Classes: AdditionPattern FRQ

Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
Unless otherwise noted in the question, assume that parameters in method calls are not null and that methods are called only when their preconditions are satisfied.
In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods will not receive full credit.

This question involves the implementation of the `AddtionPattern` class, which generates a number pattern. The `AddtionPattern` object is constructed with two positive integer parameters, as described below.

The first positive integer parameter indicates the starting number in the pattern.
The second positive integer parameter indicates the value that is to be added to obtain each subsequent number in the pattern.
The `AddtionPattern` class also supports the following methods.

`currentNumber`, which returns the current number in the pattern
`next`, which moves to the next number in the pattern
`prev`, which moves to the previous number in the pattern or takes no action if there is no previous number

The following table illustrates the behavior of an *AddtionPattern* object that is instantiated by the following statement.

```
AdditionPattern plus3 = new AdditionPattern(2, 3);
```

Method Call	**Value Returned** *(blank if no value)*	**Explanation**
`plus3.currentNumber();`	`2`	The current number is initially the starting number in the pattern.
`plus3.next();`		The pattern adds 3 each time, so move to the next number in the pattern (5).
`plus3.currentNumber();`	`5`	The current number is 5.
`plus3.next();`		The pattern adds 3 each time, so move to the next number in the pattern (8).
`plus3.next();`		The pattern adds 3 each time, so move to the next number in the pattern (11).
`plus3.currentNumber();`	`11`	The current number is 11.
`plus3.prev();`		Move to the previous number in the pattern (8).
`plus3.prev();`		Move to the previous number in the pattern (5).
`plus3.prev();`		Move to the previous number in the pattern (2).
`plus3.currentNumber();`	`2`	The current number is 2.
`plus3.prev();`		There is no previous number in the pattern prior to 2, so no action is taken.
`plus3.currentNumber();`	`2`	The current number is 2.

Write the complete *AdditionPattern* class. Your implementation must meet all specifications and conform to all examples.