CSE12 - Lecture 23

Wednesday, November 29, 2023 8:00 AM

PA8 and PA6 Late/Resubmit - slip day - due Thursday @ 8am
Note: PA8 Late/Resubmit due Friday of Week 10 @ 8am
Exam 3 - next Wednesday - Trees, BST, Heaps, Iterators, Improving Lists

- No design patterns

Lecture 23

Create a RandomStream class that generates random numbers in an enhanced for loop:.

Generating a random number:

From java.util.*

class Random

public int nextInt(int bound)

Returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence.

Random random; = new Random();
int value = random.nextInt(100); //random number between 0 and 99

Note: always use a field, never create a new one over and over again in a loop.

```
RandomStream r = new RandomStream(10, 100);
for (Integer i : r) {
   System.out.println(i);
 }
```

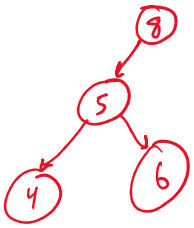
55
75
61
0
1
99 ← bound - 1
7
10
11
30

```
class RandomStream implements Iterable<Integer> {
    int size;
    int bound;
    Random random;
    public RandomStream(int size, int bound) {
        this.size = size;
        this.bound = bound;
        this.random = new Random()
    }
    class RandomIterator implements Iterator<Integer> {
        int current = 0;
        public boolean hasNext() {
            return current < size;
        }
        public Integer next() {
            (1) int value = random.nextInt(bound);
            (2) current += 1;
            (3) return value;
        }
    }
    public Iterator<Integer> iterator() {
        return new RandomIterator();
    }
}
```

could use int seed → repeat sequence

Name: _____ PID: _____ Code: 3655

How would we make a BST iterator?



pre
post
in-order
traversal

Iterator class (add to BST class)
- save state
  - create an ArrayList
  - fill AL with an in-order traversal
- iterator → next / hasNext → same as ArrayList

Run-time:
Constructor - O(n) to copy array list

next() - O(1) to get any element in an array list using an index

How would we make a Heap iterator?

1. copy the heap into the iterator
   - use the poll() of the copy in next
     - all elements in proper heap order

Run-time:
Constructor - O(n) to copy heap (the array)

next() - O(logn) to poll the element at the top of the heap

2. copy the array in the iterator
   - sort in heap order
   - in next() just use the array like an ArrayList version of iterator

Run-time:
Constructor - O(nlogn) copy array is O(n), but sorting is nlogn (quick or merge)

next() - O(1) to get any element in an array list using an index