1) MyArrayIterator.java

```java
package state.e7;

public class MyArrayIterator implements MyIterator {

    MyArray myArray;
    int current;
    MyIteratorState state;

    StateCurrent stateCurrent = new StateCurrent(this);
    StateCurrentEol stateCurrentEol = new StateCurrentEol(this);
    StateNoCurrent stateNoCurrent = new StateNoCurrent(this);
    StateNoCurrentEol stateNoCurrentEol = new StateNoCurrentEol(this);

    public MyArrayIterator(MyArray myArray) {
        this.myArray = myArray;
        this.current = 0;
        this.state = myArray.getCount() > 0 ? stateNoCurrent : stateNoCurrentEol;
    }

    @Override
    public int getNext() {
        return state.getNext();
    }

    @Override
    public boolean isEnd() {
        return state.isEnd();
    }

    @Override
    public void remove() {
        state.remove();
    }
}
```

2) MyIteratorState.java

```java
package state.e7;

public abstract class MyIteratorState {

    MyArrayIterator wrapper;

    public MyIteratorState(MyArrayIterator myArrayIterator) {
        this.wrapper = myArrayIterator;
    }

    abstract int getNext();
    abstract boolean isEnd();
    abstract void remove();

    int getNextImpl() {
        return wrapper.myArray.data[wrapper.current++];
    }

    void removeImpl(MyIteratorState nextState) {
        wrapper.state = nextState;
        --wrapper.current;
        --wrapper.myArray.count;
        for (int i = wrapper.current; i < wrapper.myArray.count; ++i)
            wrapper.myArray.data[i] = wrapper.myArray.data[i + 1];
    }
}
```

3) StateCurrent.java

```java
package state.e7;

public class StateCurrent extends MyIteratorState {

    public StateCurrent(MyArrayIterator myArrayIterator) {
        super(myArrayIterator);
    }

    @Override
    public int getNext() {
        if (wrapper.current < wrapper.myArray.count - 1)
            wrapper.state = wrapper.stateCurrent;
        else
            wrapper.state = wrapper.stateCurrentEol;
        return getNextImpl();
    }

    @Override
    public boolean isEnd() {
        return false;
    }

    @Override
    public void remove() {
        removeImpl(wrapper.stateNoCurrent);
    }
}
```

4) StateCurrentEol.java

```java
package state.e7;

import java.util.NoSuchElementException;

public class StateCurrentEol extends MyIteratorState {

    public StateCurrentEol(MyArrayIterator myArrayIterator) {
        super(myArrayIterator);
    }

    @Override
    public int getNext() {
        throw new NoSuchElementException();
    }

    @Override
    public boolean isEnd() {
        return true;
    }

    @Override
    public void remove() {
        removeImpl(wrapper.stateNoCurrentEol);
    }
}
```

5) StateNoCurrent.java

```
1  package state.e7;
2
3  import java.util.NoSuchElementException;
4
5  public class StateNoCurrent extends MyIteratorState {
6
7      public StateNoCurrent(MyArrayIterator myArrayIterator) {
8          super(myArrayIterator);
9      }
10
11     @Override
12     public int getNext() {
13         if (wrapper.current < wrapper.myArray.count - 1)
14             wrapper.state = wrapper.stateCurrent;
15         else
16             wrapper.state = wrapper.stateCurrentEol;
17         return getNextImpl();
18     }
19
20     @Override
21     public boolean isEnd() {
22         return false;
23     }
24
25     @Override
26     public void remove() {
27         throw new NoSuchElementException();
28     }
29 }
```

6) StateNoCurrentEol.java

```java
package state.e7;

import java.util.NoSuchElementException;

public class StateNoCurrentEol extends MyIteratorState {

    public StateNoCurrentEol(MyArrayIterator myArrayIterator) {
        super(myArrayIterator);
    }

    @Override
    public int getNext() {
        throw new NoSuchElementException();
    }

    @Override
    public boolean isEnd() {
        return true;
    }

    @Override
    public void remove() {
        throw new NoSuchElementException();
    }
}
```

7) Example7.java

```java
package state.e7;

import java.util.function.Predicate;

public class Example7 {

    static void add(MyCollection col, int count) {
        for (int i = 0; i < count; ++i)
            col.add(i);
    }

    static void print(MyIterator it) {
        while (!it.isEnd())
            System.out.printf("%d ", it.getNext());
        System.out.println();
    }

    // predicate를 만족하는 항목들을 제거한다
    static void remove(MyIterator it, Predicate<Integer> predicate) {
        while (!it.isEnd())
            if (predicate.test(it.getNext()))
                it.remove();
    }

    static void doSomething1(MyCollection col) {
        add(col, 10);
        print(col.getIterator());
        remove(col.getIterator(), value -> value < 5); // 5 미만 제거
        print(col.getIterator());
        remove(col.getIterator(), value -> value > 5); // 5 초과 제거
        print(col.getIterator());
    }

    static void doSomething2(MyCollection col) {
        add(col, 10);
        MyIterator it = col.getIterator();
        it.remove();
        print(col.getIterator());
    }

    public static void main(String[] args) {
        doSomething1(new MyArray());
        doSomething2(new MyArray());
    }
}
```

실행

```
0 1 2 3 4 5 6 7 8 9
5 6 7 8 9
5
Exception in thread "main" java.util.NoSuchElementException
        at state.e7.StateNoCurrent.remove(StateNoCurrent.java:27)
        at state.e7.MyArrayIterator.remove(MyArrayIterator.java:32)
        at state.e7.Example7.doSomething2(Example7.java:37)
        at state.e7.Example7.main(Example7.java:43)
```