LEC 07

CSE 122

Sets, For-Each Loops, Iterators

Questions during Class?

Raise hand or send here

sli.do #cse122

BEFORE WE START

Talk to your neighbors:
What did you eat for dinner yesterday?

Instructor Melissa Lin

TAs
Poojitha Arangam
Darel Gunawan
Colton Harris
Atharva Kashyap
Eesha Kunisetty
Audrey Lin
Di Mao
Steven Nguyen
Ben Wang
Jaylyn Zhang

Lecture Outline

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

Announcements

- Creative Project 0 feedback released Wednesday
- Programming Assignment 1 was due yesterday
- Creative Project 1 released today
  - Focused on 2D arrays and Images!
- Resub 2 Form will be posted today
- Quiz 1 next *next* Monday (July 24th)

Lecture Outline

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

Count Unique Words

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

Lecture Outline

- 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)

"hi" "hola"

"bonjour" "hello"

"konichiwa"

- 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`

- `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
`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 set
`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]"`

Count Unique Words (version 2)

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

Lecture Outline

- 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 on Tuesday
  - 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

- 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 #cse-122

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) 3 9 3 -2 0

B) -2 0 3 3 9

C) 9 3 0 -2

D) -2 0 3 9

E) `ConcurrentModificationException`

Practice : Pair

sli.do #cse-122

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) 3 9 3 -2 0

B) -2 0 3 3 9

C) 9 3 0 -2

D) -2 0 3 9

E) ConcurrentModificationException

Lecture Outline

- 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

```
Iterator<String> itr = words.iterator();
```

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`

Filter Words

- Write a method that accepts a Set of Strings and a char c, and removes all the words in the Set that begin with the given character, and returns the number of words removed.

Practice : Think

sli.do #cse-122

What is the best method comment for filterWords()?

```
public static int filterWords(Set<String> lyrics, char c) {
```

A. Accepts a Set of lyrics and a character c and removes the lyrics that start with c using an Iterator. Returns countReturned, the number of words removed from the Set.

B. Removes the Strings in lyrics that start with c, returning the number of elements removed.

C. Removes the Strings in lyrics that start with c.

D. Loops through each word in lyrics, and if the word starts with c, updates a count variable and removes that word from the Set. Returns the count.

Practice : Pair

sli.do #cse-122

What is the best method comment for filterWords()?

public static int filterWords(Set<String> lyrics, char c) {

A. Accepts a Set of song lyrics and a character c and removes the lyrics that start with c using an Iterator. Returns countReturned, the number of words removed from the Set.

B. Removes the Strings in lyrics that start with c, returning the number of elements removed.

C. Removes the Strings in lyrics that start with c.

D. Loops through each word in lyrics, and if the word starts with c, updates a count variable and removes that word from the Set. Returns the count.