CSE12 - Lecture 24

Friday, December 1, 2023   8:00 AM

PA7 Late/Resubmit - due Wednesday @ 8am
PA8 Late/Resubmit due Friday of Week 10 @ 8am
Exam 3 - next Wednesday - Trees, BST, Heaps, Iterators, Improving Lists
   - No design patterns

Lecture 24
Complete expandCapacity for the Circular ArrayList

```
public class CAList<E> implements List<E> {
  E[] contents;
  int size;
  int start;

  @SuppressWarnings("unchecked")
  public CAList(int capacity) {
    this.contents = (E[])(new Object[capacity]);
    this.size = 0;
    this.start = 0;
  }

  private int indexFor(int index) {
    int ans = (this.start + index) %
              this.contents.length;
    System.out.println("Index for " + index +
                  " is " + ans);
    return ans;
  }

  public E get(int index) {
    // ASSUME index is in bounds
    int toLookup = this.indexFor(index);
    return this.contents[toLookup];
  }

  public void prepend(E value) {
    expandCapacity();
    this.size += 1;
    this.start -= 1;
    if (this.start == -1) {
      this.start = this.contents.length - 1;
    }
    this.contents[this.start] = value;
  }

  public static void main(String[] args) {
    CAList<Integer> a = new CAList<>(30);
    System.out.println(a);
    a.prepend(30);
    System.out.println(a);
    a.add(40);
    System.out.println(a);
    a.prepend(20);
    System.out.println(a);
    a.add(70);
    System.out.println(a);
  }
}
}
```

$\Theta(1)$

```
@SuppressWarnings("unchecked")
private void expandCapacity() {
  int currentCapacity = this.contents.length;
  if(this.size < currentCapacity) { return; }
```

// save old memory
E[] old Array = this.contents;

// create new array
E[] new Array = (E[]) New Object[2 * this.contents.length];

// iterate through old array
For (int i=0; i < this.size; i++) {
   // New Array [i] = old Array [i];
      New Array [i] = OldArray [this.indexFor(i)];
}

// copy contents
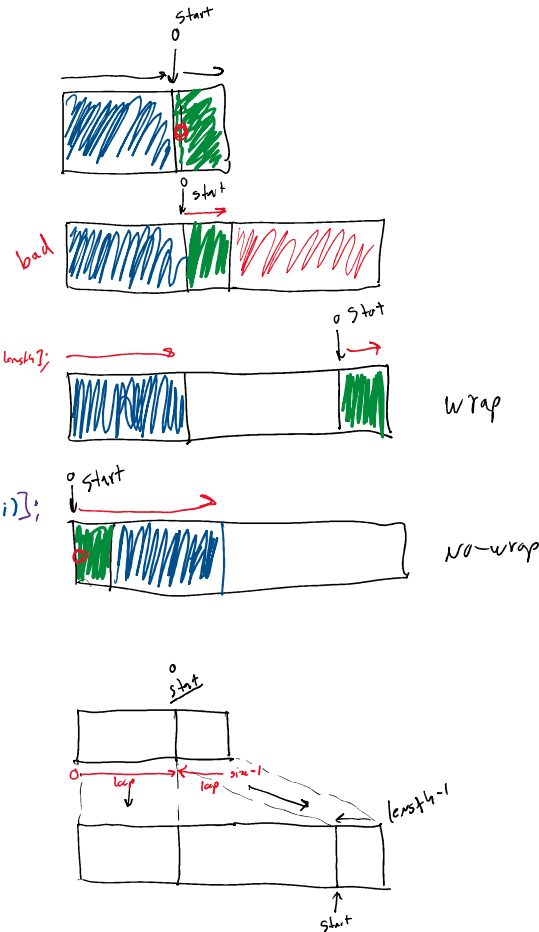this.contents = NewArray;

// reset start
this.start = 0;
}

Name: _____ PID: _____ Code: 9999

start = 8 ↑ 6
size = 8 ↑ 2 ↑ 3 4

```
  0 1 2 3 4 5 6 7
 |40|70| | | | |20|30|
```
6 + 3 = 9 % 8 → 1

Write several tests to confirm that expandCapacity works

CALst lst = CAList(2)

   lst. pre(30)
   lst. add(40)
   lst. pre(20)
   lst. add(70)

NO wrap

|40|30|

|20|40|70|20|

|20|30|40|20| | | | |

wrap

|40|70|20|30|

|40|70| | | | |20|30|

What is the run-time of prepend()?

Worst Case   expand capacity $\Theta(n) + \Theta(1) \rightarrow \Theta(n)$
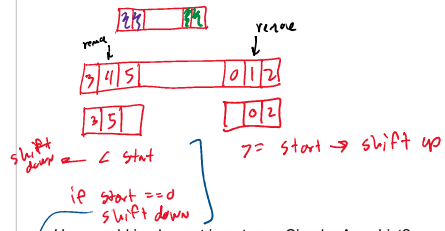
Best Case $\Theta(1)$

Lectures Page 1

Worst Case

expand capacity $\Theta(n) + \Theta(1) \rightarrow \Theta(n)$

Best Case $\Theta(1)$

Average Case $\Theta(1)$ *amortized*
                 per prepend

How would implement remove on a Circular ArrayList?

| 7 | 4 | 5 | | | | 0 | 1 | 2 |

| 7 | 5 | | 0 | 2 |

shift down $\leftarrow$ *< start*

if start == 0
shift down

$7 =$ *start* $\rightarrow$ *shift up*

How would implement insert on a Circular ArrayList?

    0 1 2
| 0 | 1 | 2 | *insert* (1, 5)

    0 1 2 3
| 0 | 5 | 1 | 2 |