Lecture 5 Search Algorithms

- **Array & Tree**
    - Use arrays to store the tree
    - For Binary Trees
        - Left-child: A[2 * i + 1]
        - Right-child: A[2 * i + 2]
        - Parent: A[⌊(i – 1)/2⌋]
- **Binary Search**
    - Recursive
    - mid = ⌊(low + high)/2⌋
    - Cases
        - ❶ k = key(mid), the search is completed
        - ❷ k < key(mid), search continued, high = mid - 1
        - ❸ k > key(mid), search continued, low = mid + 1
    - Complexity
        - O(log n)
- **Binary Search Tree (BST)**
    - All elements in the left subtree of a node v are less than or equal to its element e
    - All elements in the right subtree of a node v are greater than or equal to its element e
    - Methods
        - findElement(e)
            - O(h)
        - insertItem(k, o)
            - O(h)
    - Performance
        - Height h
            - Worst case: O(n)
            - Best case: O(log n)
        - Space Complexity
            - O(n)
    - Inefficiency
        - Not balanced (for example, a degenerate tree)