University of Glasgow

Tuesday, 2 May 2017
2.00 pm – 3.30 pm
(Duration: 1 hour 30 minutes)

DEGREES OF MSci, MEng, BEng, BSc, MA and MA (Social Sciences)

Algorithms and Data Structures 2

(Answer all questions.)

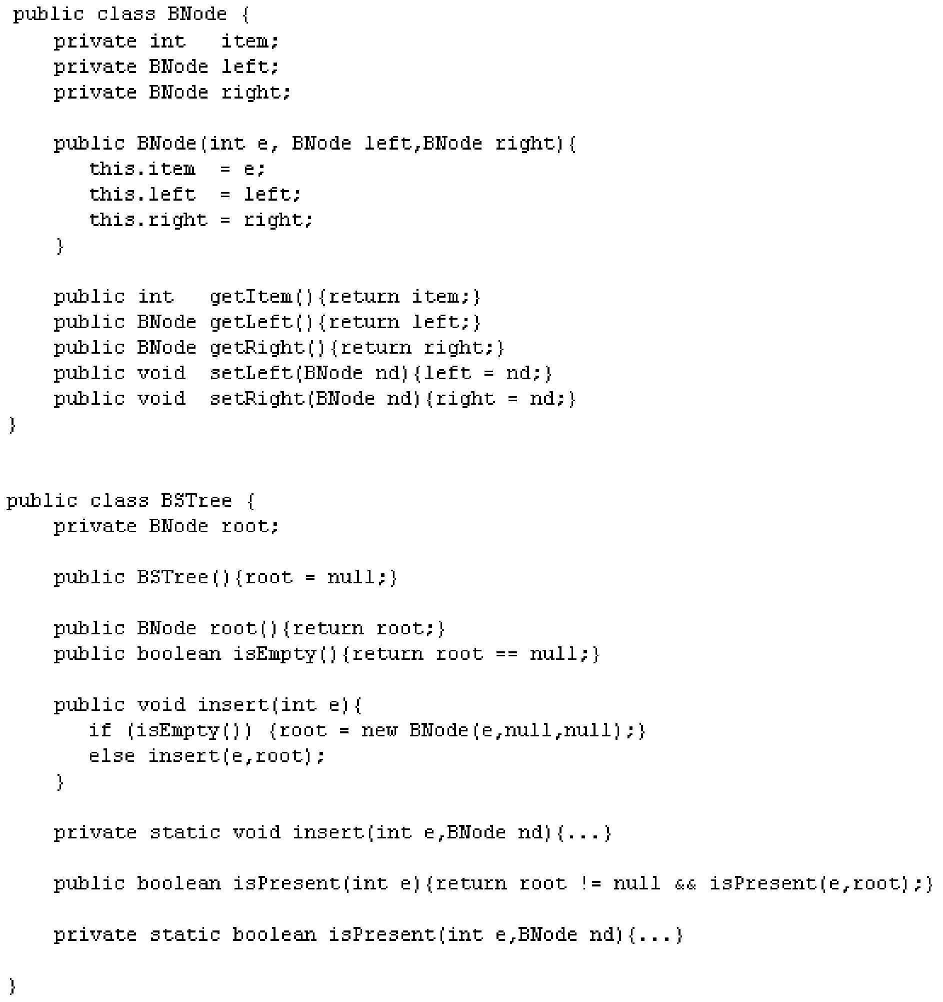
This examination paper is worth a total of 50 marks

The use of a calculator is not permitted in this examination

INSTRUCTIONS TO INVIGILATORS

Please collect all exam question papers and exam answer scripts and retain for school to collect. Candidates must not remove exam question papers.

1. A binary search tree is a binary tree T such that each node of T stores an item e. Items stored in the left subtree of T rooted at a node v are less than the item in node v, and items stored in the right subtree of T rooted at a node v are greater than the item in node v. Below is java code for the `BNode` class, and below that code for the `BSTree` class which together implement a binary search tree suitable for storing a set of integers.



```
public class BNode {
   private int   item;
   private BNode left;
   private BNode right;

   public BNode(int e, BNode left,BNode right){
      this.item  = e;
      this.left  = left;
      this.right = right;
   }

   public int   getItem(){return item;}
   public BNode getLeft(){return left;}
   public BNode getRight(){return right;}
   public void  setLeft(BNode nd){left = nd;}
   public void  setRight(BNode nd){right = nd;}
}


public class BSTree {
   private BNode root;

   public BSTree(){root = null;}

   public BNode root(){return root;}
   public boolean isEmpty(){return root == null;}

   public void insert(int e){
      if (isEmpty()) {root = new BNode(e,null,null);}
      else insert(e,root);
   }

   private static void insert(int e,BNode nd){...}

   public boolean isPresent(int e){return root != null && isPresent(e,root);}

   private static boolean isPresent(int e,BNode nd){...}

}
```

(a) Write Java code for the method `insert(int e,BNode nd)` in class `BSTree`, where the method inserts the integer `e` into the tree rooted at node `nd`.

[4]

(b) Write Java code for the method `isPresent(int e,Bnode nd)`, where the method delivers true if and only if `e` is in the tree rooted at node `nd`.

[4]

(c) Assume that the following items are inserted into an empty `BSTree` in the following order: 30, 40, 24, 58, 48, 26, 11, 24, 13, 36.

- Draw the tree.
- What is the height of the tree?
- Write out the preorder, inorder and postorder traversals of the tree.

[5]

(d) How might we modify the `BNode` class such that our binary search tree can represent a multiset? In your answer consider insertion and deletion of entries. Note, Java code is not expected.

[3]

2. An organisation has a data set of 1 million customers. The information the organisation holds on customers includes their height in centimetres (cm). The organisation wants to sort that data using height as a key, where height is an integer in the range 100cm to 220cm. This might be done using a pigeonhole sort or a radix sort.

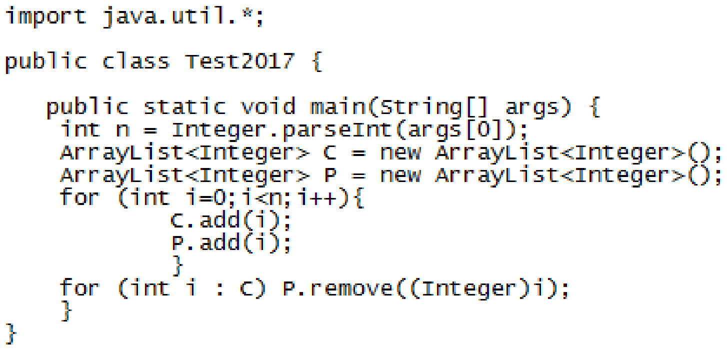
(a) What is a pigeonhole sort? What is its complexity? Why might it be suitable for this task? Assuming Java is being used, what data structures might you use?

[8]

(b) What is radix sort? What is its complexity? Why might it be suitable for this task?

[6]

3. The toy program below, `Test2017`, takes as argument on the command line an integer (`n`). Two ArrayLists, `C` and `P`, are then created and both are filled with the integers `0` to `n-1`. Finally, the contents of `C` are removed from `P`.



```
import java.util.*;

public class Test2017 {

   public static void main(String[] args) {
    int n = Integer.parseInt(args[0]);
    ArrayList<Integer> C = new ArrayList<Integer>();
    ArrayList<Integer> P = new ArrayList<Integer>();
    for (int i=0;i<n;i++){
            C.add(i);
            P.add(i);
            }
    for (int i : C) P.remove((Integer)i);
    }
}
```

(a) What is the complexity of `Test2017`? Explain your answer.

[2]

(b) We are using a “for-each” loop to remove values from `P`. What are the consequences of that on space and run time?

[2]

(c) When creating the ArrayLists we might have specified their size. What effect might that have on run time? Explain your answer.

[2]

4. In Java, we might represent a set of integers using (i) a BitSet, (ii) a HashSet, (iii) a TreeSet or (iv) a LinkedList.

(a) What is the complexity of adding an element to the set when it is represented as a BitSet, a HashSet, a TreeSet or a LinkedList? Explain your answers.

[4]

(b) What is the complexity of removing an element from the set, for each of the four possible representations? Explain your answers.

[4]

(c) Given two sets S1 and S2 with the same representation, what will be the complexity of the set intersection operation for each of the four possible representations? Explain your answers.

[6]