Data Structures in Java

Notes for 2014-07-10

Monica Quaintance

mjq2102@columbia.edu

Abstract Data Types (ADT)

Lists, Stacks, Queues, Interface (for example, collection)

Collections

–have no ordering

–using "iterable" interface must provide a method named iterator

> check extended for loops, which internally create iterator

List

–ordered

–extends collection (includes all the methods in collection)

inserting into a list:

```
int[] arr = new int[10];
int[] newarr = new int[11];
for int i=0; i < r.length, i++){
  newarr[i] = arr[1];
}
arr = newarr;
```

–inserting is order N

finding an indexed element is order 1

removing an item could be order 1 (from end) or N (from beginning)

Simple Linked List

Items in list have next pointers until the last element points to NULL

Size of linked list–must have locaiton of object, and pointer to next object

Order of making is still O(N), but factor is > 1 for all elements

Retrieving an element from a linked list:

```
ListNode temp = a
while(temp.next==null){
  //do whatever
  temp = temp.next;
}
```

If linked list is NOT a doubly linked list (only has pointers going one way), then adding to end cost is O(n)

Adding at the beginning is O(1)

Removals or additions are O(1) if you are already at the location

Doubly Linked List

Adding to a DLL:

```
temp = head.next; (to tail)
head.next = new ListNode;
head.next.prev = head;
head.next.next = temp;
head.next.next.prev = head.next;
```

Iterator type from java.util:

```
public interface Iterator<AnyType>
{
boolean hasNext( );
AnyType next( );
void remove( );
}
```

Looking at MyArrayList.java from Weiss code selections (on class website)

For a LL remove will be instant, but for an array it will be O(n). But if using iterator you cannot index jump on array, no there is no array benefit.

static– means it hold for entire class

final– means cannot be changed

Inner classes can use the methods of the outer methods that invoke it

++ and – behind a variable get incremented AFTER expression is evaluated

++ and – before a variable get incremented BEFORE expression is evaluated

In order to GET a value, you must go past the value

Therefore, remove takes out the *previous* value

Stack ADT

Push – add element to top of stack

Last thing you put in is the first thing you pull out (Last In, First Out, LIFO)

Pop – take data off the top of the stack

Top/Peek – tells you what the thing is on top, but doesn't remove

Methods: isEmpty, doClear, etc

Array-based implementation:

> Set top-of-stack to new value
>
> Pop just reassigns new value (constant time)
>
> However, doesn't keep track of number of things in stack, so you could go over the amount of space (stack overflow)
>
> Don't have to preallocate space

Any ordered list can be turned into a stack

Uses for stacks:

Reverse Polish Notation, Checking Open/Close Brackets, etc

Miscellaneous

Syllabus covers Chapters in Weiss:

1,2,3,4 (not the end), 5 (not the middle), up to 6.5, up to 7.7, chapter 8, much of 9