Generics and Dictionaries

Instructors: Sam McCauley and Dan Barowy
February 21, 2022
Admin

- Remember to do the reading! (We may ask about it on quizzes, especially on Monday)
Admin: Masking

• Instructors now allowed to unmask in classrooms???

• We’ll send around a google form after all three lecture sections

• Idea: gauge what you all think. We really have no idea where all of you are coming from, so hard to make a decision until then
  • (Not a poll; just trying to get high-level idea)

• We don’t really mind masking. Want to ensure best possible experience for you.

• Only lectures! We’ll continue all masking in labs, office hours, etc.
• Any questions?

• If you’re still working on it, be sure to make time to attend TA hours over the next couple days.

• Almost definitely one of the harder labs in the course! But you’re almost there.
Today

- How can we create a `Vector` class?
- Look at how the actual `Vector` class is created
- Start time and space analysis, as well as asymptotics
Building the Vector class
Creating a Vector class

- You have everything you need to create your own Vectors!
- Let’s work through it together. (Then we’ll look at the structure5 code and see how we did.)
- Goal: hold sequence of items. Should be able to handle add(E), get(int), set(int, E), contains(E)
- Use generics to handle any type of item
Designing the class

• What questions should you ask yourself when you start thinking about how to design a class?
  
  • What data does this class need to store? How should we store it?
  
  • What methods do we want to use to interact with this data?
A caveat about Generics: arrays don’t work

- Cannot create an array of a generic type in Java
  - Due to some back-end issues with how generics are implemented

- What can we do instead?

- Create an array of type `Object`. Handle casting manually.

- Good news: we do the casting in the `Vector` class. This issue is invisible to the end user!

```java
private E[] items; //not allowed! Will give an error
private Object[] items; //allowed! We have to do casting, etc. manually
```
Thinking about Storing the Data

• What does an array not support that a Vector does?

• Need to declare the size of an array up front. But don’t need to for Vector!

• How can our Vector class deal with this?
  
  • Let’s start with an array of size 10. What do we do when it fills up?

  • Answer: allocate a bigger array and copy it over!

  • Let’s create a method that does this for us: checks if there’s enough room in the array, and grows it if not. We can call this ensureCapacity(int minCapacity). We’ll come back to this; let’s fill in some methods in the meantime.
Drafting the Vector methods

- Let’s write `add`, `set`, `get`

- Don’t forget to cast when appropriate!

- Once we have these we can test.

- Finally, let’s write `contains`
Ensuring the Capacity

- If the array is too small, how large should we make it?

- One option: make it `minCapacity` size

- Another option: double its size (until large enough)
  - Downside: wastes space
  - Upside: much longer until we have to resize it again
  - We’ll see on Friday that doubling leads to much better performance

- Let’s write out `ensureCapacity`
public vs private methods

- The methods we use to interact with the data stored in an object have to be public (so that they can be called)

- But methods that are only used internally should be private

- Which would you say `ensureCapacity` is?

- Somewhat debatable, but probably private. Only make the array larger when necessary to carry out operations like `add`.

- All done! Let’s test. Then, let’s quickly check what our work looks like compared to the `structure5` implementation
Time and Space Analysis
How efficient is a given method?

• We saw how to do contains in a Vector. How many items did we have to look through in the worst case?

• Let’s say I’m looking through a literal dictionary. Is my contains method very efficient? Do you have a faster way?

• What if I say I’m a really fast reader. Is your method still faster?
  • Probably
  • Unless the dictionary is really short. A fast reader may be able to read through a dictionary with 10 elements better than a more clever search method

• Idea here: analyze the efficiency of a methodology. Your speed—or your computer’s speed—shouldn’t be a factor.
What do we mean by efficiency?

- Perhaps: how long does a method take to run in seconds?

- How much space does it take? (How many bits do we need to store on our computer during the calculation)?
We are looking for worst-case guarantees.

When you write a piece of code, the goal here is to say “I promise that my code will always run efficiently.”

It’s a much more widely applicable statement than “I tested my code out and it seems to run efficiently.”

What if your tests didn’t take into account a key scenario?
The Challenge of Analyzing Time

- Different computers run at different speeds

- Computers are complicated! Adding two numbers together (for example) can take drastically different times depending on context.

- Good news: often times these details don’t change much

- **Example:** It doesn’t matter (too much) how fast I read if I’m scanning thousands of extra dictionary pages.