This question involves the implementation of the `AddtionPattern` class, which generates a number pattern.
The `AdditionPattern` 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 `AdditionPattern` 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 `AdditionPattern` 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 `AdditonPattern` class. Your implementation must meet all specifications and conform to all examples.

Rubric

AdditionPattern class (9 points)

Points earned:

+1 [Skill 3.B] Correct class header (must not be private)

+1 [Skill 3.B] Declares appropriate instance variables (must be private)

+1 [Skill 3.B] Correct constructor header (must not be private)

+1 [Skill 3.B] Constructor correctly initializes instance variables

Response earns this point, but incurs general penalty z below if it...

· returns a value from the constructor

Note that general penalty z can only be incurred once, even if the error is made multiple times.

+1 [Skill 3.B] Correct method headers for next and prev

+1 [Skill 3.B] Correct method header for currentNumber

+1 [Skill 3.B] Correct implementation of currentNumber

+1 [Skill 3.B] next and prev each update state by the appropriate amount

Response earns this point, but incurs general penalty below if it...

· returns a value from either method

Note that general penalty can only be incurred once, even if the error is made multiple times.

+1 [Skill 3.C] prev prevents moving to a number less than the start value

Response earns this point, but incurs general penalty below if it...

· returns a value from the method

Note that general penalty can only be incurred once, even if the error is made multiple times.

General Penalties:

-1 Extraneous code that causes side-effect (e.g., printing to output, incorrect precondition check)

-1 Local variables used but none declared

-1 Void method or constructor that returns a value