We represent a set by the sorted sequence of its elements. By specializing the auxiliary methods he generic merge algorithm can be used to perform basic set operations:
- union
- intersection
- subtraction

The running time of an operation on sets $A$ and $B$ should be at most $O(n_A + n_B)$.

**Set union:**
- $a < b$
  - $a \text{IsLess}(a, S)$
  - $S.\text{insertFirst}(a)$
  - $A.\text{remove}(A.\text{first}())$
- $b < a$
  - $b \text{IsLess}(b, S)$
  - $B.\text{remove}(B.\text{first}())$
- $a = b$
  - $\text{bothAreEqual}(a, b, S)$
  - $S.\text{insertLast}(a)$

**Set intersection:**
- $a < b$
  - $\text{do nothing}$
- $b < a$
  - $\text{do nothing}$
- $a = b$
  - $\text{bothAreEqual}(a, b, S)$
  - $S.\text{insertLast}(a)$

**Set subtraction:**
- $a < b$
  - $\text{do nothing}$
- $b < a$
  - $\text{do nothing}$
- $a = b$
  - $\text{bothAreEqual}(a, b, S)$
  - $S.\text{insertLast}(a)$

---

**Storing a Set in a List**
- We can implement a set with a list.
- Elements are stored sorted according to some canonical ordering.
- The space used is $O(n)$.

**Generic Merging**
- Generalized merge of two sorted lists $A$ and $B$.
- Template method `genericMerge`.
- Auxiliary methods:
  - $a < b$
    - $a \text{IsLess}(a, S)$
    - $S.\text{insertFirst}(a)$
    - $A.\text{remove}(A.\text{first}())$
  - $b < a$
    - $b \text{IsLess}(b, S)$
    - $B.\text{remove}(B.\text{first}())$
  - $a = b$
    - $\text{bothAreEqual}(a, b, S)$
    - $S.\text{insertLast}(a)$

**Algorithm** `genericMerge(A, B)`
```
S - empty sequence
while ¬A.isEmpty()∧¬B.isEmpty() do
    a ← A.first().element
    b ← B.first().element
    if $a < b$
        aIsLess(a, S)
        A.remove(A.first())
    else if $b < a$
        bIsLess(b, S)
        B.remove(B.first())
    else
        bothAreEqual(a, b, S)
        A.remove(A.first())
        B.remove(B.first())
return S
```

**Using Generic Merge for Set Operations**
- Any of the set operations can be implemented using a generic merge.
- For example:
  - For intersection: only copy elements that are duplicated in both list.
  - For union: copy every element from both lists except for the duplicates.
- All methods run in linear time.