

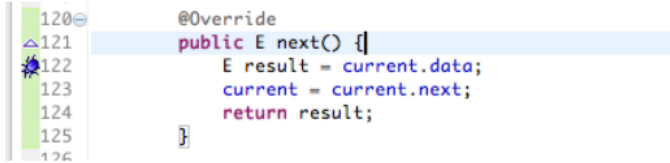
Exercise Labels

Correct = the response correctly explained the problem and provided a solution that removed the notification

Incorrect = the response gives both an incorrect explanation and an incorrect solution

Questionable = the response either a) gives an incorrect explanation but correct solution, b) gives an incorrect or correct explanation but the solution led to a new notification, or c) does not appear to be complete.

Example Response and Labeling

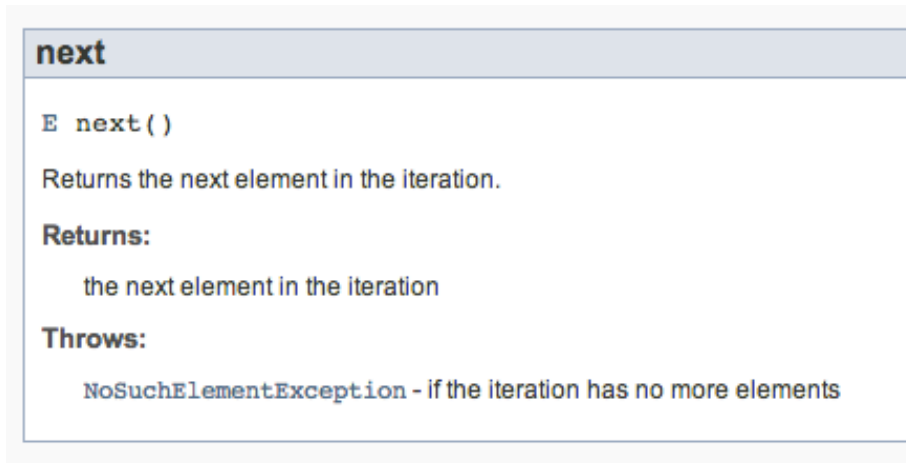
a) 

```
@Override
public E next() {
    E result = current.data;
    current = current.next;
    return result;
}
```

Bug: edu.ncsu.csc216.array_list.ArrayList\$ArrayListIterator.next() can't throw NoSuchElementException

b) This class implements the `java.util.Iterator` interface. However, its `next()` method is not capable of throwing `java.util.NoSuchElementException`. The `next()` method should be changed so it throws `NoSuchElementException` if is called when there are no more elements to return.

Confidence: Low, **Rank:** Of Concern (19)
Pattern: IT_NO_SUCH_ELEMENT
Type: It, **Category:** BAD_PRACTICE (Bad practice)



Let us look at three responses given on the exercise with the bug in the screenshot above to get a better understanding of what it means for a response to be correct, incorrect or questionable. For this notification, students were also directed to look at the `Iterator next()` method API (pictured under notification).

Responses to the following questions determined the correctness of a student's response:

- **Question 1:** What is FindBugs complaining about (what are the notifications trying to tell you about your code)?
- **Question 2:** What can be done to make the bug go away (what modifications need to be made to the code)?

One student responded to Questions 1 and 2 as follows (*Explanation 1*):

- **Question 1:** "next() is supposed to throw a `NoSuchElementException`, but next() as it is implemented now does not. FindBugs is saying that the exception should be thrown if there are no elements to examine."
- **Question 2:** `if (current == null) {throw new NoSuchElementException();}`

Another student responded to the same questions as follows (*Explanation 2*):

- **Question 1:** "There is no check to see if the code has another element, and if it tries to return an element that isn't there, an exception is thrown."
- **Question 2:** "Call `hasNext()` from `next()` so it always checks if there is an element to return."

Again, responding to the same questions for the same notification another student responded as follows (*Explanation 3*):

- **Question 1:** “The next method needs to have a throw NoSuchElementException in its header.”
- **Question 2:** “Type throw NoSuchElementException in the method header.”

Explanation 1 was labeled as **correct**, *Explanation 2* **questionable** and *Explanation 3* **incorrect**.

One of the main differences between these three responses is the level of detail; *Explanations 1 and 2* are much more detailed than *Explanation 3*. The fact that *Explanations 1 and 2* went into more detail suggests that they had a better understanding of the notification than the participant who submitted *Explanation 3*.

Explanation 1 also includes a specific code snippet for resolving the bug. The notification does not say anything about checking for null, however the student understood that checking for null is an important part of fixing this bug (aside from throwing the exception).

Explanation 2 is also detailed, however there are a couple problems with this response.

1. The explanation given has little, if anything, to do with the notification we asked them to explain.
2. The solution given was not enough for us to label it as correct.

These two things combined led to *Explanation 2* being labeled as “questionable”.

Explanation 3 was labeled “incorrect” because the explanation given does not properly explain the problem nor does the solution fix the problem.