PRINCETON UNIVERSITY

COS 226–Algorithms and Data Structures

Week 2: *Analysis of Algorithms, Iterators and Memory (Algorithms §1.4 & videos 3.E & 2.E)*

Version: September 20, 2017

Exercise 1 – Mathematical models for runtime analysis (Algorithms §1.4)

A. Study the code below.

```
int count = 0;
for (int i = 0; i  < m; i++)
    for (int j = 0; j < n; j++)
        for (int k = j+1; k < n; k++)
            if (a[j] >= a[k]) count++;
```

Count the number of array accesses made during code execution using tilde notation.

B. Suppose it takes 100 second to execute this code fragment for $n = 1000$ and $m = 100$. Estimate the run time for $n = 1\,000\,000$ and $m = 1000$.

C. (Optional) Rank the following functions from smallest to largest in order of growth: $n\log_3 n$, $2\log_3 n$, $\sqrt[3]{n}$, n^2, $3\log_2 n$.

D. (Optional) What is the order of growth of the running time of the code fragment below as a function of n?

```
In in = new In(args[0]);
int[] a = in.readAllInts();
int n = a.length;

int uniqueCount = 0;
Arrays.sort(a); //assume that this line takes n log n time.
int i = 0;

while (i < n) {
  int current = a[i];
  uniqueCount += 1;
  int j = i + 1;
  while (j < n) {
    if (a[j] != current)
      break;
    j++;
  }
  i = j;
}
```

Exercise 2 – Iterators (video §3.E)

Consider the partial code of LinkedBag.java below. The container data structure for the LinkedBag is a linked list. We need to develop a ListIterator that can traverse through the list and produce the elements in the list sequentially starting with the head of the list.

```
public class LinkedBag<Item> implements Iterable<Item> {
 private Node first;    // beginning of bag
 private int n;         // number of elements in bag

 // helper linked list class
 private class Node {
    private Item item;
    private Node next;
 }

 /**
 * Initializes an empty bag.
 */
 public LinkedBag() {
    first = null;
    n = 0;
 }

 /**
```

```
20   * Is this bag empty?
21   * @return true if this bag is empty; false otherwise
22   */
23  public boolean isEmpty() {
24     return first == null;
25  }
26
27  public int size() {
28     return n;
29  }
30
31  public void add(Item item) {
32    // code given below
33  }
34
35  public Iterator<Item> iterator()  {
36     return new ListIterator();
37  }
38
39  // an iterator, doesn't implement remove() since it's optional
40  private class ListIterator implements Iterator<Item> {
41     private Node current;
42
43     public ListIterator(){
44
45
46     }
47
48     public boolean hasNext()  {                                          }
49     public void remove()      { throw new UnsupportedOperationException();  }
50
51     public Item next() {
52
53
54
55
56     }
57  }
```

A. Complete the missing code in ListIterator(), hasNext() and next().

B. The code for the add() method in the LinkedBag class is given below.

```
1 public void add(Item item) {
2   Node oldfirst = first;
3   first = new Node();
4   first.item = item;
5   first.next = oldfirst;
6   n++;
7 }
```

When a new element is added to the LinkedBag, does it add the new element to the beginning or to the end of the Linked list?

C. The code below is a client of LinkedBag class. What output is produced by this code?

```
1  LinkedBag<Integer> myBag = new LinkedBag<Integer>();
2  myBag.add(3);
3  myBag.add(1);
4  myBag.add(2);
5  for (int i : myBag){
6       for (int j : myBag) {
7               StdOut.println(i + "␣" + j);
8  }
9   StdOut.println();
10 }
```

Exercise 3 – Memory (video §2.E)

Using the 64-bit memory cost model from Section 1.4 of the textbook, find the answers to following memory questions. Briefly justify your answers and show your work.

A. How much memory (in bytes) does your data types use to store an object of type Node as defined in the LinkedBag class? Do not include the memory for the items themselves (as this memory is allocated by the client and depends on the item type).

B. How much memory (in bytes) do your data types use to store a LinkedBag object of n items in the worst case? Do not include the memory for the items themselves (as this memory is allocated by the client and depends on the item type). Use tilde notation to simplify your answer.

