Module sui_system::validator_set

Event containing staking and rewards related information of each validator, emitted during epoch advancement.

V2 of ValidatorEpochInfoEvent containing more information about the validator.

Event emitted every time a new validator joins the committee. The epoch value corresponds to the first epoch this change takes place.

Event emitted every time a validator leaves the committee. The epoch value corresponds to the first epoch this change takes place.

Key for the extra_fields bag to store the start epoch of allowing admission of new validators based on a minimum voting power rather than a minimum stake.

Called by sui_system to add a new validator candidate.

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

return (min, low, very low voting power) thresholds

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

Update the validator set at the end of epoch. It does the following things:

This function does the following:

Effectuate pending next epoch metadata if they are staged.

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

Get the total number of validators in the next epoch.

Returns true iff the address exists in active validators.

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

Checks whether new_validator is duplicate with any currently pending validators.

Get mutable reference to either a candidate or an active validator by address.

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

Remove validator from self and return the amount of stake that was removed

Sort all the pending removal indexes.

Process all active validators' pending stake deposits and withdraws.

Calculate the total active validator stake.

Process the pending stake changes for each validator.

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

Sum up the total stake of a given list of validator addresses.

Return the active validators in self

Returns true if the addr is a validator candidate.

Returns true if addr is an active validator

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

Return true if addr is currently an at-risk validator below the minimum stake for removal

Struct

```bash

```

Event containing staking and rewards related information of each validator, emitted during epoch advancement.

```bash

```

V2 of ValidatorEpochInfoEvent containing more information about the validator.

```bash

```

Event emitted every time a new validator joins the committee. The epoch value corresponds to the first epoch this change takes place.

```bash
```

Event emitted every time a validator leaves the committee. The epoch value corresponds to the first epoch this change takes place.

```bash

```

Key for the extra_fields bag to store the start epoch of allowing admission of new validators based on a minimum voting power rather than a minimum stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Called by sui_system to add a new validator candidate.

`bash (package)`

`bash (package)`

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

`bash (package)`

`bash (package)`

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

`bash (package)`

`bash (package)`

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash
```

```bash
```

return (min, low, very low voting power) thresholds

```bash
```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash

```

```
bash (package)

bash (package)
```

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash

```

Struct

Event containing staking and rewards related information of each validator, emitted during epoch advancement.

```bash

```

V2 of ValidatorEpochInfoEvent containing more information about the validator.

```bash

```

Event emitted every time a new validator joins the committee. The epoch value corresponds to the first epoch this change takes place.

```bash

```

Event emitted every time a validator leaves the committee. The epoch value corresponds to the first epoch this change takes place.

```bash

```

Key for the extra_fields bag to store the start epoch of allowing admission of new validators based on a minimum voting power rather than a minimum stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system to add a new validator candidate.

bash (package)

bash (package)

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

bash (package)

bash (package)

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

bash (package)

bash (package)

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash
```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Struct

V2 of ValidatorEpochInfoEvent containing more information about the validator.

```bash

```

Event emitted every time a new validator joins the committee. The epoch value corresponds to the first epoch this change takes place.

```bash

```

Event emitted every time a validator leaves the committee. The epoch value corresponds to the first epoch this change takes place.

```bash

```

Key for the extra_fields bag to store the start epoch of allowing admission of new validators based on a minimum voting power rather than a minimum stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Called by sui_system to add a new validator candidate.

`bash (package)`

`bash (package)`

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

`bash (package)`

`bash (package)`

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

`bash (package)`

`bash (package)`

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash
```

```bash
```

return (min, low, very low voting power) thresholds

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

`bash (package)`

`bash (package)`

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

`bash (package)`

```
bash (package)
```

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

Update the validator set at the end of epoch. It does the following things:

```
bash (package)
```

```
bash (package)
```

This function does the following:

```bash
```

```bash
```

Effectuate pending next epoch metadata if they are staged.

```bash
```

```bash
```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

`bash (package)`

`bash (package)`

Get the total number of validators in the next epoch.

`bash (package)`

`bash (package)`

Returns true iff the address exists in active validators.

`bash (package)`

`bash (package)`

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash
```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash
```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Struct

Event emitted every time a new validator joins the committee. The epoch value corresponds to the first epoch this change takes place.

```bash

```

Event emitted every time a validator leaves the committee. The epoch value corresponds to the first epoch this change takes place.

```bash
```

Key for the extra_fields bag to store the start epoch of allowing admission of new validators based on a minimum voting power rather than a minimum stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system to add a new validator candidate.

bash (package)

bash (package)

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

bash (package)

bash (package)

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

bash (package)

bash (package)

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

`bash (package)`

`bash (package)`

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash

```

`bash (package)`

`bash (package)`

```bash

```

```bash

```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash
```

Struct

Event emitted every time a validator leaves the committee. The epoch value corresponds to the first epoch this change takes place.

```bash

```

Key for the extra_fields bag to store the start epoch of allowing admission of new validators based on a minimum voting power rather than a minimum stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system to add a new validator candidate.

bash (package)

bash (package)

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

bash (package)

bash (package)

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

bash (package)

bash (package)

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash
```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```
bash (package)
```

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash
```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash
```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)

bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash

```

Struct

Key for the extra_fields bag to store the start epoch of allowing admission of new validators based on a minimum voting power rather than a minimum stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system to add a new validator candidate.

bash (package)

bash (package)

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

bash (package)

bash (package)

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

bash (package)

bash (package)

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash
```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

`bash (package)`

`bash (package)`

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash

```

`bash (package)`

`bash (package)`

```bash
```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash
```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Constants

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system to add a new validator candidate.

bash (package)

bash (package)

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

bash (package)

bash (package)

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

bash (package)

bash (package)

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash
```

Function

`bash (package)`

`bash (package)`

Called by sui_system to add a new validator candidate.

`bash (package)`

`bash (package)`

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

`bash (package)`

`bash (package)`

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

`bash (package)`

`bash (package)`

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

`bash (package)`

`bash (package)`

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

`bash (package)`

`bash (package)`

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

`bash (package)`

`bash (package)`

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

`bash (package)`

`bash (package)`

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

Update the validator set at the end of epoch. It does the following things:

```
bash (package)

bash (package)
```

This function does the following:

```bash
```

```bash
```

Effectuate pending next epoch metadata if they are staged.

```bash
```

```bash
```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash
```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Called by sui_system to add a new validator candidate.

bash (package)

bash (package)

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

bash (package)

bash (package)

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

bash (package)

bash (package)

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash
```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

`bash (package)`

`bash (package)`

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

`bash (package)`

`bash (package)`

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

`bash (package)`

`bash (package)`

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

Update the validator set at the end of epoch. It does the following things:

`bash (package)`

`bash (package)`

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash
```

```bash
```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

Get the total number of validators in the next epoch.

```
bash (package)
```

```
bash (package)
```

Returns true iff the address exists in active validators.

```
bash (package)
```

```
bash (package)
```

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

`bash (package)`

`bash (package)`

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Called by sui_system to remove a validator candidate, and move them to inactive_validators.

bash (package)

bash (package)

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

bash (package)

bash (package)

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash
```

```bash
```

Effectuate pending next epoch metadata if they are staged.

```bash
```

```bash
```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash
```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash
```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash

```

Function

Called by sui_system to add a new validator to pending_active_validators, which will be processed at the end of epoch.

`bash (package)`

`bash (package)`

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash
```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash
```

```bash
```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Return true if a candidate validator with stake will have sufficeint voting power to join the validator set

```bash

```

```bash

```

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

return (min, low, very low voting power) thresholds

```bash

```

```bash

```

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

```
bash (package)
```

```
bash (package)
```

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

Update the validator set at the end of epoch. It does the following things:

```
bash (package)
```

```
bash (package)
```

This function does the following:

~~~bash

```bash

Effectuate pending next epoch metadata if they are staged.

```bash

```bash

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```bash

```bash

```bash

```bash
~~~

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash
```

```bash
```

`bash (package)`

`bash (package)`

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash
```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

bash (package)

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```bash
```

```bash
```

Function

Called by sui_system , to remove a validator. The index of the validator is added to pending_removals and will be processed at the end of epoch. Only an active validator can request to be removed.

```
bash (package)
```

bash (package)

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Get the total number of validators in the next epoch.

`bash (package)`

`bash (package)`

Returns true iff the address exists in active validators.

`bash (package)`

`bash (package)`

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)

bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash
```

Function

Called by sui_system , to add a new stake to the validator. This request is added to the validator's staking pool's pending stake entries, processed at the end of the epoch. Aborts in case the staking amount is smaller than MIN_STAKING_THRESHOLD

bash (package)

bash (package)

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash
```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Called by sui_system , to withdraw some share of a stake from the validator. The share to withdraw is denoted by principal_withdraw_amount. One of two things occurs in this function:

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Get the total number of validators in the next epoch.

`bash (package)`

`bash (package)`

Returns true iff the address exists in active validators.

`bash (package)`

`bash (package)`

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the

Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash
```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash
```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

bash (package)

bash (package)

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

`bash (package)`

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)

bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash
```

Function

`bash (package)`

`bash (package)`

Update the validator set at the end of epoch. It does the following things:

`bash (package)`

`bash (package)`

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

~~~
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
~~~

```bash

```

```bash

```

~~~
bash (package)

bash (package)
~~~

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

~~~
bash (package)

bash (package)
~~~

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

Function

Update the validator set at the end of epoch. It does the following things:

bash (package)

bash (package)

This function does the following:

```bash
```

```bash
```

Effectuate pending next epoch metadata if they are staged.

```bash
```

```bash
```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash

```

`bash (package)`

`bash (package)`

Get the total number of validators in the next epoch.

`bash (package)`

`bash (package)`

Returns true iff the address exists in active validators.

`bash (package)`

`bash (package)`

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

`bash (package)`

`bash (package)`

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash

```

Function

This function does the following:

```bash

```

```bash

```

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Effectuate pending next epoch metadata if they are staged.

```bash

```

```bash

```

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

`bash (package)`

`bash (package)`

Returns true iff the address exists in active validators.

`bash (package)`

`bash (package)`

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

`bash (package)`

`bash (package)`

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash
```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)

bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash
```

Function

Called by sui_system to derive reference gas price for the new epoch. Derive the reference gas price based on the gas price quote submitted by each validator. The returned gas price should be greater than or equal to 2/3 of the validators submitted gas price, weighted by stake.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```
bash (package)
```

```
bash (package)
```

Get the total number of validators in the next epoch.

```
bash (package)
```

```
bash (package)
```

Returns true iff the address exists in active validators.

```
bash (package)
```

```
bash (package)
```

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

```
bash (package)
```

```
bash (package)
```

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

```
bash (package)

bash (package)
```

Get the total number of validators in the next epoch.

```
bash (package)

bash (package)
```

Returns true iff the address exists in active validators.

```
bash (package)

bash (package)
```

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

```
bash (package)

bash (package)
```

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash

```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash
```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Function

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Get the total number of validators in the next epoch.

`bash (package)`

`bash (package)`

Returns true iff the address exists in active validators.

```
bash (package)
```

```
bash (package)
```

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

bash (package)

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

```
bash (package)

bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash
```

Function

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash
```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

```bash

```

```bash

```

```bash

```

```bash

```

```bash

```

```bash
```

`bash (package)`

`bash (package)`

Get the total number of validators in the next epoch.

`bash (package)`

`bash (package)`

Returns true iff the address exists in active validators.

`bash (package)`

`bash (package)`

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

`bash (package)`

`bash (package)`

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

Function

```bash
```

```bash
```

```bash
```

```bash
```

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash
```

```
bash (package)

bash (package)
```

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)

bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)

bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash
```

Function

```bash
```

```bash
```

```
bash (package)

bash (package)
```

Get the total number of validators in the next epoch.

```
bash (package)

bash (package)
```

Returns true iff the address exists in active validators.

```
bash (package)

bash (package)
```

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash
```

```bash
```

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Checks whether new_validator is duplicate with any currently pending validators.

```bash
```

```bash
```

```bash
```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash
```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash

```

```
bash (package)

bash (package)
```

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash
```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Get the total number of validators in the next epoch.

bash (package)

bash (package)

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Returns true iff the address exists in active validators.

bash (package)

bash (package)

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash
```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Checks whether new_validator is duplicate with any currently active validators. It differs from is_active_validator_by_sui_address in that the former checks only the sui address but this function looks at more metadata.

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

```
bash (package)
```

```
bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash
```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

```
bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash
```

```
bash (package)

bash (package)
```

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash
```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

```bash

```

```bash

```

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash
```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash
```

```bash
```

`bash (package)`

`bash (package)`

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash
```

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash
```

```bash
```

`bash (package)`

`bash (package)`

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

Function

Checks whether new_validator is duplicate with any currently pending validators.

```bash

```

```bash

```

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash
```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash

```

Returns true if addr is an active validator

~~~
bash (package)

bash (package)
~~~

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

~~~
bash (package)

bash (package)

bash (package)

bash (package)
~~~

```bash

```

```bash

```

Function

```bash

```

```bash

```

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```bash
```

```bash
```

Function

Get mutable reference to either a candidate or an active validator by address.

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)

bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

```
bash (package)

bash (package)

bash (package)
```

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```bash

```

```bash

```

Function

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash
```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Find validator by validator_address, in validators. Returns (true, index) if the validator is found, and the index is its index in the list. If not found, returns (false, 0).

```bash

```

```bash

```

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash

```

`bash (package)`

`bash (package)`

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

`bash (package)`

`bash (package)`

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```bash

```

```bash

```

Function

Given a vector of validator addresses, return their indices in the validator set. Aborts if any address isn't in the given validator set.

```bash

```

```bash

```

```
bash (package)
```

```
bash (package)
```

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash
```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

    bash (package)

    bash (package)

    bash (package)

    bash (package)

```bash

```

```bash

```

Function

Get mutable reference to an active or (if active does not exist) pending or (if pending and active do not exist) or candidate validator by address. Note: this function should be called carefully, only after verifying the transaction sender has the ability to modify the Validator.

```bash

```

```bash

```

    bash (package)

    bash (package)

    bash (package)

    bash (package)

    bash (package)

    bash (package)

```bash

```

```bash

```

    bash (package)

    bash (package)

```bash

```

```bash

```

```bash
```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

bash (package)

bash (package)

```bash
```

```bash
```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```bash

```

```bash

```

Function

bash (package)

bash (package)

```bash

```

```bash

```

bash (package)

bash (package)

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash
```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash

```

Function

```bash

```

```bash

```

```
bash (package)

bash (package)
```

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)

bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

bash (package)

bash (package)

```bash

```

```bash

```

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

```
bash (package)
```

```
bash (package)
```

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

Function

```bash

```

```bash

```

```bash

```

```bash

```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

Function

```bash
```

```bash
```

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash
```

```bash
```

Remove validator from self and return the amount of stake that was removed

```bash
```

```bash
```

```bash
```

```bash
```

Sort all the pending removal indexes.

```bash
```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Verify the capability is valid for a Validator. If active_validator_only is true, only verify the Cap for an active validator. Otherwise, verify the Cap for au either active or pending validator.

bash (package)

bash (package)

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Process the pending withdraw requests. For each pending request, the validator is removed from validators and its staking pool is put into the inactive_validators table.

```bash

```

```bash

```

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash
```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)
```

```
bash (package)
```

bash (package)

bash (package)

```bash

```

```bash

```

Function

Remove validator from self and return the amount of stake that was removed

```bash

```

```bash

```

```bash

```

```bash

```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

```bash

```

```bash
```

Sort all the pending removal indexes.

```bash

```

```bash

```

Process all active validators' pending stake deposits and withdraws.

```bash

```

```bash

```

Calculate the total active validator stake.

```
bash (package)
```

```
bash (package)
```

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Function

Sort all the pending removal indexes.

```bash
```

```bash
```

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash
```

```bash
```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)

bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash
```

```bash
```

Function

Process all active validators' pending stake deposits and withdraws.

```bash
```

```bash
```

Calculate the total active validator stake.

bash (package)

bash (package)

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Calculate the total active validator stake.

`bash (package)`

`bash (package)`

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Process the pending stake changes for each validator.

```bash

```

```bash

```

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash

```

```bash

```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Function

Compute both the individual reward adjustments and total reward adjustment for staking rewards as well as storage fund rewards.

```bash
```

```bash
```

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash
```

```bash
```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)

bash (package)

bash (package)

bash (package)
```

```bash

```

```bash

```

Function

Process the validator report records of the epoch and return the addresses of the non-performant validators according to the input threshold.

```bash

```

```bash

```

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash

```

```bash

```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Function

Given the current list of active validators, the total stake and total reward, calculate the amount of reward each validator should get, without taking into account the tallying rule results. Returns the unadjusted amounts of staking reward and storage fund reward for each validator.

```bash
```

```bash
```

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash
```

```bash
```

```bash
```

```bash
```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash

```

Function

Use the reward adjustment info to compute the adjusted rewards each validator should get. Returns the staking rewards each validator gets and the storage fund rewards each validator gets. The staking rewards are shared with the stakers while the storage fund ones are not.

```bash

```

```bash

```

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

```bash

```

```bash

```

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash

```

```bash

```

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash
```

```bash
```

Function

Emit events containing information of each validator for the epoch, including stakes, rewards, performance, etc.

```bash
```

```bash
```

Sum up the total stake of a given list of validator addresses.

```bash
```

```bash
```

Return the active validators in self

```bash
```

```bash
```

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

`bash (package)`

`bash (package)`

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Sum up the total stake of a given list of validator addresses.

```bash

```

```bash

```

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash

```

```bash

```

Function

Return the active validators in self

```bash

```

```bash

```

Returns true if the addr is a validator candidate.

```bash

```

```bash

```

Returns true if addr is an active validator

bash (package)

bash (package)

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

bash (package)

bash (package)

bash (package)

bash (package)

```bash
```

```bash
```

Function

Returns true if the addr is a validator candidate.

```bash
```

```bash
```

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash
```

Return true if addr is currently an at-risk validator below the minimum stake for removal

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```
bash (package)
```

```bash
```

```bash
```

Function

Returns true if addr is an active validator

```
bash (package)
```

```
bash (package)
```

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash
```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash

```

Function

Returns true if the staking pool identified by staking_pool_id is of an inactive validator.

```bash

```

```bash

```

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash

```

Function

Return true if addr is currently an at-risk validator below the minimum stake for removal

`bash (package)`

`bash (package)`

`bash (package)`

`bash (package)`

```bash

```

```bash
```

Function

bash (package)

bash (package)

```bash

```

```bash

```

Macro function

```bash

```

```bash

```