

LEC 05

CSE 122

Sets, For-Each Loops, Iterators

Questions during Class?

Raise hand or send here

sli.do



BEFORE WE START

*Slido vote & chat with neighbors:
What's the last show you watched?*

Instructor: Ivory & Colin

TAs: Wesley
Rohan
Connor
Dani
Yang

Lecture Outline (1/6)

- **Announcements** 
- Practice Problem
- Sets Review
- Tradeoffs with Different Data Structures
- For-Each Loop
- Iterators
- Exceptions

Announcements

- Creative Assignment 1 (C1) due July 15th!
- Resubmission Cycle 1 form posting tomorrow, July 16th
 - Due July 21st by 11:59pm PT - Eligible assignments: C0, P0

Lecture Outline (2/6)

- Announcements
- Practice Problem 
- Sets Review
- Tradeoffs with Different Data Structures
- For-Each Loop
- Iterators
- Exceptions

Practice Problem:

Write a program that, given a Scanner over a large text file (e.g., *Moby Dick* or the King James Bible), counts the number of unique words in the text.

Lecture Outline (3/6)

- Announcements
- Practice Problem
- **Sets Review** 
- Tradeoffs with Different Data Structures
- For-Each Loop
- Iterators
- Exceptions

(PCM) Sets (ADT)

- A collection of unique values (no duplicates allowed!) that can perform the following operations efficiently:
 - add
 - remove
 - search (contains)
- We don't think of a set as having indices; we just add things to the set in general and don't worry about order



(PCM) Sets in Java

- Set is an interface in Java
 - In `java.util` package
- HashSet and TreeSet are classes that implement the Set interface in Java
 - HashSet: Very fast! Implemented using a “hash table” array
 - *Elements are stored in an unpredictable order*
 - TreeSet: Pretty fast! Implemented using a “binary search tree”
 - *Elements are stored in sorted order*

Set Methods

Method	Description
<code>add(value)</code>	Adds the given value to the set, returns whether or not the given value was added successfully
<code>contains(value)</code>	Returns <code>true</code> if the given value is found in this set
<code>remove(value)</code>	Removes the given value from the set; returns <code>true</code> if the set contained the value, <code>false</code> if not
<code>clear()</code>	Removes all elements from the set
<code>size()</code>	Returns the number of elements in list
<code>isEmpty()</code>	Returns <code>true</code> if the set's size is 0; <code>false</code> otherwise
<code>toString()</code>	Returns a <code>String</code> representation of the set such as <code>"[3, 42, -7, 15]"</code>

Lecture Outline (4/6)

- Announcements
- Practice Problem
- Sets Review
- **Tradeoffs with Different Data Structures** 
- For-Each Loop
- Iterators
- Exceptions

Choosing a Data Structure: Tradeoffs

- Things to consider:
 - Functionality
 - If you need duplicates or indexing, Sets are not for you!
 - Efficiency
 - Different data structures are “good at” different things!

Lecture Outline (5/6)

- Announcements
- Practice Problem
- Sets Review
- Tradeoffs with Different Data Structures
- **For-Each Loop** ◀
- Iterators
- Exceptions

For-Each Loop

- A new kind of loop!

```
Set<String> words = new HashSet<>();  
for (String s : words) {  
    System.out.println(s);  
}
```

- BUT, you cannot modify the data structure inside a for-each loop
 - You will get a **ConcurrentModificationException**
 - They are “read-only”



Practice : Think



sli.do

#cse122

What output is produced by this code?

```
Set<Integer> nums = new TreeSet<>();  
nums.add(3);  
nums.add(9);  
nums.add(3);  
nums.add(-2);  
nums.add(0);
```

```
for (int n : nums) {  
    System.out.print(n + " ");  
}
```

A. -2 0 3 9

B. 3 9 3 -2 0

C. 9 3 0 -2

D. -2 0 3 3 9

E. ConcurrentModificationException



Practice : Pair

[sli.do](#) [#cse122](#)

What output is produced by this code?

```
Set<Integer> nums = new TreeSet<>();  
nums.add(3);  
nums.add(9);  
nums.add(3);  
nums.add(-2);  
nums.add(0);  
  
for (int n : nums) {  
    System.out.print(n + " ");  
}
```

A. -2 0 3 9

B. 3 9 3 -2 0

C. 9 3 0 -2

D. -2 0 3 3 9

E. ConcurrentModificationException

Lecture Outline (6/7)

- Announcements
- Practice Problem
- Sets Review
- Tradeoffs with Different Data Structures
- For-Each Loop
- **Iterators** 
- Exceptions

Iterators

A new object that has access to all of the elements of a given structure and can give them to you, one at a time.

Iterators

- Returned by the `iterator()` method

Methods	Description
<code>hasNext()</code>	Returns true if there are more elements for the iterator to return
<code>next()</code>	Returns the next element in the iteration
<code>remove()</code>	Removes and returns the element that was last returned by <code>next()</code>

- You must use the iterator's `remove()` method to remove things from what you're iterating over – otherwise you will get a **ConcurrentModificationException**

Lecture Outline (7/7)

- Announcements
- Practice Problem
- Sets Review
- Tradeoffs with Different Data Structures
- For-Each Loop
- Iterators
- **Exceptions** ◀

Exceptions

- Sometimes we want to limit someone's input into our method to "valid" options we define
 - Previously printed out "hey don't do that" messages which isn't great...
- Allow us to "fail fast" and immediately halt execution
- No longer need to wrap code in conditionals
- Can include custom error messages about what went wrong

```
if (/* invalid input */) {  
    throw new IllegalArgumentException("Error Message");  
}
```