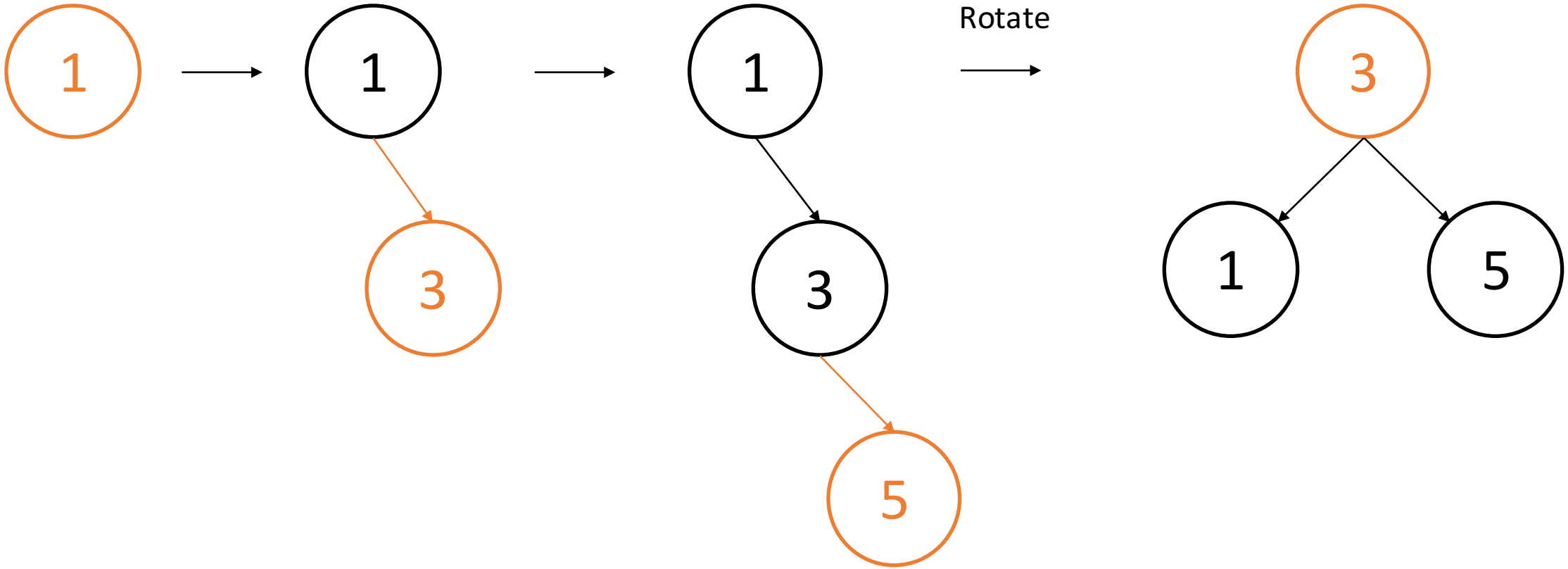
演算法 #Hw4

賴冠廷 陳昀靖

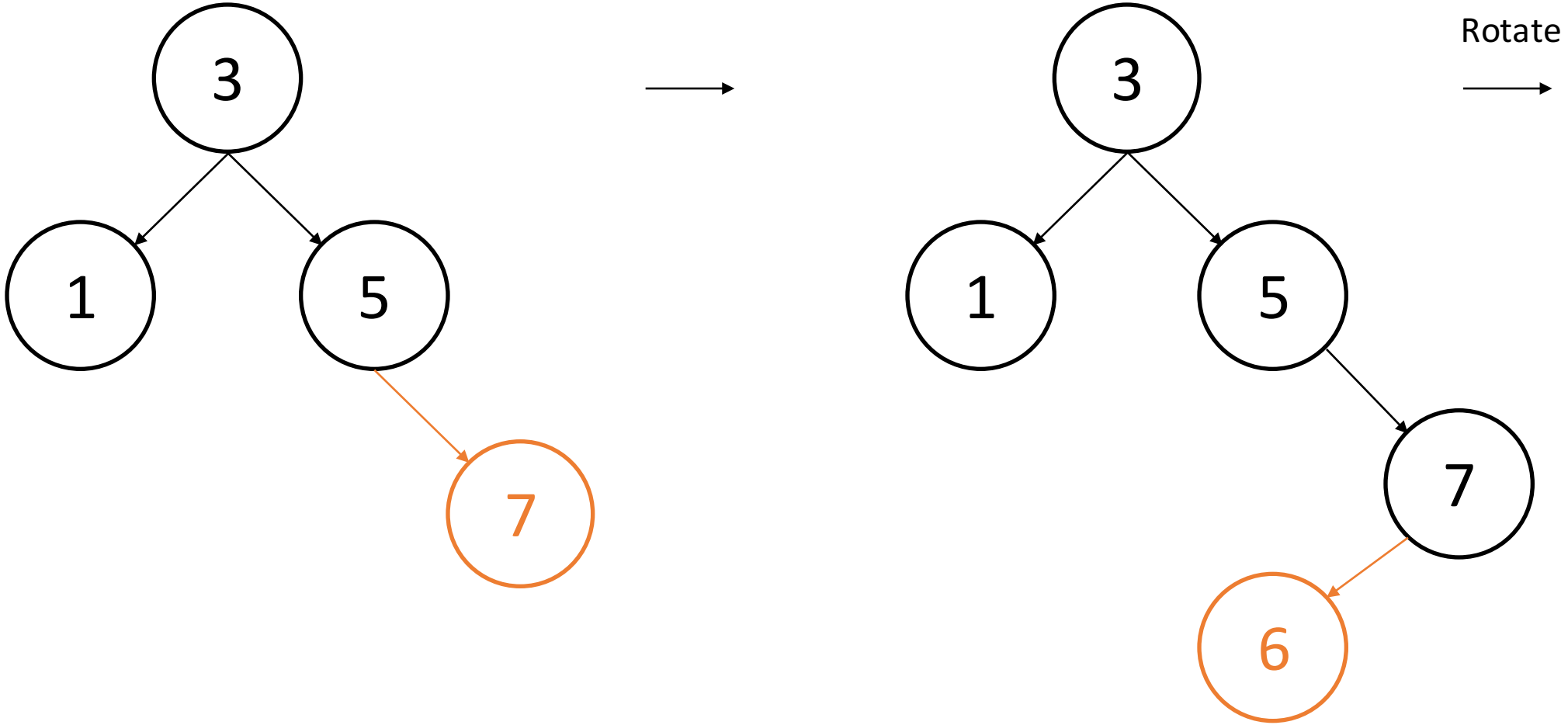
Q1.

- Show the AVL tree formed by inserting numbers 1, 3, 5, 7, 6, 4 in order. For each insertion, show the AVL trees before and after each rotation.

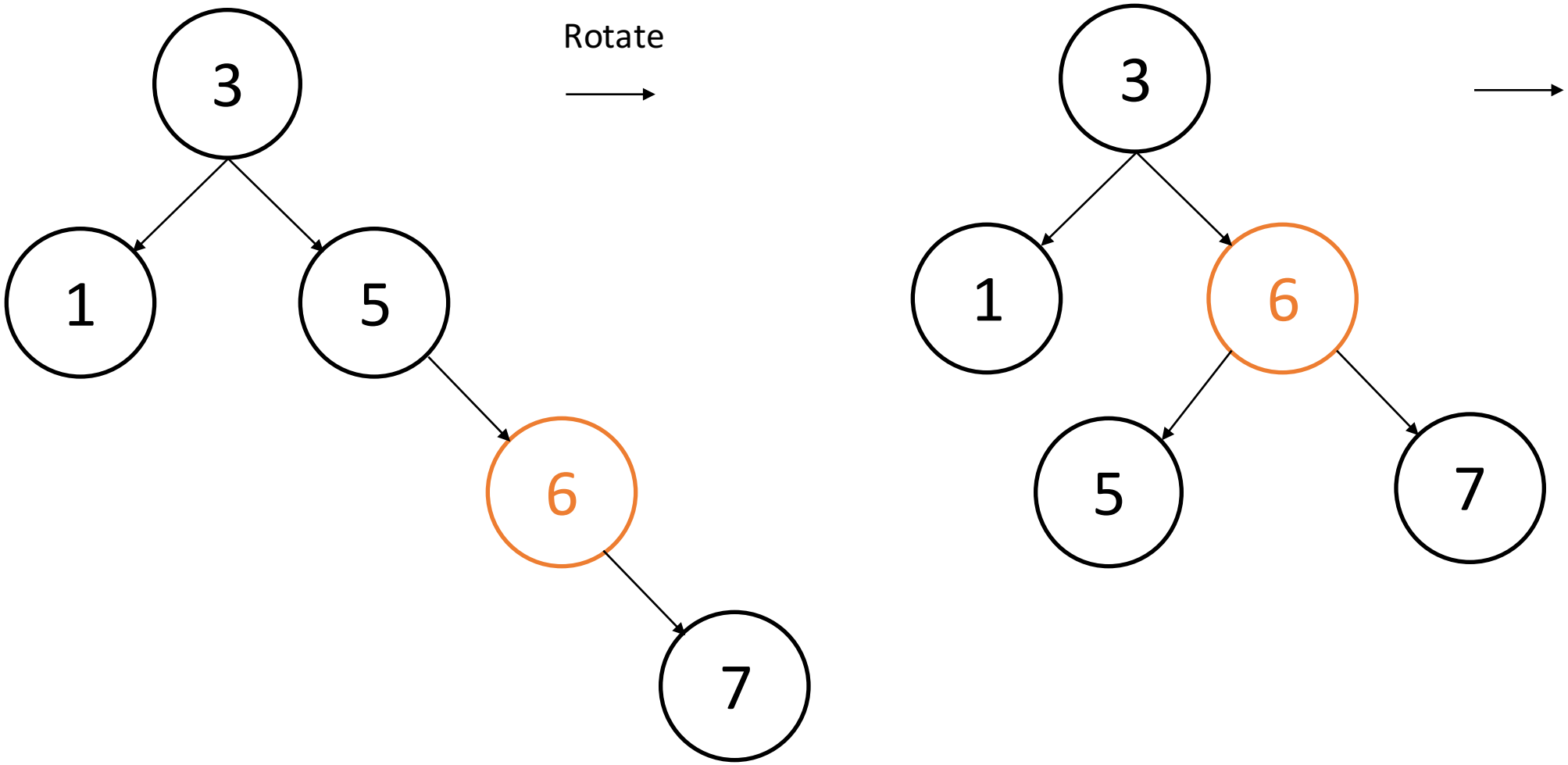
Q1.



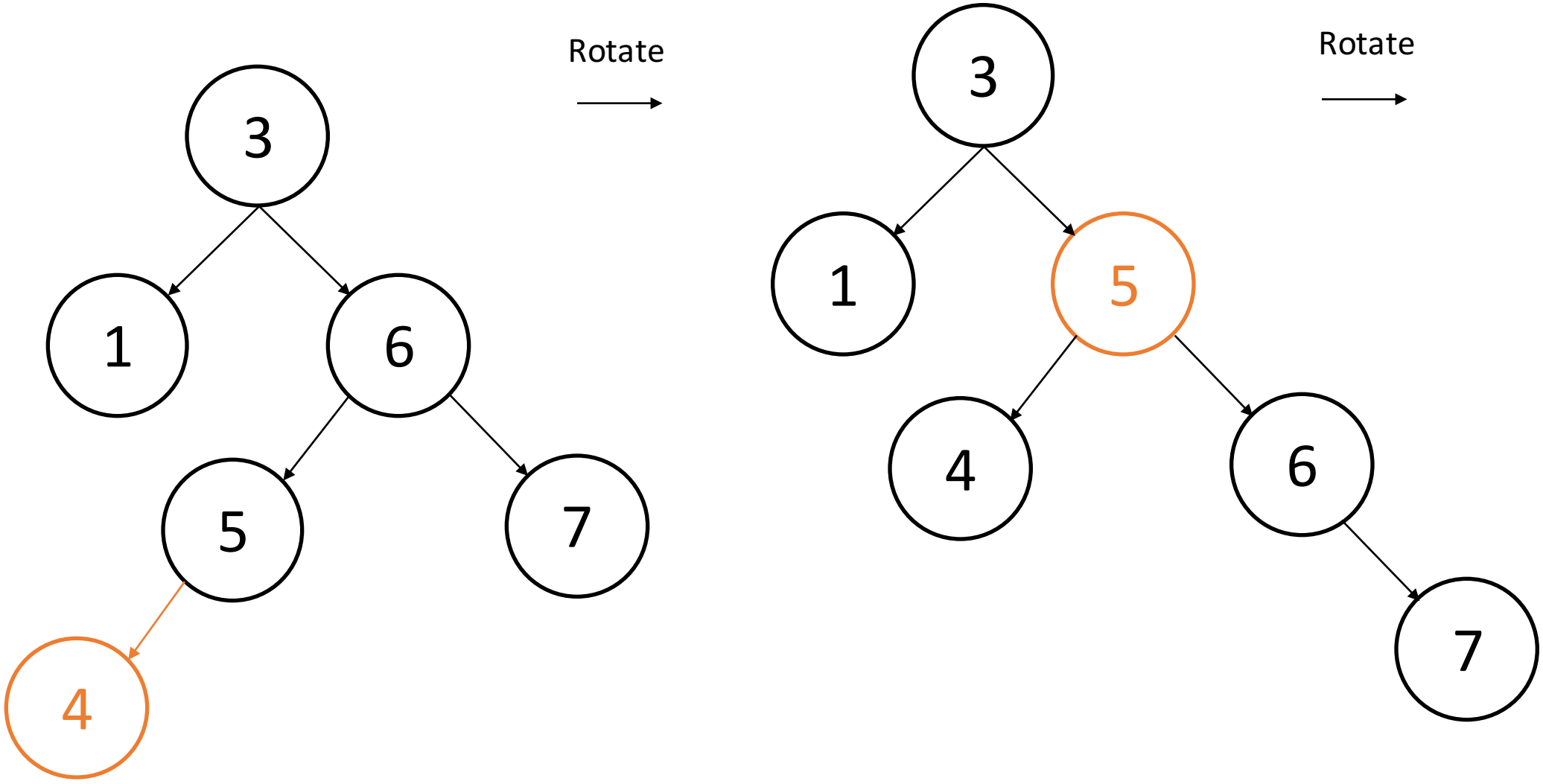
Q1.



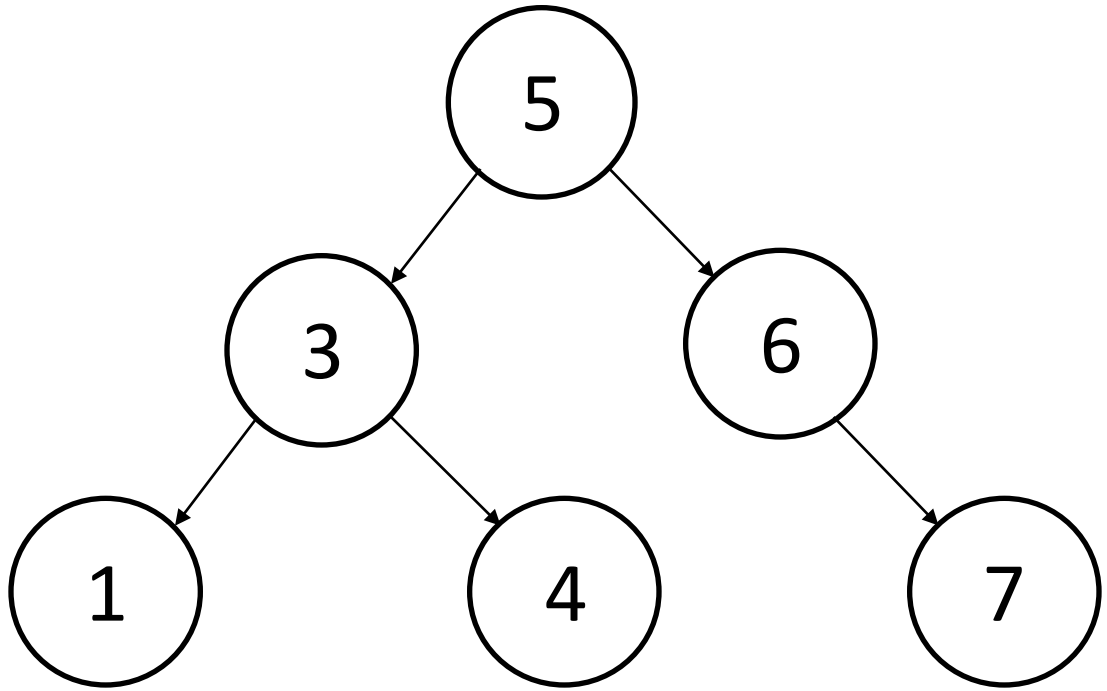
Q1.



Q1.



Q1.



Q2.

- Construct an example for which interpolation search will use $\Omega(n)$ comparisons for searching in a table of size n. The size n of your example should be larger than 5.

Q2.

A = [1, 2, 3, 4, 5, 100000] → Find 5

Cuz in Int_Find (z, Left, Right): $Next_{Guess} = \left\lceil L + \frac{(z - X[L])(R - L)}{X[R] - X[L]} \right\rceil$

$$Guess1 = \left\lceil 1 + \frac{(5-1)(6-1)}{100000-1} \right\rceil = 2$$

$$Guess2 = \left\lceil 2 + \frac{(5-2)(6-2)}{100000-2} \right\rceil = 3 \qquad and\ so\ on\ \ldots$$

因分母太大，Guess 每次只前進一格，因此至少要做n-1次，Thus, $\Omega(n)$.

Q3.

- Given an array of integers $A[1..n]$, such that, for all i, $1 \leq i \leq n$, we have $|A[i] - A[i+1]| \leq 1$. Let $A[1] = x$ and $A[n] = y$, such that $x < y$. Design an efficient search algorithm to find j such that $A[j] = z$ for a given value z, $x \leq z \leq y$. What is the maximal number of comparisons to z that your algorithm takes.

Q3.

```
Algorithm Find_j(A, n, x, y, z)
begin
        Left := 1
        Right := n
        while (Left != Right) :
                Middle := ⌈(Left + Right)/2⌉
                if z < A[Middle] then Right := Middle – 1
                else if z > A[Middle] then Left := Middle +1
                else return Middle
        if A[Left] = z then return Left
        else return 0
end
```

Q4.

- The input is a set S containing n real numbers, and a real number x. The problem is to determine whether there are two elements of S whose sum is exactly x. Suppose now that the set S is given in a sorted order. Design an algorithm to solve this problem in time $O(n)$.

Q4.

Algorithm Find_Sum_Exist(S, n, x):
begin
    startPoint := 1
    endPoint := n
    while startPoint < endPoint :
        if S[startPoint] + S[endPoint] = x then **return** True
        else if S[startPoint] + S[endPoint] < x **then**
            startPoint := startPoint + 1
        else then endPoint := endPoint - 1
    return False
end

Q5.

- Given two sets S_1 and S_2 , and a real number x, find whether there exists an element from S_1 and an element from S_2 whose sum is exactly x. The algorithm should run in time $O(n \log n)$, where n is the total number elements in both sets.

Q5.

```
Algorithm Find_Sum_In_Two_Arrays(S1, S2, n1, n2, x)
begin
        S1_AVLTree := BuildAVLTree(S1)                  O(nlogn)
        for i := 1 to n2:                               n次
                difference := x – S2[i]
                canFind := Find(S1_AVLTree, difference) O(logn)
                if canFind:
                        return True
        return False
end
```

Thus, $O(nlogn)$