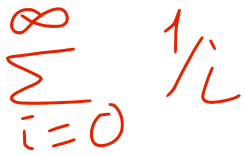

Der bestehende Iterator Ihres **ADS_set** ist um einen zusätzlichen „Modus“ zu erweitern. In der bisherigen Implementierung liefert der Iterator alle Elemente in einer beliebigen Reihenfolge, wobei die Reihenfolge immer dieselbe sein muss, solange das **ADS_set** nicht geändert wird (Modus „normal“). Im neuen Modus „speziell“ sollen die Werte in derselben Reihenfolge geliefert werden, aber es werden nur Werte geliefert, die größer als ein Vergleichswert (**limit**) sind. Die anderen Werte werden sozusagen übersprungen. In beiden Modi erreicht der Iterator **end()**, sobald keine (passenden) Elemente mehr vorhanden sind.

Details: Erweitern Sie Ihre Implementierung **ADS_set** um die Methode

```
const_iterator w(const key_type &limit) const;
```

Diese soll einen Iterator im Modus „speziell“ liefern. Dieser zeigt auf das erste Element, das größer als **limit** ist. Wenn kein entsprechendes Element im **ADS_set** vorhanden ist, dann gilt **w()**== **end()**. Der von **w()** gelieferte Iterator soll beim Inkrementieren in weiterer Folge nur Elemente liefern, die größer als **limit** sind. Befinden sich keine weiteren solchen Elemente im **ADS_set**, dann entspricht der Iterator (wie ein normaler Iterator nach dem letzten Element) dem Enditerator.

Für den Vergleich von Werten ist **std::less** (bzw. der alias **key_compare**, sofern vorhanden) zu verwenden. Der Aufruf **std::less<T>{}(x,y)** für die beiden Werte **x** und **y** vom Typ **T** liefert **true**, falls **y** größer als **x** ist, und **false** sonst.

Die Zeitkomplexität und Speicherkomplexität der Operatorfunktionen müssen unverändert bleiben. So sind zB zusätzliche Felder mit nicht konstanter Größe unzulässig.

Beispiel: Angenommen der Iterator Ihres **ADS_set** liefert alle gespeicherten Elemente in der Reihenfolge (4,7,1,3,6,8,10,2), dann gilt

- **w(8)** liefert einen Iterator, der auf **10** zeigt.
  Wenn der Iterator bis **end()** inkrementiert wird, liefert er die Folge **(10)**
- **w(5)** liefert einen Iterator, der auf **7** zeigt.
  Wenn der Iterator bis **end()** inkrementiert wird, liefert er die Folge **(7,6,8,10)**
- **w(0)** liefert einen Iterator, der auf **4** zeigt.
  Wenn der Iterator bis **end()** inkrementiert wird, liefert er die Folge **(4,7,1,3,6,8,10,2)**
- **w(11)** liefert **end()**

Anleitung: Schreiben Sie **keine** neue Iteratorklasse! Erweitern Sie die bestehende Iterator-Klasse um

- zusätzliche Instanzvariablen, um den Grenzwert (**limit**) zu speichern, und um zu speichern, ob der Iterator im Modus „normal“ oder „speziell“ ist.
- einen zusätzlichen Konstruktor für den Modus „speziell“, mit dem diese Instanzvariablen und die Startposition entsprechend gesetzt werden. Eventuell sind vorhandene Konstruktoren ebenfalls anzupassen (diese liefern wie bisher einen Iterator im Modus „normal“).

Passen Sie die Inkrement-Operationen so an, dass nicht passende Werte übersprungen werden, wenn der Iterator im Modus „speziell“ ist.

Die Methode **ADS_set::w()** erzeugt mit Hilfe des neuen Iterator-Konstruktors einen Iterator im Modus „speziell“ und retourniert diesen. Die Methode **ADS_set::begin()** liefert wie bisher einen Iterator im Modus „normal“ (bleibt also unverändert).