package Zadanie2;  
  
public class Main {  
 public static void main(String[] args) {  
 NumberIterator it = new NumberIterator(10, 25);  
 FilterNumberIterator primalIterator = new FilterNumberIterator(13, 47, (number) -> {  
 for(int i = 2; i \* i <= number; i++){  
 if(number % i == 0){  
 return false;  
 }  
 }  
  
 return true;  
 });  
  
 System.*out*.println("Iterator liczbowy");  
 while(!it.isDone()){  
 System.*out*.println(it.currentItem());  
 it.next();  
 }  
  
 System.*out*.println("**\n**Iterator liczb pierwszych");  
 while(!primalIterator.isDone()){  
 System.*out*.println(primalIterator.currentItem());  
 primalIterator.next();  
 }  
 }  
}

package Zadanie2;  
  
public interface Iterator<T>{  
 void first();  
 void next();  
 boolean isDone();  
 T currentItem();  
}

package Zadanie2;  
  
public class NumberIterator implements Iterator<Integer>{  
 private int minNumber;  
 private int currentNumber;  
 private int maxNumber;  
  
 public NumberIterator(int min, int max) {  
 minNumber = min;  
 currentNumber = min;  
 maxNumber = max;  
 }  
  
 @Override  
 public void first() {  
 currentNumber = minNumber;  
 }  
  
 @Override  
 public void next() {  
 currentNumber++;  
 }  
  
 @Override  
 public boolean isDone() {  
 return currentNumber > maxNumber;  
 }  
  
 @Override  
 public Integer currentItem() {  
 return currentNumber;  
 }  
}

package Zadanie2;  
  
import podpunktA.Predicate;  
  
public class FilterNumberIterator implements Iterator<Integer>{  
 private Predicate<Integer> predicate = null;  
 private NumberIterator iterator = null;  
  
 private boolean hasNext = true;  
 private Integer next = null;  
  
 public FilterNumberIterator(int min, int max, Predicate<Integer> pred){  
 predicate = pred;  
 iterator = new NumberIterator(min, max);  
  
 findNextValid();  
 }  
  
 @Override  
 public void first() {  
 iterator.first();  
 }  
  
 @Override  
 public void next() {  
 iterator.next();  
 findNextValid();  
 }  
  
 @Override  
 public boolean isDone() {  
 return !hasNext;  
 }  
  
 @Override  
 public Integer currentItem() {  
 return next;  
 }  
  
 private void findNextValid(){  
 while(!iterator.isDone()){  
 next = iterator.currentItem();  
  
 if(predicate.accept(next)){  
 return;  
 }  
  
 iterator.next();  
 }  
  
 hasNext = false;  
 next = null;  
 }  
}

Wynik powyższego programu:

Iterator liczbowy

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

Iterator liczb pierwszych

13

17

19

23

29

31

37

41

43

47