CSE 3010 – Data Structures & Algorithms

Lecture #45

What will be covered today

- Priority queue and implementations
- Using heap data structure

Handling priorities using ADT Priority Queue

Method 1: Using the LIST data structure

Method 2: Using the QUEUE data structure

```
typedef int ITEM;

typedef struct node {
         ITEM item;
         int priority;
         struct node *next;
} QNODE;
```

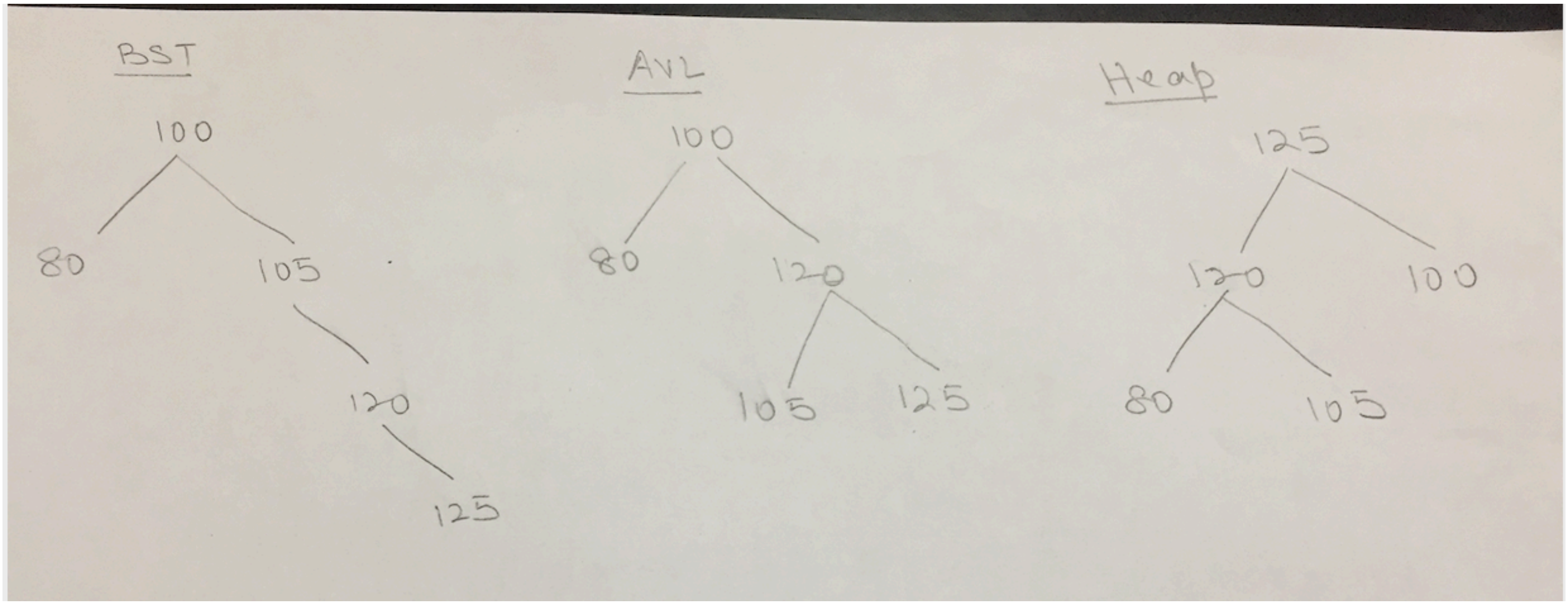
Method 3: Using HEAP data structure – Max or Min HEAP

Heap data structure

- Is a binary tree with an item in each node
- Is an almost complete or complete binary tree
- Properties of heap
    - All leaves of the tree are on two adjacent levels
    - All leaves on the lowest level occur to the left
    - All levels, except the lowest level, are filled
    - Item in the root is at least as large (small) as their children
    - Left and right subtrees are heaps
- Is weakly ordered
    - No specific organization of keys like binary search tree
    - Traversal and other operations are not either difficult or impossible

Difference between BST, AVL tree and Heap

Example: 100, 80, 105, 120, 125



Implementation of HEAP data structure

- Using an array
- Other than root children can be arrived at by:

  `2 * i + 1` (left child)

  `2 * i + 2` (right child)
- Parent node of a child can be arrived at by:

  `(i – 1) / 2` (for left child)

  `(i – 2) / 2` (for right child)
- No holes in the array as HEAP is an almost complete or complete binary tree
- Inserting an item into HEAP uses 'trickling' or 'bubbling' up
- Deleting an item form HEAP uses 'trickling' or 'bubbling' down