The existing iterator of your **ADS_set** needs to be expanded to include an additional “mode”. In the previous implementation, the iterator returns all *n* elements of the **ADS_set** in any order, whereby the order must always be the same as long as the **ADS_set** is not changed (“normal” mode). In the special mode, the iterator returns the elements in the same order, but after the first element only those elements are returned that are larger[1] than the element returned immediately before. All other elements are skipped. In both modes, the iterator reaches **end() after the last element returned.**

Mathematically speaking, the iterator in mode specifically returns the longest ascending sorted subsequence that begins with the first element.

Details: Extend your implementation to include the **ADS_set** method

const_iterator x() const;

x() creates an iterator in special mode. If there is no element in the **ADS_set ,** then **x()== end().**

The time complexity and memory complexity of the operator functions must remain unchanged.
So are e.g. B. additional fields with a non-constant size are not permitted.

Examples:

Suppose the iterator returned by **begin()** returns all n stored elements in the order (1,2,4,3,5,6,7) (1,2,3,4,5,6) (1,4,2 ,6,5) (4,3,6,1,5)	Then the iterator returned by x() returns the following elements in the following order
(9,7,8) (7,6) (7) ()	(1,2,4,5,6,7)
	(1,2,3,4,5,6)
	(1,4,6)
	(4,6)
	(9)
	(7)
	(7)
	()

Instructions: Do **not** write a new iterator class! Extend the existing iterator class as follows (this is just one of the possible solutions, different correct solutions are of course permitted):

- It must be possible to create an iterator in “special” mode. To do this, a new constructor needs to be written and/or an existing one needs to be extended. You may need additional constructor parameters and/or instance variables.

- Adjust the increment operations (only!) for the “special” mode: All elements that are not larger than the previously delivered element, will be skipped. After the last element to be returned, the iterator is set to **end()** (as always) .

As before , the **ADS_set::begin()** method returns an iterator in “normal” mode.

[1] The call **std::less<key_type>{}(key1,key2)** for the two values **key1** and **key2** returns **true** if **key1** is smaller than **key2** and **false** otherwise (alternatively, the alias **key_compare** can be used if it is defined accordingly) .