146069092919063ffffff16565b50
50505050565b60008060009054906101000a900473ffffffffffffffffffffffffffffff16905090565b85
60006002600083815260200190815260200160002060030154905060008114158015612fe65
750804210155b801561300a57506000801b6002600084815260200190815260200160002060
080154145b61305f576040517f08c379a000
000000000000008152600401808060200182810382526028815260200180615a46602891396
0400191505060405180910390fd5b5060026000888152602001908152602001600020600901
600d9054906101000a900460ff166130d9576040517f08c379a00000000000000000000000000
00000000000000000000000000000008152600401808060200182810382526028815260200
18061579e6028913960400191505060405180910390fd5b600186511480156130eb57506001
8551145b613140576040517f08c379a00
0000000000000081526004018080602001828103825260278152602001806159556027913960
400191505060405180910390fd5b600061314b33611978565b90506131a8613183828960008
151811061316157fe5b60200260200101518960008151811061317657fe5b602002602001015
16138da565b600260008b81526020019081526020016000206007015461424890919063ffffff
16565b6131fd576040517f08c379a00
000000000081526004018080602001828103825260318152602001806157c66031913960400
191505060405180910390fd5b61320c88888888888888336146a8565b5061321688611a0d565b
50505050505050505050565b60085481565b60008660026000828152602001908152602001600
020600301544210613298576040517f08c379a000000000000000000000000000000000000
0000000000000000000008152600401808060200182810382526022815260200180615b17602
2913960400191505060405180910390fd5b6132a788888888888888336146a8565b9150509695
505050505050565b6103e881565b60036020528060005260406000206000915054906101000
a900473ffffffffffffffffffffffffffffff1681565b60095481565b60008760026000828152602001908
152602001600020600301544210613364576040517f08c379a00000000000000000000000000
000000000000000000000000000008152600401808060200182810382526022815260200
180615b176022913960400191505060405180910390fd5b61337389898989898989896146a856
5b915050979650505050505050565b6133896141c7565b73ffffffffffffffffffffffffffffff16600080
54906101000a900473ffffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffffff161461344957604
0517f08c379a00815260
04018080602001828103825260208152602001807f4f776e61626c653a2063616c6c65722069
73206e6f7420746865206f776e657281525060200191505060405180910390fd5b600073ffffff
ffffffffffffffffffff168173ffffffffffffffffffffffffffffff1614156134cf576040517f08c379a0000000
00081526004018080602001828103
8252602681526020018061581a6026913960400191505060405180910390fd5b8073ffffffffffffff
ffffffffffff1660008054906101000a900473ffffffffffffffffffffffffffffff1673ffffffffffffffffffffff
ffff167f8be0079c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e06040516
0405180910390a3806000806101000a81548173ffffffffffffffffffffffffffffff021916908373ffffff

[illegible]

[illegible]

[illegible]

[illegible]

1807f4d757374206861766520494420746f20676574204164647265737300000000008152506
0200191505060405180910390fd5b8260000160006001840167ffffffffffffff1667ffffffffffffff16815
260200190815260200160002060009054906101000a900473ffffffffffffffffffffffffffff1690509
2915050565b60008061501484613e42565b9250505060006bffffffffffffffffffffff16816bffffffffffffff
ffff1614156150ab576040517f08c379a000
0000000000000815260040180806020018281038252601f8152602001807f496e73657274696
e67207a65726f206973206e6f7420737570706f7274656400815250602001915050604051809
10390fd5b600160001b84141580156150e257507fffffffffffffffffffffffffffff0000000000000000
0000000160001b8414155b615154576040517f08c379a0000000000000000000000000000000000
0000000000000000000000000815260040180806020018281038252601e8152602001807f
496e73657274696e6720656c656d656e74206973206e6f742076616c696400081525060200
191505060405180910390fd5b61515e85856141cf565b1561516d5760009150506152af565b6
00160001b831415801561519857506000801b85600101600085815260200190815260200160
002054145b156151a75760009150506152af565b6151ba848461424890919063fffffff16565b6
151c85760009150506152af565b5b6000801b85600001600085815260200190815260200160
0020541415615206578460010160008481526020019081526020016000205492506151c9565
b6000808490505b80915086600001600082815260200190815260200160002054905061523b
868261424890919063fffffff16565b61520d57858760000160008481526020019081526020016
000208190555085876001016000838152602001908152602001600020819055508187600101
600088815260200190815260200160002081905550808760000160008881526020019081526
0200160002081905550600193505050505b9392505050565b60606152c58484600085615354
565b90509392505050565b60008073ffffffffffffffffffffffffffff168360000160006001850167ffff
ffffffff1667ffffffffffffff16815260200190815260200160002060009054906101000a900473ffffff
ffffffffffffffffffffffff1673ffffffffffffffffffffffffffff161415905092915050565b6060824710156153
af576040517f08c379a000
81526004018080602001828103825260268152602001806158dc60269139604001915050604
05180910390fd5b6153b8856154fd565b61542a576040517f08c379a000000000000000000000
000000000000000000000000000000000815260040180806020018281038252601d8152
602001807f416464726573733a2063616c6c20746f206e6f6e2d636f6e7472616374000000815
25060200191505060405180910390fd5b600060608673ffffffffffffffffffffffffffff168587604051
8082805190602001908083835b6020831061547a57805182526020820191506020810190506
02083039250615457565b6001836020036101000a0380198251168184511680821785525050
5050505090500191505060006040518083038185875af1925050503d80600081146154dc576
040519150601f19603f3d011682016040523d82523d6000602084013e6154e1565b606091505
b50915091506154f1828286615510565b92505050949350505050565b600080823b90506000
8111915050919050565b60608315615520578290506155d5565b60008351111561553357825
18084602001fd5b816040517f08c379a00
000000000000081526004018080602001828103825283818151815260200191508051906020
019080838360005b8381101561559a57808201518184015260208101905061557f565b50505

050905090810190601f1680156155c75780820380516001836020036101000a031916815260
200191505b509250505060405180910390fd5b9392505050565b82805460018160011615610
1000203166002900490600052602060002090601f016020900481019282601f1061561d57805
160ff191683800117855561564b565b8280016001018555821561564b579182015b828111156
1564a57825182559160200191906001019061562f565b5b509050615658919061575a565b50
90565b50805460018160011615610100020316600290046000825580601f106156825750615
6a1565b601f0160209004906000526020600020908101906156a0919061575a565b5b50565b
604051806101c00160405280600073ffffffffffffffffffffffffffffffff168152602001600073ffffffffffffffff
ffffffffffffffff1681526020016000815260200160008152602001600080191681526020016000
815260200160008152602001600080191681526020016000801916815260200160006bffffffff
ffffffff16815260200160001515815260200160001515815260200160008152602001600081
525090565b5b8082111561577357600081600090555060010161575b565b509056fe5361666
5436173743a2076616c756520646f65736e27742066697420696e20393620626974736e6f742
0616c6c6f77656420746f20736574746c652061756374696f6e2061746f6d6963616c6c797072
6563616c63756c61746553656c6c416d6f756e7453756d20697320616c726561647920746f6f2
0616476616e6365644f6e6c792074686520757365722063616e2063616e63656c2068697320
6f72646572734f776e61626c653a206e6577206f776e657220697320746865207a65726f20616
46472657373746f6b656e732063616e6e6f742062652061756374696f6e656420666f72206672
65656e6f206c6f6e67657220696e206f7264657220706c6163656d656e7420616e642063616e
63656c6174696f6e2070686173655f6d696e427579416d6f756e7473206d7573742062652067
726561746572207468616e2030547279696e6720746f20676574206e657874206f66206c6173
7420656c656d656e74416464726573733a20696e73756666696369656e742062616c616e636
520666f722063616c6c466565206973206e6f7420616c6c6f77656420746f2062652073657420
686967686572207468616e20312e352561756374696f6e20656e642064617465206d7573742
0626520696e20746865206675747572654f6e6c79206f6e65206f726465722063616e2062652
0706c616365642061746f6d6963616c6c79547279696e6720746f20676574206e657874206f6
6206e6f6e2d6578697374656e7420656c656d656e746d696e696d756d42696464696e67416d
6f756e745065724f72646572206973206e6f7420616c6c6f77656420746f206265207a65726f53
6166654d6174683a206d756c7469706c69636174696f6e206f766572666c6f77536166654361
73743a2076616c756520646f65736e27742066697420696e20363420626974736f6e6c792061
6c6c6f77656420746f20636c61696d20666f722073616d65207573657241756374696f6e206e6
f7420696e20736f6c7574696f6e207375626d697373696f6e2070686173657573657220697320
6e6f7420616c6c6f77656420746f20706c6163652073616d65206f726465722074776963656c6
96d6974207072696365206e6f7420626574746572207468616e206d696d696d616c206f66666
57274696d6520706572696f647320617265206e6f7420636f6e6669677572656420636f727265
63746c795361666545524332303a204552433230206f7065726174696f6e20646964206e6f74
20737563636565646e6f206c6f6e67657220696e206f7264657220706c6163656d656e742070
68617365a26469706673582212209d3f3585da1ffe9756c8a2e5f28991662e7f098068faf72dd2
ec014b6bd536c964736f6c634300060c0033", "devdoc": { "kind": "dev", "methods": { "owner()":

```

{ "details": "Returns the address of the current owner." }, "renounceOwnership()": { "details":
"Leaves the contract without owner. It will not be possible to call `onlyOwner` functions
anymore. Can only be called by the current owner. NOTE: Renouncing ownership will leave
the contract without an owner, thereby removing any functionality that is only available to the
owner." }, "transferOwnership(address)": { "details": "Transfers ownership of the contract to a
new account (`newOwner`). Can only be called by the current owner." } }, "version": 1 },
"userdoc": { "kind": "user", "methods": {}, "version": 1 }, "storageLayout": { "storage": [ { "astId":
30, "contract": "contracts/EasyAuction.sol:EasyAuction", "label": "_owner", "offset": 0, "slot":
"0", "type": "t_address" }, { "astId": 1172, "contract": "contracts/EasyAuction.sol:EasyAuction",
"label": "sellOrders", "offset": 0, "slot": "1", "type":
"t_mapping(t_uint256,t_struct(Data)3111_storage)" }, { "astId": 1176, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "auctionData", "offset": 0, "slot": "2", "type":
"t_mapping(t_uint256,t_struct(AuctionData)1168_storage)" }, { "astId": 1180, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "auctionAccessManager", "offset": 0, "slot":
"3", "type": "t_mapping(t_uint256,t_address)" }, { "astId": 1184, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "auctionAccessData", "offset": 0, "slot": "4",
"type": "t_mapping(t_uint256,t_bytes_storage)" }, { "astId": 1186, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "registeredUsers", "offset": 0, "slot": "5",
"type": "t_struct(Data)2920_storage" }, { "astId": 1188, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "numUsers", "offset": 0, "slot": "7", "type":
"t_uint64" }, { "astId": 1190, "contract": "contracts/EasyAuction.sol:EasyAuction", "label":
"auctionCounter", "offset": 0, "slot": "8", "type": "t_uint256" }, { "astId": 1199, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "feeNumerator", "offset": 0, "slot": "9", "type":
"t_uint256" }, { "astId": 1205, "contract": "contracts/EasyAuction.sol:EasyAuction", "label":
"feeReceiverUserId", "offset": 0, "slot": "10", "type": "t_uint64" } ], "types": { "t_address": {
"encoding": "inplace", "label": "address", "numberOfBytes": "20" }, "t_bool": { "encoding":
"inplace", "label": "bool", "numberOfBytes": "1" }, "t_bytes32": { "encoding": "inplace", "label":
"bytes32", "numberOfBytes": "32" }, "t_bytes_storage": { "encoding": "bytes", "label": "bytes",
"numberOfBytes": "32" }, "t_contract(IERC20)478": { "encoding": "inplace", "label": "contract
IERC20", "numberOfBytes": "20" }, "t_mapping(t_address,t_uint64)": { "encoding": "mapping",
"key": "t_address", "label": "mapping(address => uint64)", "numberOfBytes": "32", "value":
"t_uint64" }, "t_mapping(t_bytes32,t_bytes32)": { "encoding": "mapping", "key": "t_bytes32",
"label": "mapping(bytes32 => bytes32)", "numberOfBytes": "32", "value": "t_bytes32" },
"t_mapping(t_uint256,t_address)": { "encoding": "mapping", "key": "t_uint256", "label":
"mapping(uint256 => address)", "numberOfBytes": "32", "value": "t_address" },
"t_mapping(t_uint256,t_bytes_storage)": { "encoding": "mapping", "key": "t_uint256", "label":
"mapping(uint256 => bytes)", "numberOfBytes": "32", "value": "t_bytes_storage" },
"t_mapping(t_uint256,t_struct(AuctionData)1168_storage)": { "encoding": "mapping", "key":
"t_uint256", "label": "mapping(uint256 => struct EasyAuction.AuctionData)", "numberOfBytes":

```

```
"32", "value": "t_struct(AuctionData)1168_storage" },
"t_mapping(t_uint256,t_struct(Data)3111_storage)": { "encoding": "mapping", "key":
"t_uint256", "label": "mapping(uint256 => struct IterableOrderedOrderSet.Data)",
"numberOfBytes": "32", "value": "t_struct(Data)3111_storage" },
"t_mapping(t_uint64,t_address)": { "encoding": "mapping", "key": "t_uint64", "label":
"mapping(uint64 => address)", "numberOfBytes": "32", "value": "t_address" },
"t_struct(AuctionData)1168_storage": { "encoding": "inplace", "label": "struct
EasyAuction.AuctionData", "members": [ { "astId": 1141, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "auctioningToken", "offset": 0, "slot": "0",
"type": "t_contract(IERC20)478" }, { "astId": 1143, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "biddingToken", "offset": 0, "slot": "1", "type":
"t_contract(IERC20)478" }, { "astId": 1145, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "orderCancellationEndDate", "offset": 0,
"slot": "2", "type": "t_uint256" }, { "astId": 1147, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "auctionEndDate", "offset": 0, "slot": "3",
"type": "t_uint256" }, { "astId": 1149, "contract": "contracts/EasyAuction.sol:EasyAuction",
"label": "initialAuctionOrder", "offset": 0, "slot": "4", "type": "t_bytes32" }, { "astId": 1151,
"contract": "contracts/EasyAuction.sol:EasyAuction", "label":
"minimumBiddingAmountPerOrder", "offset": 0, "slot": "5", "type": "t_uint256" }, { "astId": 1153,
"contract": "contracts/EasyAuction.sol:EasyAuction", "label": "interimSumBidAmount", "offset":
0, "slot": "6", "type": "t_uint256" }, { "astId": 1155, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "interimOrder", "offset": 0, "slot": "7", "type":
"t_bytes32" }, { "astId": 1157, "contract": "contracts/EasyAuction.sol:EasyAuction", "label":
"clearingPriceOrder", "offset": 0, "slot": "8", "type": "t_bytes32" }, { "astId": 1159, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "volumeClearingPriceOrder", "offset": 0,
"slot": "9", "type": "t_uint96" }, { "astId": 1161, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "minFundingThresholdNotReached",
"offset": 12, "slot": "9", "type": "t_bool" }, { "astId": 1163, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "isAtomicClosureAllowed", "offset": 13, "slot":
"9", "type": "t_bool" }, { "astId": 1165, "contract": "contracts/EasyAuction.sol:EasyAuction",
"label": "feeNumerator", "offset": 0, "slot": "10", "type": "t_uint256" }, { "astId": 1167, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "minFundingThreshold", "offset": 0, "slot":
"11", "type": "t_uint256" } ], "numberOfBytes": "384" }, "t_struct(Data)2920_storage": {
"encoding": "inplace", "label": "struct IdToAddressBiMap.Data", "members": [ { "astId": 2915,
"contract": "contracts/EasyAuction.sol:EasyAuction", "label": "idToAddress", "offset": 0, "slot":
"0", "type": "t_mapping(t_uint64,t_address)" }, { "astId": 2919, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "addressTold", "offset": 0, "slot": "1", "type":
"t_mapping(t_address,t_uint64)" } ], "numberOfBytes": "64" }, "t_struct(Data)3111_storage": {
"encoding": "inplace", "label": "struct IterableOrderedOrderSet.Data", "members": [ { "astId":
```

```
3106, "contract": "contracts/EasyAuction.sol:EasyAuction", "label": "nextMap", "offset": 0,
"slot": "0", "type": "t_mapping(t_bytes32,t_bytes32)" }, { "astId": 3110, "contract":
"contracts/EasyAuction.sol:EasyAuction", "label": "prevMap", "offset": 0, "slot": "1", "type":
"t_mapping(t_bytes32,t_bytes32)" } ], "numberOfBytes": "64" }, "t_uint256": { "encoding":
"inplace", "label": "uint256", "numberOfBytes": "32" }, "t_uint64": { "encoding": "inplace", "label":
"uint64", "numberOfBytes": "8" }, "t_uint96": { "encoding": "inplace", "label": "uint96",
"numberOfBytes": "12" } } } }
```