Homework 2

蘇俊杰、劉韋成、曾守瑜

Question1

Consider again the inductive definition in HW#1 for the set of all binary trees that store non-negative integer key values:

(a) The empty tree, denoted $\perp$, is a binary tree.

(b) If t_l and t_r are binary trees, then $node(k, t_l, t_r)$, where $k \in Z$ and $k \geq 0$, is also a binary tree.

Refine the definition to include only binary *search* trees where an inorder traversal of a binary search tree produces a list of all stored key values in *increasing* order. Then, define inductively a function that outputs the rank of a given key value (the position of the key value in the aforementioned sorted list, starting from position 1) if it is stored in the tree, or 0 if the key is not in the tree.

Question1

1. The empty tree, denoted $\perp$, is a binary **search** tree.
2. If t_l and t_r are binary search tree,
   every key value (of descendants) in the nodes of t_l is smaller than k, and
   every key value (of descendants) in the nodes of t_r is larger than k,
   then $node(k, t_l, t_r)$, where $k \in Z$ and $k \geq 0$, is also a binary search tree.

Question1

t is a BST and n is the given key value.

$$Rank(t,n) = \begin{cases} 0, \neg Exist(t,n) \\ Rank'(t,n), otherwise \end{cases}$$

$$Exist(t,n) = \begin{cases} false, t = \bot \\ true, t = node(n, t_l, t_r) \\ Exist(t_l, n), t = node(k, t_l, t_r) \text{ and } n < k \\ Exist(t_r, n), t = node(k, t_l, t_r) \text{ and } n > k \end{cases}$$

$$Rank'(node(k, t_l, t_r), n) = \begin{cases} Rank'(t_l, n), n < k \\ Count(t_l) + 1, n = k \\ Count(t_l) + 1 + Rank'(t_r, n), n > k \end{cases}$$

$$Count(t) = \begin{cases} 0, t = \bot \\ Count(t_l) + 1 + Count(t_r), t = node(k, t_l, t_r) \end{cases}$$

Question1

很多人抄這個

$$Rank(t, n) = Rank'(t, n, 0)$$

$$Rank'(t, n, x) = \begin{cases} 0, t = \perp \\ Rank'(t_l, n, x), t = node(k, t_l, t_r) \text{ and } n < k \\ x + Count(t_l) + 1, t = node(k, t_l, t_r) \text{ and } n = k \\ Rank'(t_r, n, x + Count(t_l) + 1), otherwise \end{cases}$$

$$Count(t) = \begin{cases} 0, t = \perp \\ Count(t_l) + 1 + Count(t_r), t = node(k, t_l, t_r) \end{cases}$$

Question2

Consider the following recurrence relation:

$$\begin{cases} T(0) = 0 \\ T(1) = 1 \\ T(h) = T(h-1) + T(h-2) + 1, \quad h \geq 2 \end{cases}$$

Prove by induction the relation $T(h) = F(h+2) - 1$, where $F(n)$ is the n-th Fibonacci number ($F(1) = 1$, $F(2) = 1$, and $F(n) = F(n-1) + F(n-2)$, for $n \geq 3$).

Question2

[Base Case] (h=0) $T(0) = 0 = 1 - 1 = F(0+2) - 1$
(h=1) $T(1) = 1 = 2 - 1 = F(1+2) - 1$

[Induction Step]
$$
\begin{aligned}
T(h) &= T(h-1) + T(h-2) + 1 \\
&= (F(h+1) - 1) + (F(h) - 1) + 1 \\
&= F(h+1) + F(h) - 1 \\
&= F(h+2) - 1
\end{aligned}
$$

Question3

(2.30) A **full binary tree** is defined inductively as follows. A full binary tree of height 0 consists of 1 node which is the root. A full binary tree of height $h + 1$ consists of two full binary trees of height h whose roots are connected to a new root. Let T be a full binary tree of height h. The **height** of a node in T is h minus the node's distance from the root (e.g., the root has height h, whereas a leaf has height 0). Prove that the sum of the heights of all the nodes in T is $2^{h+1} - h - 2$.

Question3

[Base Case] height=0

[Induction Hypothesis] height=h+1

$$(2 * \textit{Sum of the height in } T) + \textit{height of root}$$
$$= 2 * (2^{h+1} - h - 2) \ + \ (h+1)$$
$$= 2^{h+2} - 2h - 4 + h + 1$$
$$= 2^{(h+1)+1} - (h+1) - 2$$

Question4

(2.23) The **lattice** points in the plane are the points with integer coordinates. Let P be a polygon that does not cross itself (such a polygon is called **simple**) such that all of its vertices are lattice points (see Figure 1). Let p be the number of lattice points that are on the boundary of the polygon (including its vertices), and let q be the number of lattice points that are inside the polygon. Prove that the area of polygon is $\frac{p}{2} + q - 1$.

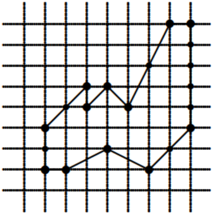


Figure 1: A simple polygon on the lattice points.

Question4

[Base Case] p=3, q=0

Area=$\frac{3}{2}$+0-1=$\frac{1}{2}$=$\frac{p}{2}$+q-1

[Induction Step] p=3, q>0

We can find a point q_1 in this triangle.
Connect this q_1 to the three vertices.
Split the large triangle to three small triangle.
Assume that there are q_d nodes on the line.
Induction Hypothesis :
The area of each small triangle is $\frac{p}{2}$+q-1.

Area=$\frac{9+2q_d}{2}$+(q-1-q_d)-3 (三個三角形)

Question4(Continue)

[Induction Step] p>3, $q \geq 0$

We can split the shape to a triangle and a polygon.
Assume that there are dq_d nodes on the line.

Area$=\frac{p+2q_d}{2}+$(q-1-q_d)$=\frac{p}{2}+$q-1

By M.I., we can prove that the area of the polygon is $\frac{p}{2}+$q-1

Question5

Consider the following pseudocode that represents the selection sort. The elements of an array of size n are indexed from 1 through n. Function *indexofLargest* gives the index of the largest element of the input array within the specified range of indices.

Algorithm selectionSort(A, n);
begin
    // the number of elements in A equals $n > 0$
    $last := n$;
    while $last > 1$ **do**
        $m := indexofLargest(A, 1, last)$;
        $A[m], A[last] := A[last], A[m]$;   // swap
        $last := last - 1$;
    od;
end

State a suitable loop invariant for the main loop and prove its correctness.

Question5

Selection sort
當原本的陣列 A，在迴圈中發生一次改變，變成 A'_1
更仔細說，$A[1]$ 到 $A[n]$ 當中最大的值被放到 n 號位，變成 A'_1

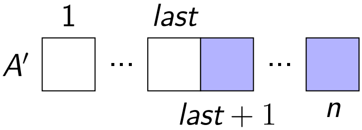
A	9	4	8	7	$last = n$
A'_1	7	4	8	9	$last = n - 1$

再從 $A'_1[1]$ 到 $A'_1[n-1]$ 當中挑最大的值放到 $n-1$ 號位，變成 A'_2

A'_2	7	4	8	9	$last = n - 2$

設 k 為迴圈執行過的圈數，則迴圈執行 k 次後，$last = n - k$
藍底色的部份是已經排序完畢的元素，它一定會比左邊所有元素都大

Question5



對 $A'[last+1]$ 而言，它的值一定比 1 到 *last* 位置的元素都來得大，也就是

$$indexofLargest(A', 1, last+1) = last+1$$

而在迴圈的上一個 iteration 就已經排好的 $A'[last+2]$，它的值一定比 1 到 $last+1$ 位置的元素來得大，也就是

$$indexofLargest(A', 1, last+2) = last+2$$

以此類推，能夠統整出一條規則

$$\forall\ last+1 \leq i \leq n.\ indexofLargest(A', 1, i) = i$$

Question5

迴圈不變量

$$Inv(last, A, n) =$$
$$(1 \le last \le n) \ \wedge \ (\forall \ last + 1 \le i \le n.\ indexofLargest(A', 1, i) = i)$$

給定一個陣列 A 與其長度 n，迴圈開始前 $last = n$
迴圈執行 1 步後，陣列內容發生改變，在數學上會將它視為另一個陣列，這裡用 A'_1 表示

$$Inv(n, A, n) \to Inv(n - 1, A'_1, n)$$

如果左邊的不變量是對的，右邊也會是對的
概念上就是把「確定已經排好」的範圍往左擴大一格

$$Inv(n - k, A'_k, n) \to Inv(n - (k + 1), A'_{k+1}, n)$$

Question5

Proof
Base case: $k = 0$, $last = n - 0 = n$,
"$Inv(n, A, n) = (1 \leq n \leq n) \;\wedge\; (\forall\ n + 1 \leq i \leq n.\ indexofLargest(A', 1, i) = i)$" is automatically true.

Question5

Induction: $n-1 \geq k > 1$, $last = n-k$,
From the inductive hypothesis of $last = n-k+1$, we get "$\forall\ n-k+2 \leq i \leq n.\ indexofLargest(A', 1, i) = i$" and on the k-th iteration of the loop, we pick the largest element between $A'[1]$ and $A'[n-k+1]$ to put in the $n-k+1$-th position.
So, with the new status of the array, say A'', we can say more about A'' than A': $A''[n-k+1]$ is larger than any element on its left side.
That is, $indexofLargest(A'', 1, n-k+1) = n-k+1$.
Therefore, $\forall\ last+1 \leq i \leq n.\ indexofLargest(A'', 1, i) = i$ is satisfied with $last = n-k$. □