tensor.greater_equal

tensor.greater_equal

```

Copy fngreater_equal(self:@Tensor, other:@Tensor)->Tensor;

```

Check if each element of the first tensor is greater than or equal to the corresponding element of the second tensor. The input tensors must have either:

- Exactly the same shape
- The same number of dimensions and the length of each dimension is either a common length or 1.
-

Args

- self
- (@Tensor
-) - The first tensor to be compared
- other
- (@Tensor
-) - The second tensor to be compared
-

Panics

- Panics if the shapes are not equal or broadcastable
-

Returns

A newTensor of booleans (0 or 1) with the same shape as the broadcasted inputs.

Examples

Case 1: Compare tensors with same shape

```

Copy usecore::array::{ArrayTrait,SpanTrait};

useorion::operators::tensor::{TensorTrait,Tensor,U32Tensor};

fngreater_equal_example()->Tensor { lettensor_1=TensorTrait::::new( shape:array![3,3,3].span(), data:array!
[0,1,2,3,4,5,6,7,8].span(), );

lettensor_2=TensorTrait::::new( shape:array![3,3,3].span(), data:array![0,1,2,3,4,5,9,1,5].span(), );

// We can call greater_equal function as follows. returntensor_1.greater_equal(@tensor_2); }

                    [1,1,1,1,1,1,0,1,1]

```

Case 2: Compare tensors with different shapes

```

Copy usecore::array::{ArrayTrait,SpanTrait};

useorion::operators::tensor::{TensorTrait,Tensor,U32Tensor};

fngreater_equal_example()->Tensor { lettensor_1=TensorTrait::::new( shape:array![3,3,3].span(), data:array!
[0,1,2,3,4,5,6,7,8].span(), );

lettensor_2=TensorTrait::::new(shape:array![3].span(), data:array![0,1,2].span(),);
```

// We can call greater_equal function as follows. returntensor_1.greater_equal(@tensor_2); }

[1,1,1,1,1,1,0,0,0]

```

Previous tensor.greater Next tensor.less

Last updated3 months ago
```