Sequences

Define a type class `Sequence a` which consists of the functions `next` and `prev` to query the next and previous element in the sequence.

```
    Main> prev 't'
    's'

    Main> next 'z'
    *** Exception: no value after 'z'

    Main> next (2 :: Int)
    3

    Main> next True
    False

    Main> next False
    True
```

Define the instances of `Sequence a` for `Integer`, `Char`, and `Bool`. The instance for `Char` should only consider the small letters from a to z.
Hint: have a look in the `Data.Char` module.

Make two subclasses of this type class: `LeftBoundedSequence a` and `RightBoundedSequence a`. The former has a function `firstElem` and the latter has a function `lastElem` to query the first and the last elements in the sequence.

```
    Main> firstElem :: Char
    'a'

    Main> lastElem :: Bool
    True
```

Define instances for these two classes for `Char` and `Bool`. Note that `Integer` does not have left or right bounds, so it shouldn't be made an instance of these classes.