LEC 07

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 movie you watched?

Music: 122 26sp Lecture Jams 🎧

Instructor: Elba Garza

TAs:

David	Caleb	Cole	Yang
William	Neha	Blake R.	Cady
Dani	Wesley	Carson	Diya
Rohan	Isis	Sushma	
Andrew	Colin	Connor	
Ava	Naomi	Mahima	
Shreyank	Hanna	Nicolae	
Nicole	Blake P.	Ivory	

Lecture Outline (1/6)

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

Announcements

- Programming Assignment 1 (P1) due April 30th!
  - Stacks, Queues, Exceptions
- Resubmission Cycle 1 due April 28th
  - Remember that grades from a resubmission **completely replace** your previous grades for that assignment
  - Resubmission Cycle 2 will open Thursday, April 30th
- Heads up: Quiz 1 scheduled for Thursday, May 7th
  - `ArrayLists`, Reference Semantics, Stacks and Queues, `Sets`, `Maps`
- How to Use the IPL
- Programming Assignment 2 released on Friday, May 1st
  - Yes, two Programming Assignments in a row!
  - BUT, you have one and a half week to complete this assignment

Lecture Outline (2/6)

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

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

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

Lecture Outline (4/6)

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

Choosing a Data Structure: Tradeoffs

- You got a bit of practice with this in your quiz sections yesterday!
  - Solving the same problem with an `ArrayList`, a `Stack`, and a Queue

- 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

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 (6/6)

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

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

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