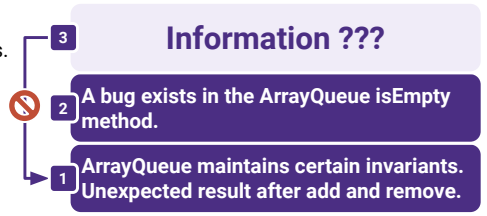
Software Experiments

How are bugs fixed? Here's one proposal.

The point here is that information is the most important thing and you need to do whatever's necessary to get information.

- Productive changes fix bugs.
- Information gathered about the system informs productive changes.
- A hypothesis guides information gathering and testing.
- Things we know about the problem inform how we choose hypotheses.



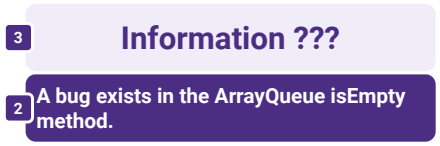
What does debugging a program look like? (Julia Evans); The Debugging Mindset (Devon H. O'Dell/ACM Queue)

Generating Hypotheses

A good hypothesis identifies the cause of failure separate from where and when the program actually fails. The **state** of the ArrayQueue determines the behavior of isEmpty.

State

```
Item[] data
int size
int front
int back
```



The Debugging Mindset (Devon H. O'Dell/ACM Queue)

Generating Hypotheses

A good hypothesis identifies the cause of failure separate from where and when the program actually fails. The **state** of the ArrayQueue determines the behavior of isEmpty.

The hypothesis on the left suggests more about the problem than the one on the right.

The size variable is not set correctly, causing isEmpty to return false.

A bug exists in the ArrayQueue isEmpty method.

?: What is it about the hypothesis on the left that suggests more about the problem?

```
ArrayQueue1<String> queue = new ArrayQueue1<>();
queue.add("a");
queue.remove();
queue.add("big");
queue.remove();
queue.remove();
queue.add("car");
queue.remove();
System.out.println(
    "isEmpty() expected true, got " + queue.isEmpty());
```

Tests as a Source of Information

A good hypothesis describes a problem and is both testable and falsifiable.

Q1: Propose a new hypothesis from the test code.

Gathering information

Debugging is about integrating various different sources of information to identify the source of an error.

- Trying new inputs.
- Writing a unit test to reproduce the bug.
- Explaining to yourself the behavior of each line of code.
- Searching online to understand what error messages mean.
- Changing or removing code.
- Poking at memory values with a debugger or print statements.

?: Bugs often appear away from their root causes. How does each information gathering method help us learn more about the problem?

Software Experiments

How are bugs fixed? Here's one proposal.

- Productive changes fix bugs.
- Information gathered about the system informs productive changes.
- A hypothesis guides information gathering and testing.
- Things we know about the problem inform how we choose hypotheses.

The point here is that information is the most important thing and you need to do whatever's necessary to get information.

3 **Modify the remove method to handle the special case of removing if empty.**

2 **The remove method decrements the size variable even when the queue is empty.**

1 **ArrayQueue maintains certain invariants. Unexpected result after add and remove.**

What does debugging a program look like? (Julia Evans); The Debugging Mindset (Devon H. O'Dell/ACM Queue)

?: What are the differences between this new hypothesis and the hypothesis that we started with? How did we get from the starting hypothesis to this new hypothesis?

A Storytelling with Representation Invariants

Give an invariant for **ArrayQueue2.front**.

How did the program break the invariant?

Experimentation for **correctness** and **efficiency**

While testing software is not a major focus of this course, it's an important tool for analysis of algorithms.

There are three issues with using random testing.

1. **Reproducibility**. Seed the random number generator to get the same sequence of "random" numbers each time.
2. **Reference model**. Random tests often need a reference implementation to compare for correctness. It's not always easy to come up with a reference.
3. **Debuggability**. Testing infrastructure is needed to aid in interpreting results.

Once we have a correct implementation, we can use the Stopwatch class to compute the time between runs.