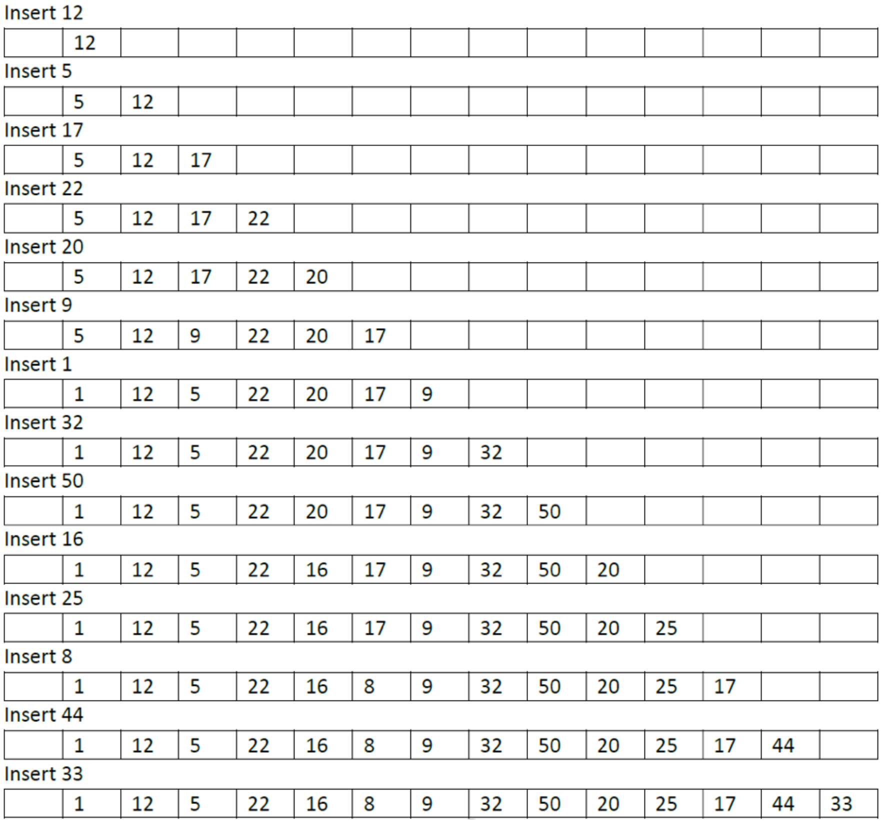
CSC 212 Tutorial #13 Solution

Heap

Problem 1



Insert 12

	12													

Insert 5

	5	12												

Insert 17

	5	12	17											

Insert 22

	5	12	17	22										

Insert 20

	5	12	17	22	20									

Insert 9

	5	12	9	22	20	17								

Insert 1

	1	12	5	22	20	17	9							

Insert 32

	1	12	5	22	20	17	9	32						

Insert 50

	1	12	5	22	20	17	9	32	50					

Insert 16

	1	12	5	22	16	17	9	32	50	20				

Insert 25

	1	12	5	22	16	17	9	32	50	20	25			

Insert 8

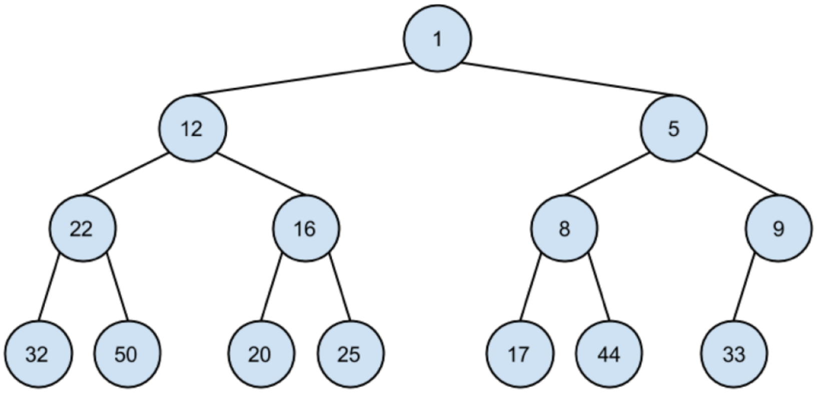
	1	12	5	22	16	8	9	32	50	20	25	17		

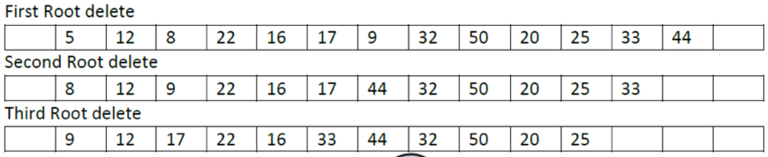
Insert 44

	1	12	5	22	16	8	9	32	50	20	25	17	44	

Insert 33

	1	12	5	22	16	8	9	32	50	20	25	17	44	33




First Root delete

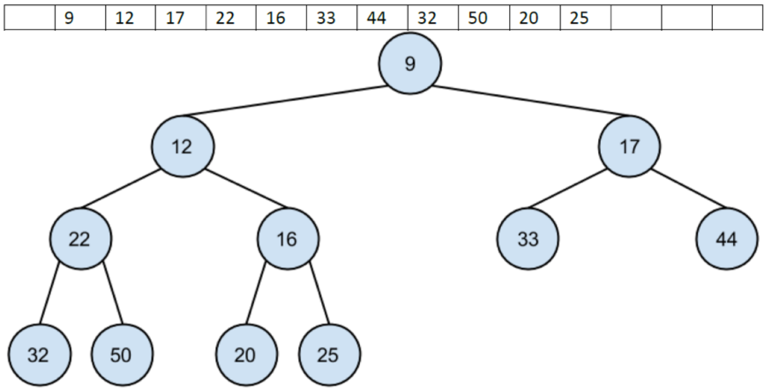
	5	12	8	22	16	17	9	32	50	20	25	33	44	

Second Root delete

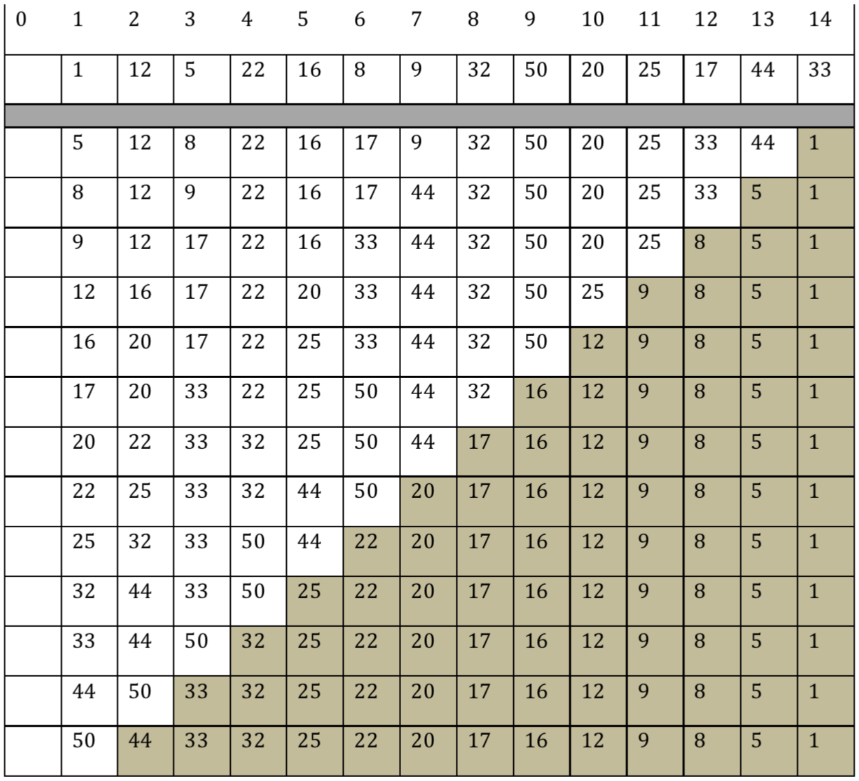
	8	12	9	22	16	17	44	32	50	20	25	33		

Third Root delete

	9	12	17	22	16	33	44	32	50	20	25			

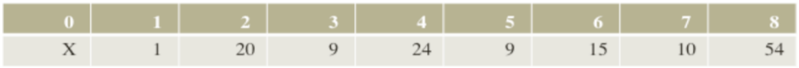


```
public void sort() {
    int n = size;
    for (int i = 1; i < n; i++) {
        int tmpKey = keys[1];
        T tmpData = data[1];
        keys[1]= keys[size];
        data[1]= data[size];
        size--;
        siftDown(1);
        keys[size + 1] = tmpKey;
        data[size + 1] = tmpData;
    }
}
```

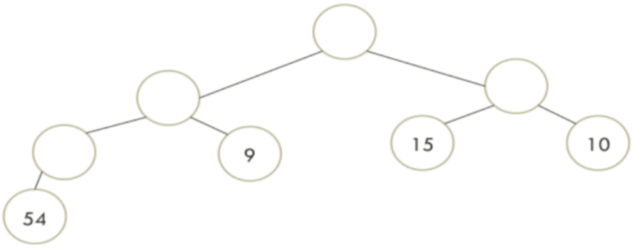


0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	1	12	5	22	16	8	9	32	50	20	25	17	44	33
	5	12	8	22	16	17	9	32	50	20	25	33	44	1
	8	12	9	22	16	17	44	32	50	20	25	33	5	1
	9	12	17	22	16	33	44	32	50	20	25	8	5	1
	12	16	17	22	20	33	44	32	50	25	9	8	5	1
	16	20	17	22	25	33	44	32	50	12	9	8	5	1
	17	20	33	22	25	50	44	32	16	12	9	8	5	1
	20	22	33	32	25	50	44	17	16	12	9	8	5	1
	22	25	33	32	44	50	20	17	16	12	9	8	5	1
	25	32	33	50	44	22	20	17	16	12	9	8	5	1
	32	44	33	50	25	22	20	17	16	12	9	8	5	1
	33	44	50	32	25	22	20	17	16	12	9	8	5	1
	44	50	33	32	25	22	20	17	16	12	9	8	5	1
	50	44	33	32	25	22	20	17	16	12	9	8	5	1

Problem 2

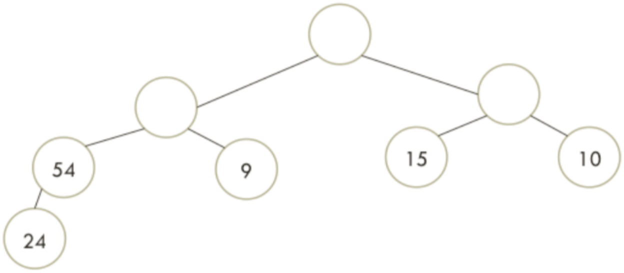


0	1	2	3	4	5	6	7	8
X	1	20	9	24	9	15	10	54



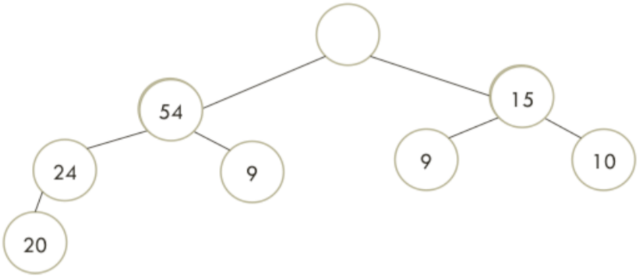
Draw an empty complete binary tree of 8 nodes; fill the leaf nodes with elements from the back of the array

0	1	2	3	4	5	6	7	8
X	1	20	9	54	9	15	10	24



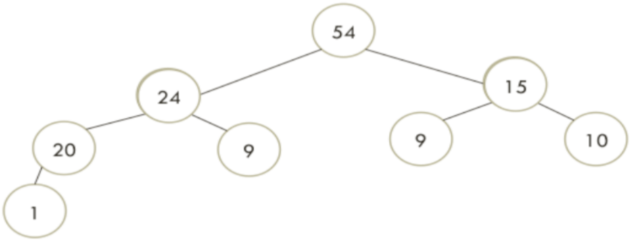
Add 24 → sift down

0	1	2	3	4	5	6	7	8
X	1	54	15	24	9	9	10	20



Add 9 & 20 from right to left → sift down

0	1	2	3	4	5	6	7	8
X	54	24	15	20	9	9	10	1



Add 1 in root → sift down

Problem 3

- Sorted according to the heap property: O(n)
- Sorted in the inverse of the heap property: O(nlogn)

Problem 4

```java
public static boolean isMaxBinaryHeap(int[] elements, int size) {
    for (int  i = 1; i <= size / 2; i++){
        if (i * 2 <= size)
            if (elements[i * 2] > elements[i])
                return false;
        if (i * 2 + 1 <= size)
            if (elements[i * 2 + 1] > elements[i])
                return false;
    }
    return true;
}
```