CSE12 - Lecture 24 - B00

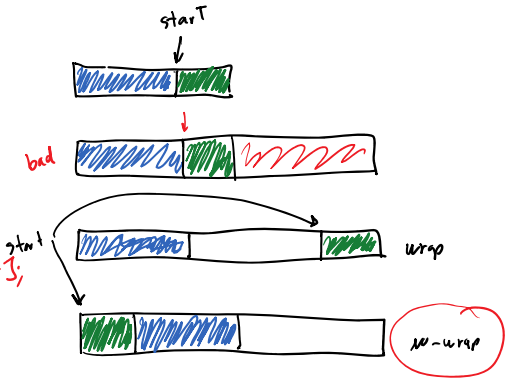
Monday, November 21, 2022 9:00 AM

PA8 hard deadline → tonight
PA6 Late / Resubmit → Tuesday
PA7/ PA8 Late/Resubmit → Friday of Week 10

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);
  }
}
```

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

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

  // create a new array
  E[] newArray =
      new Object[2 * currentCapacity];

  // iterate through old array
  for (int i=0; i< this.size; i++) {
    // newArray[i] = oldArray[i];
    newArray[i] = oldArray[this.indexFor(i)];
  }

  // copy contents into orig fields
  this.contents = newArray;

  // reset start
  start = 0;
}
```



Write several tests to confirm that expandCapacity works

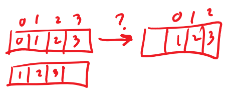
What is the run-time of prepend()?

Worst Case $\Theta(1) + \underset{ec}{\Theta(n)} \to \Theta(N)$

Best Case $\Theta(1)$

Average Case $\Theta(1)$ per add
└ amortized analysis

How would implement remove on a Circular ArrayList?



How would implement insert on a Circular ArrayList?