CS 3310 – Data and File Structures, Instructor: *<Gupta>*, Western Michigan University Lab TA: *<YG>*

Parker Loomis

**SOFTWARE LIFE CYCLE REPORT – FOR LAB ASSIGNMENT** *3*

**Priority Queues**

**PHASE 1: SPECIFICATION (“What do we build?”)**

1. In class we discussed various ways to implement priority queues. Two prominent ones are using heaps in their explicit or implicit representations. We will practice implementing these and compare their advantages and disadvantages. In particular, implement (i) a min-heap using a size-balanced binary tree SBT in its explicit representation, and (b) a min-heap using an array-based complete binary tree CBT.  Write a JAVA application to solve the following problem,
   1. 1)  Given are a sequence of heap-operations namely, “findmin”, “deletemin”, and “insert <key, value>” where key is an integer and value is also an integer (in practice value would be a reference / pointer to the record but for this homework assignment this simplification will suffice)
   2. 2)  Execute each operation on SBT and CBT, i..e,, if the operation is deletemin, delete the record with the smallest key form SBT and CBT; if the operation is “insert <key, value>”, then insert the record with the integer key along with its value in the min-heaps SBT and CBT; obviously if the operation is findmin, then simply return the appropriate record from each of CBT and SBT.
   3. 3)  In order to empirically compare the computational requirements of the two implementations, we will randomly generate data and the sequence of operations (in practice these come from another application). Generation of data
      1. Operation – randomly generate an integer x in the range [1, 10000], if x **ε** [1, 2000], then the operation is insert, if x **ε** [2001, 4000], then the operation is deletemin, and if x **ε** [4001, 10000] then the operation is a findmin. Note that we can always vary the lower and upper bounds of these ranges to vary the probability of each operation.
      2. Key – a randomly generated integer in the range [1, 1000]

1

.

CS 3310 Spring 2017 A3 Priority Queues and Heaps c. Value – a randomly generated integer in the range [5000, 20000]

d. Repeat a-c for the desired data sizes (i.e., number of operations or the length of the sequence of operations) in your experiment

* 4)  Measure the average execution time for each of the operations in each experiment.
* 5)  Measure the total time for the sequence of operations generated in an experiment.
* 6)  Run at least 100 different experiments with sequence lengths 100, 200, 300, ....
* 7)  Plot the measured times in items 4) and 5) above (so the x-axis would be the sequence lengths and y-axis would be measured times). Compare these. Which one is better CBT or SBT? For what sizes? Do the observed times agree with theoretical analysis?
* 8)  Just so we can easily make sure your implementations are working correctly, print the before- and after-states of CBT and SBT when the heap-size reaches 30 for the first time. That is print CBT and SBT before and after each of the operations, insert, deletemin, and findmin, CBT is simply an array so just print the <key, value> pairs separated by commas as they are stored in your CBT. SBT uses explicit representation of binary trees with an additional “size” field at every node, so print SBT twice, once using a preorder traversal with node-data separated by commas (as in [<key1, value1>, size1], [<key2, value2>, size2], ...) and then using an inorder traversal (*pause: why do we want SBT printed using different traversals?*). Thus, in total, there will be only six prints of CBT (once before and once after each heap-operation) and twelve prints of SBT (twice before and twice after each heap-operation) in this assignment. Exclude “printing” time from your measurements.

**PHASE 2: DESIGN**

**Program should have 7 classes**

1. **Main method**
2. **SBT**
3. **LinkedListBST**
4. **Node**

**PHASE 3: RISK ANALYSIS (“What can go wrong, and how bad can it be?”)**

Heap could go wrong. Printing could go to wrong node.

**PHASE 4: VERIFICATION (“Are the algorithms correct?”)**

Size balanced doesn’t work, and printing doesn’t work like it should.

**PHASE 5: CODING**

**package** main;

**import** java.util.Random;

**import** java.util.Scanner;

**import** minHeap.MinHeap;

**public** **class** Main {

**public** Main() {

// **TODO** Auto-generated constructor stub

}

**public** **static** **void** main(String[] args) {

**long** start = System.*currentTimeMillis*();;

// **TODO** Auto-generated method stub

Random rand = **new** Random();// call random class

Scanner kbd = **new** Scanner(System.***in***);

//kbd.close();

**int** i= 100;//number of integers in tree

**int** k= 10000;//max number int node can be

**int** l;

//int key=5;

**int** key= rand.nextInt(1000);

**int** operation= (rand.nextInt(9999)+1);

//generates random numbers. adds to list and array, and prints out tree

MinHeap minHeap = **new** MinHeap(i);

minHeap.insert(5);

//System.out.println("The Min Heap is ");

**for** (**int** j = 0; j < i-1; j++) {

l=rand.nextInt(k);

//System.out.print(l+", ");

minHeap.insert(l);

}

**while**(minHeap.getsize()>0){

**if**(minHeap.getsize()>=30){

minHeap.print();

}

System.***out***.println("What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit. ");

**int** ans=kbd.nextInt();

**int** value=(rand.nextInt(15000)+5000);

**if**(ans==1) { //insert key,print

System.***out***.println("Entering the value "+value+ " into tree.");

minHeap.insert(value);

}

**else** **if**(ans==2) {//delete min,print

minHeap.remove();

System.***out***.println("Tree has "+minHeap.getsize()+" nodes.");

}

**else** **if**(ans==3) {//find min,print

System.***out***.println("Min value is "+minHeap.min()+".");

}

**else** **if**(ans==4) {

**int** x= rand.nextInt(10000);

**if**(x<=2000){

System.***out***.println("Entering the value "+value+ " into tree.");

minHeap.insert(value);

//insert key,print

}

**else** **if**(x>2000 &&x<=4000){

minHeap.remove();

System.***out***.println("Tree has "+minHeap.getsize()+" nodes.");

}

**else** **if**(x>4000){

//find min,print

System.***out***.println("Min value is "+minHeap.min()+".");

}

}**else** **if**(ans==5){

**break**;

}

**else**{

System.***out***.println("Incorrect choice.");

}

}

minHeap.print();

// System.out.println("The Min val is " + minHeap.remove());

// System.out.println("The Min val is " + minHeap.min());

**long** end = System.*currentTimeMillis*();;

System.***out***.println((end - start) + " ms");

}

}

**package** minHeap;

**public** **class** MinHeap {

**private** **int**[] Heap;

**private** **int** size;

**private** **int** maxsize;

**private** **static** **final** **int** ***FRONT*** = 1;

**public** MinHeap(**int** maxsize)

{

**this**.maxsize = maxsize;

**this**.size = 0;

Heap = **new** **int**[**this**.maxsize + 1];

Heap[0] = Integer.***MIN\_VALUE***;

}

**private** **int** parent(**int** pos)

{

**return** pos / 2;

}

**private** **int** leftChild(**int** pos)

{

**return** (2 \* pos);

}

**private** **int** rightChild(**int** pos)

{

**return** (2 \* pos) + 1;

}

**private** **boolean** isLeaf(**int** pos)

{

**if** (pos >= (size / 2) && pos <= size)

{

**return** **true**;

}

**return** **false**;

}

**private** **void** swap(**int** fpos, **int** spos)

{

**int** tmp;

tmp = Heap[fpos];

Heap[fpos] = Heap[spos];

Heap[spos] = tmp;

}

**private** **void** minHeapify(**int** pos)

{

**if** (!isLeaf(pos))

{

**if** ( Heap[pos] > Heap[leftChild(pos)] || Heap[pos] > Heap[rightChild(pos)])

{

**if** (Heap[leftChild(pos)] < Heap[rightChild(pos)])

{

swap(pos, leftChild(pos));

minHeapify(leftChild(pos));

}**else**

{

swap(pos, rightChild(pos));

minHeapify(rightChild(pos));

}

}

}

}

**public** **void** insert(**int** element)

{

**if**(size==maxsize){

expand();

}

Heap[++size] = element;

**int** current = size;

**while** (Heap[current] < Heap[parent(current)])

{

swap(current,parent(current));

current = parent(current);

}

}

**public** **void** expand(){

**int** [] newArray = **new** **int** [Heap.length + 1];

//System.arraycopy(Heap, 0, newArray, 0, Heap.length);

// an alternative to using System.arraycopy would be a for-loop:

**for**(**int** i = 0; i < **this**.Heap.length; i++)

newArray[i] = **this**.Heap[i];

**this**.Heap = newArray;

**this**.maxsize++;

}

**public** **void** print()

{

**for** (**int** i = 1; i <= size; i++ )

{

// System.out.print(" PARENT : " + Heap[i] + " LEFT CHILD : " + Heap[2\*i]

// + " RIGHT CHILD :" + Heap[2 \* i + 1]);

System.***out***.print(Heap[i]+", ");

}

System.***out***.print("\n");

}

**public** **void** minHeap()

{

**for** (**int** pos = (size / 2); pos >= 1 ; pos--)

{

minHeapify(pos);

}

}

**public** **int** remove()

{

**int** popped = Heap[***FRONT***];

Heap[***FRONT***] = Heap[size--];

minHeapify(***FRONT***);

**return** popped;

}

**public** **int** min()

{

**int** popped = Heap[***FRONT***];

minHeapify(***FRONT***);

**return** popped;

}

**public** **int** getsize(){

**return** size;

}

}

**package** sbt;

/\*

\* Java Program to Implement a Binary Search Tree using Linked Lists

\*/

**import** java.util.Scanner;

/\* Class Node \*/

**class** Node

{

Node left, right;

**int** data;

/\* Constructor \*/

**public** Node(**int** n)

{

left = **null**;

right = **null**;

data = n;

}

}

/\* Class BST \*/

**class** BST

{

**private** Node root;

/\* Constructor \*/

**public** BST()

{

root = **null**;

}

/\* Functions to insert data \*/

**public** **void** insert(**int** data)

{

root = insert(root, data);

}

/\* Function to insert data recursively \*/

**private** Node insert(Node node, **int** data)

{

**if** (node == **null**)

node = **new** Node(data);

**else**

{

**if** (data <= node.data)

node.left = insert(node.left, data);

**else**

node.right = insert(node.right, data);

}

**return** node;

}

/\* Function for inorder traversal \*/

**public** **void** inorder()

{

inorder(root);

}

**private** **void** inorder(Node r)

{

**if** (r != **null**)

{

inorder(r.left);

System.***out***.print(r.data +" ");

inorder(r.right);

}

}

/\* Function for preorder traversal \*/

**public** **void** preorder()

{

preorder(root);

}

**private** **void** preorder(Node r)

{

**if** (r != **null**)

{

System.***out***.print(r.data +" ");

preorder(r.left);

preorder(r.right);

}

}

/\* Function for postorder traversal \*/

**public** **void** postorder()

{

postorder(root);

}

**private** **void** postorder(Node r)

{

**if** (r != **null**)

{

postorder(r.left);

postorder(r.right);

System.***out***.print(r.data +" ");

}

}

}

/\* Class LinkedListBST \*/

**package** sbt;

**import** java.util.Random;

**import** java.util.Scanner;

**public** **class** LinkedListBST

{

**public** **static** **void** main(String[] args)

{

Scanner scan = **new** Scanner(System.***in***);

/\* Creating object of BST \*/

BST bst = **new** BST();

System.***out***.println("Linked List Binary Search Tree Test\n");

**char** ch;

/\* Accept input \*/

**int** i= 100;//number of integers in tree

**int** k= 10000;//max number int node can be

**int** l;

Random rand = **new** Random();

**for** (**int** j = 0; j < i-1; j++) {

l=rand.nextInt(k);

//System.out.print(l+", ");

bst.insert(l);

}

**do**

{

System.***out***.println("Enter integer element to insert");

bst.insert( scan.nextInt() );

/\* Display tree \*/

System.***out***.print("\nPost order : ");

bst.postorder();

System.***out***.print("\nPre order : ");

bst.preorder();

System.***out***.print("\nIn order : ");

bst.inorder();

System.***out***.println("\nDo you want to continue (Type y or n) \n");

ch = scan.next().charAt(0);

} **while** (ch == 'Y'|| ch == 'y');

}

}

**PHASE 6: TESTING (“Did we build it correctly?”)**

5, 387, 219, 530, 806, 973, 1186, 3300, 721, 1628, 1438, 2747, 1926, 1712, 4271, 3757, 3823, 1505, 839, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 3920, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 1762, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 4661, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 3610, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

1

Entering the value 16450 into tree.

5, 387, 219, 530, 806, 973, 1186, 3300, 721, 1628, 1438, 2747, 1926, 1712, 4271, 3757, 3823, 1505, 839, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 3920, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 1762, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 4661, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 3610, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

1

Entering the value 9984 into tree.

5, 387, 219, 530, 806, 973, 1186, 3300, 721, 1628, 1438, 2747, 1926, 1712, 4271, 3757, 3823, 1505, 839, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 3920, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 1762, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 4661, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 3610, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

1

Entering the value 7732 into tree.

5, 387, 219, 530, 806, 973, 1186, 3300, 721, 1628, 1438, 2747, 1926, 1712, 4271, 3757, 3823, 1505, 839, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 3920, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 1762, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 4661, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 3610, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984, 7732,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

1

Entering the value 16943 into tree.

5, 387, 219, 530, 806, 973, 1186, 3300, 721, 1628, 1438, 2747, 1926, 1712, 4271, 3757, 3823, 1505, 839, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 3920, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 1762, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 4661, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 3610, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984, 7732, 16943,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 103 nodes.

219, 387, 973, 530, 806, 1926, 1186, 3300, 721, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 839, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 1762, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 3610, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984, 7732,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 102 nodes.

387, 530, 973, 721, 806, 1926, 1186, 3300, 839, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

3

Min value is 387.

387, 530, 973, 721, 806, 1926, 1186, 3300, 839, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

3

Min value is 387.

387, 530, 973, 721, 806, 1926, 1186, 3300, 839, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

3

Min value is 387.

387, 530, 973, 721, 806, 1926, 1186, 3300, 839, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

4

Min value is 387.

387, 530, 973, 721, 806, 1926, 1186, 3300, 839, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

4

Min value is 387.

387, 530, 973, 721, 806, 1926, 1186, 3300, 839, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

3

Min value is 387.

387, 530, 973, 721, 806, 1926, 1186, 3300, 839, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 1505, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 2971, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 4062, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450, 9984,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

4

Tree has 101 nodes.

530, 721, 973, 839, 806, 1926, 1186, 3300, 1505, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 2971, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 4062, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 9984, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

3

Min value is 530.

530, 721, 973, 839, 806, 1926, 1186, 3300, 1505, 1628, 1438, 2747, 3920, 1712, 4271, 3757, 3823, 2971, 1762, 3017, 1928, 1810, 1596, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 4062, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 2962, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 9984, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 6428, 9196, 6989, 9509, 7540, 9644, 9044, 7078, 16450,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 100 nodes.

721, 806, 973, 839, 1438, 1926, 1186, 3300, 1505, 1628, 1596, 2747, 3920, 1712, 4271, 3757, 3823, 2971, 1762, 3017, 1928, 1810, 2962, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 4062, 4350, 3610, 4930, 4574, 3640, 2275, 2285, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 9984, 5782, 6914, 4372, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 16450, 9196, 6989, 9509, 7540, 9644, 9044, 7078,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 99 nodes.

806, 839, 973, 1505, 1438, 1926, 1186, 3300, 1762, 1628, 1596, 2747, 3920, 1712, 4271, 3757, 3823, 2971, 3610, 3017, 1928, 1810, 2962, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 4062, 4350, 4372, 4930, 4574, 3640, 2275, 2285, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 7029, 8288, 8162, 8957, 16450, 9196, 6989, 9509, 7540, 9644, 9044,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 98 nodes.

839, 1438, 973, 1505, 1596, 1926, 1186, 3300, 1762, 1628, 1810, 2747, 3920, 1712, 4271, 3757, 3823, 2971, 3610, 3017, 1928, 2285, 2962, 5610, 3644, 4698, 4661, 5996, 2039, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 4062, 4350, 4372, 4930, 4574, 3640, 2275, 7029, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 5004, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 9044, 8288, 8162, 8957, 16450, 9196, 6989, 9509, 7540, 9644,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 97 nodes.

973, 1438, 1186, 1505, 1596, 1926, 1712, 3300, 1762, 1628, 1810, 2747, 3920, 2039, 4271, 3757, 3823, 2971, 3610, 3017, 1928, 2285, 2962, 5610, 3644, 4698, 4661, 5996, 5004, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 4062, 4350, 4372, 4930, 4574, 3640, 2275, 7029, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 9044, 8288, 8162, 8957, 16450, 9196, 6989, 9509, 7540,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 96 nodes.

1186, 1438, 1712, 1505, 1596, 1926, 2039, 3300, 1762, 1628, 1810, 2747, 3920, 5004, 4271, 3757, 3823, 2971, 3610, 3017, 1928, 2285, 2962, 5610, 3644, 4698, 4661, 5996, 7540, 4357, 5564, 6113, 4469, 4955, 4929, 3459, 4062, 4350, 4372, 4930, 4574, 3640, 2275, 7029, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 6592, 9147, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 9044, 8288, 8162, 8957, 16450, 9196, 6989, 9509,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 95 nodes.

1438, 1505, 1712, 1762, 1596, 1926, 2039, 3300, 2971, 1628, 1810, 2747, 3920, 5004, 4271, 3757, 3823, 3459, 3610, 3017, 1928, 2285, 2962, 5610, 3644, 4698, 4661, 5996, 7540, 4357, 5564, 6113, 4469, 4955, 4929, 6592, 4062, 4350, 4372, 4930, 4574, 3640, 2275, 7029, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 9509, 9147, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 5630, 8092, 9992, 9044, 8288, 8162, 8957, 16450, 9196, 6989,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 94 nodes.

1505, 1596, 1712, 1762, 1628, 1926, 2039, 3300, 2971, 1928, 1810, 2747, 3920, 5004, 4271, 3757, 3823, 3459, 3610, 3017, 2275, 2285, 2962, 5610, 3644, 4698, 4661, 5996, 7540, 4357, 5564, 6113, 4469, 4955, 4929, 6592, 4062, 4350, 4372, 4930, 4574, 3640, 5630, 7029, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 9509, 9147, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 6989, 8092, 9992, 9044, 8288, 8162, 8957, 16450, 9196,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 93 nodes.

1596, 1628, 1712, 1762, 1810, 1926, 2039, 3300, 2971, 1928, 2285, 2747, 3920, 5004, 4271, 3757, 3823, 3459, 3610, 3017, 2275, 7029, 2962, 5610, 3644, 4698, 4661, 5996, 7540, 4357, 5564, 6113, 4469, 4955, 4929, 6592, 4062, 4350, 4372, 4930, 4574, 3640, 5630, 9044, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 9509, 9147, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 6989, 8092, 9992, 9196, 8288, 8162, 8957, 16450,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 92 nodes.

1628, 1762, 1712, 2971, 1810, 1926, 2039, 3300, 3459, 1928, 2285, 2747, 3920, 5004, 4271, 3757, 3823, 4062, 3610, 3017, 2275, 7029, 2962, 5610, 3644, 4698, 4661, 5996, 7540, 4357, 5564, 6113, 4469, 4955, 4929, 6592, 9147, 4350, 4372, 4930, 4574, 3640, 5630, 9044, 7665, 6428, 6201, 7505, 6170, 4213, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 9509, 16450, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 6989, 8092, 9992, 9196, 8288, 8162, 8957,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

2

Tree has 91 nodes.

1712, 1762, 1926, 2971, 1810, 2747, 2039, 3300, 3459, 1928, 2285, 3644, 3920, 5004, 4271, 3757, 3823, 4062, 3610, 3017, 2275, 7029, 2962, 5610, 4213, 4698, 4661, 5996, 7540, 4357, 5564, 6113, 4469, 4955, 4929, 6592, 9147, 4350, 4372, 4930, 4574, 3640, 5630, 9044, 7665, 6428, 6201, 7505, 6170, 8957, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 9509, 16450, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 6989, 8092, 9992, 9196, 8288, 8162,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

1

Entering the value 9561 into tree.

1712, 1762, 1926, 2971, 1810, 2747, 2039, 3300, 3459, 1928, 2285, 3644, 3920, 5004, 4271, 3757, 3823, 4062, 3610, 3017, 2275, 7029, 2962, 5610, 4213, 4698, 4661, 5996, 7540, 4357, 5564, 6113, 4469, 4955, 4929, 6592, 9147, 4350, 4372, 4930, 4574, 3640, 5630, 9044, 7665, 6428, 6201, 7505, 6170, 8957, 4641, 9938, 9899, 5249, 16943, 7572, 6085, 9644, 9113, 6990, 8701, 6773, 6667, 8276, 9196, 7805, 5915, 9602, 6047, 7910, 9506, 7842, 9509, 16450, 9984, 5782, 6914, 7078, 7732, 9430, 7247, 8507, 8085, 9121, 4684, 6989, 8092, 9992, 9196, 8288, 8162, 9561,

What would you like to do? 1 for inserting into the tree, 2 to delete the min, 3 to find the min, 4 to generate a random number, and 5 to exit.

**PHASE 7: REFINING THE PROGRAM (“Add bells and whistles to the program”)**

Make it look better and fix any problems.

**PHASE 8: PRODUCTION**

I prepared a copy of the entire program for Lab TA’s evaluation, as specified by the TA. Then, I sent electronically the copy to the Lab TA.

**PHASE 9: MAINTENANCE**

Fix the issues with Linked List and printing.

**TIME AND SPACE**

Space would be depending on the amount of numbers the user wanted. The number of array elements plus any of the types.

Main: O(n)

SBT: O(log n)

LinkedListBST: O(n)

Node: O(n)