Der bestehende Iterator Ihres **`ADS_set`** ist um einen zusätzlichen „Modus“ zu erweitern. In der bisherigen Implementierung liefert der Iterator alle *n* Elemente des **`ADS_set`** in einer beliebigen Reihenfolge, wobei die Reihenfolge immer dieselbe sein muss, solange das **`ADS_set`** nicht geändert wird (Modus „normal“). Im neuen Modus „speziell“ liefert der Iterator dieselben Elemente in derselben Reihenfolge, das zweite Element aus der ursprünglichen Reihenfolge wird jedoch als letztes Element geliefert, bevor der Iterator **`end()`** erreicht.

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

```
const_iterator w() const;
```

`w()` erzeugt einen Iterator im Modus „speziell“. Wenn kein Element im **`ADS_set`** vorhanden ist, dann gilt **`w()`** == **`end()`**.

Die Zeitkomplexität und Speicherkomplexität aller Funktionen müssen unverändert bleiben. So sind z. B. zusätzliche Felder mit nicht konstanter Größe unzulässig. Für die Verwendung der STL und für die Implementierung allgemein gelten dieselben Regeln wie im gesamten Projekt (zB alle Instanzvariablen **`private`**, keine globalen Variablen, etc).

Beispiele:

Angenommen der von **`begin()`** retournierte Iterator liefert alle n gespeicherten Elemente in der Reihenfolge	Dann liefert der von **`w()`** retournierte Iterator die folgenden Elemente in der folgenden Reihenfolge
(1,2,3,4,5,6,7)	(1,3,4,5,6,7,2)
(4,2,3,1,5,6)	(4,3,1,5,6,2)
(4,2,6,5,1)	(4,6,5,1,2)
(5,3,4)	(5,4,3)
(7,8)	(7,8)
(7)	(7)
()	()

Anleitung: Schreiben Sie **keine** neue Iteratorklasse! Erweitern Sie die bestehende Iterator-Klasse wie folgt (dies ist nur einer der möglichen Lösungsansätze, abweichende korrekte Lösungen sind natürlich zulässig):

- Es muss ein Iterator im Modus „speziell“ erzeugt werden können. Dazu ist ein neuer Konstruktor zu schreiben und/oder bestehende zu erweitern. Eventuell benötigen Sie zusätzliche Konstruktorparameter und/oder Instanzvariablen.
- Passen Sie die Inkrement-Operationen (nur!) für den Modus „speziell“ an: Wenn der Iterator auf das zweite Element gesetzt wird, wird dieses übersprungen (der Iterator wird also zum dritten Element weitergeschaltet, sofern vorhanden). Wenn der Iterator vom ursprünglich letzten Element weitergeschaltet wird, wird der Iterator auf das ursprünglich zweite Element gesetzt. Bei der darauffolgenden Inkrement-Operation wird der Iterator auf **`end()`** gesetzt. Dazu kann es hilfreich sein, bestimmte Positionen im Iterator zu speichern (also die Werte der relevanten Instanzvariablen), damit der Iterator zum vorgesehenen Zeitpunkt auf diese Positionen gesetzt werden kann.

Die Methode **`ADS_set::begin()`** liefert wie bisher einen Iterator im Modus „normal“.