CS 340 Spring 2017
Homework 3

Due Monday February 27
See Homework Submission Slide
Homework 3

• Implement the SortedLinkedList class shown in the following slides
• The class implements a sorted linked list of ints stored in a binary file
• The file uses the format shown in the following slide
• The number of times an individual int is in the list is recorded in the counts part of a node
<table>
<thead>
<tr>
<th>Addr</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>236</td>
</tr>
<tr>
<td>8</td>
<td>200</td>
</tr>
<tr>
<td>16</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>1 173</td>
</tr>
<tr>
<td>56</td>
<td>3 14 20 22</td>
</tr>
<tr>
<td>92</td>
<td>0</td>
</tr>
<tr>
<td>128</td>
<td>3 80 83 97</td>
</tr>
<tr>
<td>164</td>
<td>1 107</td>
</tr>
<tr>
<td>200</td>
<td>344</td>
</tr>
<tr>
<td>236</td>
<td>2 8 11</td>
</tr>
<tr>
<td>272</td>
<td></td>
</tr>
<tr>
<td>308</td>
<td>3 122 130 134</td>
</tr>
<tr>
<td>344</td>
<td></td>
</tr>
<tr>
<td>380</td>
<td>2 62 63</td>
</tr>
<tr>
<td>416</td>
<td>3 30 42 50</td>
</tr>
</tbody>
</table>
import java.io.*;
import java.util.*;

public class SortedLinkedList {
    //implements a sorted linked list of ints stored in a binary file
    //Each node contains up to nodeSize ints. All nodes in one file have room for nodeSize ints.
    //i.e. all nodes use the same number of bytes in the file

    private RandomAccessFile f;
    private long head; //first 8 bytes in the file
    private long free; //second 8 bytes in the file
    private int nodeSize; //stored in the file at address 16
private class Node {
    private int numInts; // the number of distinct ints in the node
    private int data[];
    private int counts[]; // the number of times each data item is in the list
    private long next;

    private Node(int num, int d[], int c[], long n) {
    }

    private Node(long addr) throws IOException {
    }

    private void writeNode(long addr) throws IOException {
    }
}
public SortedLinkedList(String filename, int ns) throws IOException {
    //creates a new empty file
    //if a file with name filename exists it should be removed before a new empty file is created
    //ns is the maximum number of distinct ints in each node

}

public SortedLinkedList(String filename) throws IOException {
    //reuse an existing file

}
public void insert(int x) {
    //insert a new instance of x into the list
    //if x is not in the list then x is inserted with a counts value of 1
    //if x is in the list then the counts value of x is incremented
    //this method might have to create a new node
    
}

public void remove(int x) {
    //remove one instance of x from the list
    //if the instance removed is the last instance of x then
    //x is removed from the list
    //this method might require a node to be removed from the list
    
}
public int find(int x) {
    //if x is in the list return the number of times (counts value) x is in the list
    //if x is not in the list return 0
    }

    public String toString() {
        //return a string representation of the list
        //the string representation each int in the list separated by commas
        //if an int appear 3 times in the list (i.e. its counts field is 3)
        //then it should appear three times in the string representation
    }

    public void close() {
        //update values (such as head and free) as needed and close the random access file
        //The list cannot be used after it is closed
    }
Split a Node

• As a result of an insert you might need to split a node
• A node needs to be split when you want to insert a new int into the node but the node is full
• When you need to split put half the values that are part of the split into a new node and leave half the values in the old node
• Make sure to adjust the count value and the next value in the nodes as needed and to shift values in the old node as needed
Remove a Node

• When the last int in a node is removed the node should be removed for the list and added as the new first node in the free list
Homework 3 Submission

• You will demonstrate your program to me either on your own machine or on a machine in the CS lab
• If you demo on or before Feb. 24 and your program works on my test programs you can get 5 bonus points.
• If you demo between Monday Feb. 27 and Thursday Mar. 2 the program is on time. I will not be able to do demos for everyone on the last day so you should not wait until the last day to demo.
• You can demo from Friday Mar. 3 until Thursday Mar. 9 for up to 50% of the points.