```
1 pragma solidity ^0.5.16;
2
   pragma experimental ABIEncoderV2;
3
4
   contract Comp {
       /// @notice EIP-20 token name for this token
5
       string public constant name = "Compound";
6
 8
       /// @notice EIP-20 token symbol for this token
       string public constant symbol = "COMP";
9
10
11
        /// @notice EIP-20 token decimals for this toke
   n
       uint8 public constant decimals = 18:
12
13
14
        /// @notice Total number of tokens in circulati
   on
       uint public constant totalSupply = 100000000e18;
15
    // 10 million Comp
       /// @notice Allowance amounts on behalf of othe
17
   rs
18
       mapping (address => mapping (address => uint9
   6)) internal allowances:
19
       /// @notice Official record of token balances f
   or each account
21
       mapping (address => uint96) internal balances;
22
       /// @notice A record of each accounts delegate
       mapping (address => address) public delegates;
24
       /// @notice A checkpoint for marking number of
26
    votes from a given block
27
       struct Checkpoint {
28
           uint32 fromBlock;
29
           uint96 votes:
30
31
       /// @notice A record of votes checkpoints for e
32
   ach account, by index
       mapping (address => mapping (uint32 => Checkpoi
33
   nt)) public checkpoints;
34
       /// @notice The number of checkpoints for each
35
       mapping (address => uint32) public numCheckpoin
36
   ts:
37
       /// @notice The EIP-712 typehash for the contra
   ct's domain
       bytes32 public constant DOMAIN TYPEHASH = kecca
   k256("EIP712Domain(string name, uint256 chainId, addr
   ess verifyingContract)");
40
       /// @notice The EIP-712 typehash for the delega
41
   tion struct used by the contract
       bytes32 public constant DELEGATION_TYPEHASH = k
   eccak256("Delegation(address delegatee, uint256 nonc
   e, uint256 expiry)");
43
       /// @notice A record of states for signing / va
   lidating signatures
       mapping (address => uint) public nonces;
45
```

```
1 pragma solidity ^0.5.16:
 2 pragma experimental ABIEncoderV2;
 4
   contract Comp {
       /// @notice EIP-20 token name for this token
 5
       string public constant name = "AGORA DEFI";
 6
 8
       /// @notice EIP-20 token symbol for this token
       string public constant symbol = "AGORA";
 9
10
11
       /// @notice EIP-20 token decimals for this toke
   n
       uint8 public constant decimals = 18:
12
13
14
       /// @notice Total number of tokens in circulati
   on
       uint public constant totalSupply = 1000000000e1
15
      // 100 million
16
17
       /// @notice Allowance amounts on behalf of othe
   rs
18
       mapping (address => mapping (address => uint9
   6)) internal allowances:
19
       /// @notice Official record of token balances f
   or each account
21
       mapping (address => uint96) internal balances;
       /// @notice A record of each accounts delegate
24
       mapping (address => address) public delegates;
       /// @notice A checkpoint for marking number of
26
    votes from a given block
27
       struct Checkpoint {
28
           uint32 fromBlock;
29
           uint96 votes:
30
31
       /// @notice A record of votes checkpoints for e
32
   ach account, by index
       mapping (address => mapping (uint32 => Checkpoi
33
   nt)) public checkpoints;
34
       /// @notice The number of checkpoints for each
    account
       mapping (address => uint32) public numCheckpoin
   ts:
37
       /// @notice The EIP-712 typehash for the contra
   ct's domain
       bytes32 public constant DOMAIN TYPEHASH = kecca
   k256("EIP712Domain(string name, uint256 chainId, addr
   ess verifyingContract)");
40
       /// @notice The EIP-712 typehash for the delega
41
   tion struct used by the contract
       bytes32 public constant DELEGATION_TYPEHASH = k
   eccak256("Delegation(address delegatee, uint256 nonc
   e,uint256 expiry)");
43
       /// @notice A record of states for signing / va
   lidating signatures
       mapping (address => uint) public nonces;
```

```
/// @notice An event thats emitted when an acco
                                                              47
                                                                      /// @notice An event thats emitted when an acco
47
    unt changes its delegate
                                                                  unt changes its delegate
       event DelegateChanged(address indexed delegato
                                                                      event DelegateChanged(address indexed delegato
48
    r, address indexed fromDelegate, address indexed to
                                                                  r, address indexed fromDelegate, address indexed to
    Delegate):
                                                                  Delegate):
49
                                                              49
        /// @notice An event thats emitted when a deleg
50
                                                                      /// @notice An event thats emitted when a deleg
    ate account's vote balance changes
                                                                  ate account's vote balance changes
       event DelegateVotesChanged(address indexed dele
                                                                      event DelegateVotesChanged(address indexed dele
51
                                                              51
    gate, uint previousBalance, uint newBalance);
                                                                  gate, uint previousBalance, uint newBalance);
52
                                                              52
       /// @notice The standard EIP-20 transfer event
                                                                      /// @notice The standard EIP-20 transfer event
                                                              53
53
       event Transfer(address indexed from, address in
                                                                      event Transfer(address indexed from, address in
54
                                                              54
    dexed to, uint256 amount);
                                                                  dexed to, uint256 amount);
55
                                                              55
       /// @notice The standard EIP-20 approval event
                                                                      /// @notice The standard EIP-20 approval event
                                                              56
56
       event Approval(address indexed owner, address i
                                                                      event Approval(address indexed owner, address i
57
                                                              57
    ndexed spender, uint256 amount);
                                                                  ndexed spender, uint256 amount);
58
                                                              58
                                                              59
59
         * @notice Construct a new Comp token
60
                                                              60
                                                                       * @notice Construct a new Comp token
         * @param account The initial account to grant
                                                                       * @param account The initial account to grant
61
    all the tokens
                                                                   all the tokens
        * /
                                                                       */
62
                                                              62
       constructor(address account) public {
                                                               63
                                                                      constructor(address account) public {
63
                                                                          balances[account] = uint96(totalSupply);
            balances[account] = uint96(totalSupply);
64
            emit Transfer(address(0), account, totalSup
                                                                          emit Transfer(address(0), account, totalSup
65
                                                              65
    ply);
                                                                  ply);
66
       }
                                                                      }
67
                                                               67
                                                               68
68
69
         * @notice Get the number of tokens `spender` i
                                                                       * @notice Get the number of tokens `spender` i
    s approved to spend on behalf of `account'
                                                                  s approved to spend on behalf of `account`
70
         * @param account The address of the account ho
                                                                       * @param account The address of the account ho
    lding the funds
                                                                  lding the funds
                                                                       ^{\star} @param spender The address of the account sp
71
        * @param spender The address of the account sp
    ending the funds
                                                                  ending the funds
         * @return The number of tokens approved
                                                                       * @return The number of tokens approved
73
                                                               73
       function allowance(address account, address spe
                                                                      function allowance(address account, address spe
74
    nder) external view returns (uint) {
                                                                  nder) external view returns (uint) {
75
            return allowances[account][spender];
                                                               75
                                                                          return allowances[account][spender]:
76
                                                               76
                                                                      }
77
                                                               77
78
                                                               78
         * @notice Approve `spender` to transfer up to
                                                                       * @notice Approve `spender` to transfer up to
                                                               79
     `amount` from `src`
                                                                    `amount` from `src`
80
        * @dev This will overwrite the approval amount
                                                                       * @dev This will overwrite the approval amount
                                                              80
    for `spender
                                                                  for `spender
81
         * and is subject to issues noted [here](http
                                                              81
                                                                       * and is subject to issues noted [here](http
    s://eips.ethereum.org/EIPS/eip-20#approve)
                                                                  s://eips.ethereum.org/EIPS/eip-20#approve)
82
         * @param spender The address of the account wh
                                                              82
                                                                       * @param spender The address of the account wh
    ich may transfer tokens
                                                                  ich may transfer tokens
83
         * @param rawAmount The number of tokens that a
                                                                       * @param rawAmount The number of tokens that a
                                                              83
    re approved (2^256-1 means infinite)
                                                                  re approved (2^256-1 means infinite)
84
         * @return Whether or not the approval succeede
                                                                       * @return Whether or not the approval succeede
                                                              84
   d
                                                                  d
85
                                                              85
86
       function approve(address spender, uint rawAmoun
                                                                      function approve(address spender, uint rawAmoun
                                                              86
    t) external returns (bool) {
                                                                  t) external returns (bool) {
87
           uint96 amount;
                                                              87
                                                                          uint96 amount;
            if (rawAmount == uint(-1)) {
                                                                          if (rawAmount == uint(-1)) {
88
                                                              88
                amount = uint96(-1);
                                                              89
                                                                              amount = uint96(-1);
89
90
                                                              90
91
                amount = safe96(rawAmount, "Comp::appro
                                                                              amount = safe96(rawAmount, "Comp::appro
    ve: amount exceeds 96 bits");
                                                                  ve: amount exceeds 96 bits");
```

```
}
 93
94
             allowances[msg.sender][spender] = amount;
                                                               94
                                                                           allowances[msg.sender][spender] = amount;
95
                                                               95
96
             emit Approval(msg.sender, spender, amount);
                                                               96
                                                                           emit Approval(msg.sender, spender, amount);
97
             return true:
                                                               97
                                                                           return true:
98
                                                               98
        }
                                                                       }
99
                                                               99
         ^{\ast} @notice Get the number of tokens held by the
                                                                        ^{\star} @notice Get the number of tokens held by the
101
                                                               101
     account`
                                                                    `account`
102
          * @param account The address of the account to
                                                              102
                                                                        * @param account The address of the account to
    get the balance of
                                                                   get the balance of
         * @return The number of tokens held
                                                                        * @return The number of tokens held
103
                                                              103
104
                                                               104
105
        function balanceOf(address account) external vi
                                                              105
                                                                       function balanceOf(address account) external vi
    ew returns (uint) {
                                                                   ew returns (uint) {
106
             return balances[account];
                                                               106
                                                                           return balances[account];
107
                                                               107
        }
                                                               108
108
109
         * @notice Transfer `amount` tokens from `msg.s
                                                                        * @notice Transfer `amount` tokens from `msg.s
110
    ender `to `dst`
                                                                   ender `to `dst`
111
         * @param dst The address of the destination ac
                                                              111
                                                                        * @param dst The address of the destination ac
    count
                                                                   count
         * @param rawAmount The number of tokens to tra
                                                                        * @param rawAmount The number of tokens to tra
112
                                                               112
    nsfer
                                                                   nsfer
          * @return Whether or not the transfer succeede
                                                                        * @return Whether or not the transfer succeede
113
                                                              113
    d
                                                                   d
114
                                                               114
        function transfer(address dst, uint rawAmount)
                                                                       function transfer(address dst, uint rawAmount)
115
                                                              115
     external returns (bool) {
                                                                    external returns (bool) {
            uint96 amount = safe96(rawAmount, "Comp::tr
                                                                           uint96 amount = safe96(rawAmount, "Comp::tr
116
                                                               116
    ansfer: amount exceeds 96 bits");
                                                                   ansfer: amount exceeds 96 bits");
117
            transferTokens(msg.sender, dst, amount);
                                                              117
                                                                           transferTokens(msg.sender, dst, amount):
118
             return true;
                                                              118
                                                                           return true;
119
        }
                                                               119
120
                                                              120
121
                                                               121
         * @notice Transfer `amount` tokens from `src`
                                                                        * @notice Transfer `amount` tokens from `src`
122
                                                               122
      to `dst`
123
          * @param src The address of the source account
                                                              123
                                                                        * @param src The address of the source account
          * @param dst The address of the destination ac
                                                                        * @param dst The address of the destination ac
124
                                                              124
                                                                   count
125
         * @param rawAmount The number of tokens to tra
                                                              125
                                                                        * @param rawAmount The number of tokens to tra
    nsfer
                                                                   nsfer
         * @return Whether or not the transfer succeede
                                                                        * @return Whether or not the transfer succeede
                                                               126
    d
127
                                                               127
        function transferFrom(address src, address dst,
                                                                       function transferFrom(address src, address dst,
128
                                                              128
    uint rawAmount) external returns (bool) {
                                                                   uint rawAmount) external returns (bool) {
129
            address spender = msg.sender;
                                                              129
                                                                           address spender = msg.sender;
130
            uint96 spenderAllowance = allowances[src][s
                                                              130
                                                                           uint96 spenderAllowance = allowances[src][s
    pender1;
                                                                   pender];
             uint96 amount = safe96(rawAmount, "Comp::ap
                                                                           uint96 amount = safe96(rawAmount, "Comp::ap
    prove: amount exceeds 96 bits");
                                                                   prove: amount exceeds 96 bits");
                                                               132
                                                              133
             if (spender != src && spenderAllowance != u
                                                                           if (spender != src && spenderAllowance != u
    int96(-1)) {
                                                                   int96(-1)) {
                 uint96 newAllowance = sub96(spenderAllo
                                                                               uint96 newAllowance = sub96(spenderAllo
134
                                                              134
    wance, amount, "Comp::transferFrom: transfer amount
                                                                   wance, amount, "Comp::transferFrom: transfer amount
    exceeds spender allowance");
                                                                   exceeds spender allowance");
135
                 allowances[src][spender] = newAllowanc
                                                              135
                                                                               allowances[src][spender] = newAllowanc
                                                                   e;
136
                                                              136
137
                 emit Approval(src, spender, newAllowanc
                                                              137
                                                                                emit Approval(src, spender, newAllowanc
    e);
                                                                   e);
```

```
}
140
             _transferTokens(src, dst, amount);
                                                               140
                                                                            _transferTokens(src, dst, amount);
             return true;
                                                                           return true;
141
                                                               141
142
        }
                                                               142
                                                                       }
143
                                                               143
        /**
144
                                                               144
145
          * @notice Delegate votes from `msg.sender` to
                                                               145
                                                                        * @notice Delegate votes from `msg.sender` to
      delegatee
                                                                     `delegatee`
146
          * @param delegatee The address to delegate vot
                                                              146
                                                                        * @param delegatee The address to delegate vot
     es to
                                                                   es to
147
                                                              147
        function delegate(address delegatee) public {
                                                                       function delegate(address delegatee) public {
148
                                                              148
149
             return _delegate(msg.sender, delegatee);
                                                               149
                                                                           return _delegate(msg.sender, delegatee);
151
                                                               151
152
          ^{\star} @notice Delegates votes from signatory to `d
                                                                        ^{\ast} @notice Delegates votes from signatory to 'd
153
     elegatee`
                                                                   elegatee
          * @param delegatee The address to delegate vot
                                                                        ^{\star} @param delegatee The address to delegate vot
154
                                                              154
     es to
                                                                   es to
          * @param nonce The contract state required to
                                                                        * @param nonce The contract state required to
155
                                                               155
     match the signature
                                                                    match the signature
          * @param expiry The time at which to expire th
                                                              156
                                                                         * @param expiry The time at which to expire th
156
     e signature
                                                                   e signature
157
         * @param v The recovery byte of the signature
                                                               157
                                                                        * @param v The recovery byte of the signature
          * @param r Half of the ECDSA signature pair
                                                                         * @param r Half of the ECDSA signature pair
158
                                                               158
          * @param s Half of the ECDSA signature pair
                                                                        * @param s Half of the ECDSA signature pair
160
                                                               160
                                                                        */
        function delegateBySig(address delegatee, uint
                                                                        function delegateBySig(address delegatee, uint
     nonce, uint expiry, uint8 v, bytes32 r, bytes32 s)
                                                                    nonce, uint expiry, uint8 v, bytes32 r, bytes32 s)
     public {
                                                                   public {
             bytes32 domainSeparator = keccak256(abi.enc
                                                                           bytes32 domainSeparator = keccak256(abi.enc
162
                                                               162
     ode(DOMAIN_TYPEHASH, keccak256(bytes(name)), getCha
                                                                   ode(DOMAIN_TYPEHASH, keccak256(bytes(name)), getCha
     inId(), address(this)));
                                                                   inId(), address(this)));
163
            bvtes32 structHash = keccak256(abi.encode(D)
                                                              163
                                                                           bvtes32 structHash = keccak256(abi.encode(D
     ELEGATION_TYPEHASH, delegatee, nonce, expiry));
                                                                   ELEGATION_TYPEHASH, delegatee, nonce, expiry));
            bytes32 digest = keccak256(abi.encodePacked
                                                                           bytes32 digest = keccak256(abi.encodePacked
164
                                                              164
     ("\x19\x01", domainSeparator, structHash));
                                                                   ("\x19\x01", domainSeparator, structHash));
165
            address signatory = ecrecover(digest, v, r,
                                                               165
                                                                           address signatory = ecrecover(digest, v, r,
    s);
                                                                   s);
166
            require(signatory != address(0), "Comp::del
                                                              166
                                                                           require(signatory != address(0), "Comp::del
     egateBySig: invalid signature");
                                                                   egateBySig: invalid signature");
167
             require(nonce == nonces[signatory]++, "Com
                                                                           require(nonce == nonces[signatory]++, "Com
     p::delegateBySig: invalid nonce");
                                                                   p::delegateBySig: invalid nonce");
             require(now <= expiry, "Comp::delegateBySi</pre>
                                                                           require(now <= expiry, "Comp::delegateBySi</pre>
     g: signature expired");
                                                                   g: signature expired");
169
             return _delegate(signatory, delegatee);
                                                              169
                                                                           return _delegate(signatory, delegatee);
170
        }
                                                               170
                                                                       3
171
                                                               171
172
          * @notice Gets the current votes balance for `
                                                                        * @notice Gets the current votes balance for `
173
    account
                                                                   account
174
          * @param account The address to get votes bala
                                                              174
                                                                        * @param account The address to get votes bala
                                                                   nce
          * @return The number of current votes for `acc
                                                                         * @return The number of current votes for `acc
175
                                                              175
    ount:
                                                                   ount `
176
                                                               176
        function getCurrentVotes(address account) exter
                                                                        function getCurrentVotes(address account) exter
    nal view returns (uint96) {
                                                                   nal view returns (uint96) {
            uint32 nCheckpoints = numCheckpoints[accoun
                                                                           uint32 nCheckpoints = numCheckpoints[accoun
178
                                                              178
                                                                   t1;
179
             return nCheckpoints > 0 ? checkpoints[accou
                                                              179
                                                                           return nCheckpoints > 0 ? checkpoints[accou
     nt][nCheckpoints - 1].votes : 0;
                                                                   nt][nCheckpoints - 1].votes : 0;
180
        }
                                                               180
181
                                                              181
```

```
* @notice Determine the prior number of votes
                                                                       * @notice Determine the prior number of votes
183
                                                              183
     for an account as of a block number
                                                                    for an account as of a block number
         * @dev Block number must be a finalized block
                                                                       * @dev Block number must be a finalized block
     or else this function will revert to prevent misin
                                                                   or else this function will revert to prevent misin
    formation.
                                                                   formation.
         * @param account The address of the account to
                                                                       * @param account The address of the account to
185
    check
                                                                   check
         * @naram blockNumber The block number to get t
                                                                        * @param blockNumber The block number to get t
186
                                                              186
    he vote balance at
                                                                   he vote balance at
187
         * @return The number of votes the account had
                                                                        * @return The number of votes the account had
     as of the given block
                                                                    as of the given block
188
                                                              188
189
        function getPriorVotes(address account, uint bl
                                                              189
                                                                       function getPriorVotes(address account, uint bl
    ockNumber) public view returns (uint96) {
                                                                   ockNumber) public view returns (uint96) {
            require(blockNumber < block.number, "Comp::</pre>
                                                                          require(blockNumber < block.number, "Comp::</pre>
                                                              190
190
    getPriorVotes: not yet determined");
                                                                   getPriorVotes: not yet determined");
191
                                                              191
192
            uint32 nCheckpoints = numCheckpoints[accoun
                                                              192
                                                                           uint32 nCheckpoints = numCheckpoints[accoun
    t];
                                                                   t];
193
            if (nCheckpoints == 0) {
                                                              193
                                                                           if (nCheckpoints == 0) {
                return 0:
                                                              194
                                                                               return 0:
195
                                                              195
            }
                                                                           }
            // First check most recent balance
                                                                           // First check most recent balance
197
                                                              197
            if (checkpoints[account][nCheckpoints - 1].
                                                                           if (checkpoints[account][nCheckpoints - 1].
    fromBlock <= blockNumber) {</pre>
                                                                   fromBlock <= blockNumber) {</pre>
199
                return checkpoints[account][nCheckpoint
                                                              199
                                                                               return checkpoints[account][nCheckpoint
    s - 1].votes;
                                                                   s - 1].votes;
200
            }
                                                                           }
201
                                                              201
202
            // Next check implicit zero balance
                                                              202
                                                                           // Next check implicit zero balance
            if (checkpoints[account][0].fromBlock > blo
                                                                           if (checkpoints[account][0].fromBlock > blo
203
                                                                   ckNumber) {
    ckNumber) {
204
                 return 0:
                                                              204
                                                                               return 0:
205
            }
                                                              205
                                                                           }
206
                                                              206
207
            uint32 lower = 0;
                                                              207
                                                                           uint32 lower = 0;
208
            uint32 upper = nCheckpoints - 1;
                                                              208
                                                                           uint32 upper = nCheckpoints - 1;
209
            while (upper > lower) {
                                                              209
                                                                           while (upper > lower) {
                uint32 center = upper - (upper - lower)
                                                                               uint32 center = upper - (upper - lower)
    / 2; // ceil, avoiding overflow
                                                                  / 2; // ceil, avoiding overflow
211
                Checkpoint memory cp = checkpoints[acco
                                                              211
                                                                               Checkpoint memory cp = checkpoints[acco
    unt][center];
                                                                  unt][center];
212
                if (cp.fromBlock == blockNumber) {
                                                              212
                                                                               if (cp.fromBlock == blockNumber) {
                                                              213
213
                    return cp.votes;
                                                                                   return cp.votes;
                } else if (cp.fromBlock < blockNumber)</pre>
                                                              214
                                                                               } else if (cp.fromBlock < blockNumber)</pre>
214
     {
                                                                    {
215
                     lower = center;
                                                              215
                                                                                   lower = center;
216
                } else {
                                                              216
                                                                               } else {
                     upper = center - 1;
                                                                                   upper = center - 1;
                                                              217
218
                                                              218
                                                                               }
219
            }
                                                              219
                                                                           3
220
             return checkpoints[account][lower].votes;
                                                              220
                                                                           return checkpoints[account][lower].votes;
221
        function _delegate(address delegator, address d
                                                                       function delegate(address delegator, address d
    elegatee) internal {
                                                                   elegatee) internal {
224
            address currentDelegate = delegates[delegat
                                                                           address currentDelegate = delegates[delegat
    orl;
                                                                   orl;
                                                                           uint96 delegatorBalance = balances[delegato
            uint96 delegatorBalance = balances[delegato
                                                              225
225
    r];
                                                                   r1;
226
            delegates[delegator] = delegatee;
                                                              226
                                                                           delegates[delegator] = delegatee;
                                                              227
             emit DelegateChanged(delegator, currentDele
                                                                           emit DelegateChanged(delegator, currentDele
                                                              228
    gate, delegatee);
                                                                   gate, delegatee);
```

```
_moveDelegates(currentDelegate, delegatee,
            moveDelegates(currentDelegate, delegatee,
     delegatorBalance);
                                                                   delegatorBalance):
                                                              231
233
        function _transferTokens(address src, address d
                                                              233
                                                                      function _transferTokens(address src, address d
    st, uint96 amount) internal {
                                                                  st, uint96 amount) internal {
            require(src != address(0), "Comp::_transfer
                                                                          require(src != address(0), "Comp::_transfer
234
    Tokens: cannot transfer from the zero address"):
                                                                  Tokens: cannot transfer from the zero address");
            require(dst != address(0), "Comp::_transfer
                                                                          require(dst != address(0), "Comp::_transfer
235
    Tokens: cannot transfer to the zero address");
                                                                  Tokens: cannot transfer to the zero address");
                                                              236
236
237
            balances[src] = sub96(balances[src], amoun
                                                              237
                                                                          balances[src] = sub96(balances[src], amoun
    t, "Comp:: transferTokens: transfer amount exceeds
                                                                  t, "Comp:: transferTokens: transfer amount exceeds
                                                                   halance");
     balance"):
            balances[dst] = add96(balances[dst], amoun
                                                                          balances[dst] = add96(balances[dst], amoun
    t, "Comp::_transferTokens: transfer amount overflow
                                                                  t, "Comp::_transferTokens: transfer amount overflow
    s");
                                                                  s");
239
            emit Transfer(src, dst, amount);
                                                              239
                                                                          emit Transfer(src, dst, amount);
240
                                                              240
241
            moveDelegates(delegates[src], delegates[ds
                                                              241
                                                                          moveDelegates(delegates[src], delegates[ds
    t], amount);
                                                                  t], amount):
                                                              242
242
        }
                                                                      }
243
                                                              243
244
        function moveDelegates(address srcRep, address
                                                              244
                                                                      function moveDelegates(address srcRep, address
    dstRep, uint96 amount) internal {
                                                                  dstRep, uint96 amount) internal {
            if (srcRep != dstRep && amount > 0) {
                                                                          if (srcRep != dstRep && amount > 0) {
245
                                                              245
246
                if (srcRep != address(0)) {
                                                              246
                                                                              if (srcRep != address(0)) {
                     uint32 srcRepNum = numCheckpoints[s
                                                                                   uint32 srcRepNum = numCheckpoints[s
247
                                                              247
    rcRep1:
                                                                  rcRepl:
248
                    uint96 srcRepOld = srcRepNum > 0 ?
                                                              248
                                                                                   uint96 srcRepOld = srcRepNum > 0 ?
     checkpoints[srcRep][srcRepNum - 1].votes : 0;
                                                                   checkpoints[srcRep][srcRepNum - 1].votes : 0;
                    uint96 srcRepNew = sub96(srcRepOld,
                                                                                  uint96 srcRepNew = sub96(srcRepOld,
                                                              249
249
    amount, "Comp::_moveVotes: vote amount underflow
                                                                  amount, "Comp::_moveVotes: vote amount underflow
250
                     writeCheckpoint(srcRep, srcRepNum,
                                                              250
                                                                                   writeCheckpoint(srcRep, srcRepNum,
    srcRepOld, srcRepNew);
                                                                  srcRepOld, srcRepNew);
251
                }
                                                              251
                                                                              }
252
                                                              252
253
                 if (dstRep != address(0)) {
                                                              253
                                                                              if (dstRep != address(0)) {
254
                     uint32 dstRepNum = numCheckpoints[d
                                                              254
                                                                                   uint32 dstRepNum = numCheckpoints[d
    stRep];
                                                                  stRep];
                    uint96 dstRepOld = dstRepNum > 0 ?
                                                              255
                                                                                   uint96 dstRepOld = dstRepNum > 0 ?
255
     checkpoints[dstRep][dstRepNum - 1].votes : 0;
                                                                   checkpoints[dstRep][dstRepNum - 1].votes : 0;
                    uint96 dstRepNew = add96(dstRepOld,
                                                                                  uint96 dstRepNew = add96(dstRepOld,
256
                                                              256
    amount, "Comp::_moveVotes: vote amount overflows");
                                                                  amount, "Comp::_moveVotes: vote amount overflows");
                     writeCheckpoint(dstRep, dstRepNum,
                                                                                   writeCheckpoint(dstRep, dstRepNum,
    dstRepOld, dstRepNew);
                                                                  dstRepOld, dstRepNew);
258
                }
                                                              258
                                                                              }
259
            }
                                                              259
                                                                          }
                                                                      }
261
                                                              261
        function writeCheckpoint(address delegatee, ui
                                                                      function writeCheckpoint(address delegatee, ui
    nt32 nCheckpoints, uint96 oldVotes, uint96 newVote
                                                                  nt32 nCheckpoints, uint96 oldVotes, uint96 newVote
    s) internal {
                                                                  s) internal {
          uint32 blockNumber = safe32(block.number, "Co
                                                                        uint32 blockNumber = safe32(block.number, "Co
263
    mp::_writeCheckpoint: block number exceeds 32 bit
                                                                  mp::_writeCheckpoint: block number exceeds 32 bit
                                                                  s"):
                                                              264
          if (nCheckpoints > 0 && checkpoints[delegate
                                                                        if (nCheckpoints > 0 && checkpoints[delegate
265
    e][nCheckpoints - 1].fromBlock == blockNumber) {
                                                                  e][nCheckpoints - 1].fromBlock == blockNumber) {
              checkpoints[delegatee][nCheckpoints - 1].
                                                                            checkpoints[delegatee][nCheckpoints - 1].
    votes = newVotes;
                                                                  votes = newVotes;
267
                                                              267
268
              checkpoints[delegatee][nCheckpoints] = Ch
                                                              268
                                                                            checkpoints[delegatee][nCheckpoints] = Ch
    eckpoint(blockNumber, newVotes);
                                                                  eckpoint(blockNumber, newVotes);
269
              numCheckpoints[delegatee] = nCheckpoints
                                                              269
                                                                            numCheckpoints[delegatee] = nCheckpoints
     + 1;
                                                                   + 1;
```

```
270
                                                              270
272
          emit DelegateVotesChanged(delegatee, oldVote
                                                              272
                                                                         emit DelegateVotesChanged(delegatee, oldVote
    s, newVotes);
                                                                   s, newVotes);
273
                                                              273
275
        function safe32(uint n, string memory errorMess
                                                              275
                                                                       function safe32(uint n, string memory errorMess
    age) internal pure returns (uint32) {
                                                                   age) internal pure returns (uint32) {
276
            require(n < 2**32, errorMessage);</pre>
                                                              276
                                                                           require(n < 2**32, errorMessage);</pre>
            return uint32(n);
                                                                           return uint32(n);
277
                                                              277
278
                                                              278
279
                                                              279
        function safe96(uint n, string memory errorMess
                                                                       function safe96(uint n, string memory errorMess
280
                                                              280
    age) internal pure returns (uint96) {
                                                                   age) internal pure returns (uint96) {
            require(n < 2**96, errorMessage);</pre>
                                                                           require(n < 2**96, errorMessage);</pre>
282
            return uint96(n);
                                                              282
                                                                           return uint96(n);
283
                                                              283
284
                                                              284
285
        function add96(uint96 a, uint96 b, string memor
                                                                       function add96(uint96 a, uint96 b, string memor
    y errorMessage) internal pure returns (uint96) {
                                                                  y errorMessage) internal pure returns (uint96) {
            uint96 c = a + b;
                                                                           uint96 c = a + b;
                                                              286
286
287
            require(c >= a, errorMessage);
                                                              287
                                                                           require(c >= a, errorMessage);
            return c;
                                                              288
                                                                           return c;
289
                                                              289
        }
                                                                       }
290
                                                              290
291
        function sub96(uint96 a, uint96 b, string memor
                                                                       function sub96(uint96 a, uint96 b, string memor
    y errorMessage) internal pure returns (uint96) {
                                                                   y errorMessage) internal pure returns (uint96) {
292
            require(b <= a, errorMessage);</pre>
                                                              292
                                                                           require(b <= a, errorMessage);</pre>
293
            return a - b;
                                                              293
                                                                           return a - b;
294
        }
                                                              294
                                                                       }
295
                                                              295
        function getChainId() internal pure returns (ui
                                                                       function getChainId() internal pure returns (ui
                                                              296
296
    nt) {
                                                                   nt) {
297
            uint256 chainId;
                                                              297
                                                                           uint256 chainId;
298
            assembly { chainId := chainid() }
                                                              298
                                                                           assembly { chainId := chainid() }
             return chainId;
                                                                           return chainId;
299
                                                              299
300
                                                              300
301 }
                                                              301 }
302
                                                              302
```