Home Work 1
Com S 311

Due: Jan 24, 11:59PM

Late Submission Due: Jan 25, 11:59PM (25% penalty)

Outcomes. Be comfortable with proof techniques and use them to prove program correctness. There are 4 problems and each problem is worth 50 points.

1. *Fibonacci numbers* are defined recursively as follows:

$$\begin{aligned} F_0 &= 0 \\ F_1 &= 1 \\ F_n &= F_{n-1} + F_{n-2} \end{aligned}$$

   Show the following property of Fibonacci numbers by induction.

   For every $n \geq 1$,

$$F_1^2 + F_2^2 + F_3^2 + \cdots + F_n^2 = F_n \times F_{n+1}.$$

   Your proof must use mathematical induction; otherwise you will receive zero credit.

   Ans. **Base Case:** $n = 1$. $F_1 = 1$ and $F_2 = 1$.

$$F_1^2 = F_1 \times F_2 = 1.$$

   The claim is true when $n = 1$.

   Induction Hypothesis: Assume

$$F_1^2 + F_2^2 + \cdots + F_m^2 = F_m \times F_{m+1}$$

   Induction Step: We have to show

$$F_1^2 + F_2^2 + \cdots + F_m^2 + F_{m+1}^2 = F_{m+1} \times F_{m+2}$$

   By definition of Fibonacci numbers, $F_{m+2} = F_{m+1} + F_m$. Thus

$$F_{m+1}^2 = F_{m+1}F_{m+2} - F_{m+1}F_m$$

$$
\begin{aligned}
\Sigma_{i=1}^{m+1} F_i^2 &= F_{m+1}^2 + \Sigma_{i=1}^{m} F_i^2 \\
&= F_{m+1}^2 + F_m F_{m+1} (\text{ By induction hypothesis}) \\
&= (F_{m+1}F_{m+2} - F_{m+1}F_m) + F_m F_{m+1} \\
&= F_{m+1}F_{m+2}
\end{aligned}
$$

Thus the claim is true for every $n \in \mathbb{N}$.

2. Refer to the definition of Full Binary Tree from the notes. For a Full Binary Tree T, we use $n(T), h(T)$, and $i(T)$ to refer to number of nodes, height, and number of internal nodes (non-leaf nodes) respectively. Note that the height of a tree with single node is 1 (not zero). Using structural induction, prove the following:

   (a) For every Full Binary Tree T, $n(T) \geq 2h(T) - 1$.

   Ans.

   Base Case: For a tree with one node, $n(T) = 1$ and $h(T) = 1$ and the claim holds.

   IH: Assume that A and B two FBT's such that $n(A) \geq 2h(A) - 1$ and $n(B) \geq 2h(B) - 1$. Let T be a tree with r as root, A as left subtree and B as right subtree. Note that $n(T) = n(A) + n(B) + 1$, and $h(T) = \max\{h(A), h(B)\} + 1$. Consider the case that $h(A) \geq h(B)$ thus $h(T) = h(A) + 1$.

$$
\begin{aligned}
n(T) &= n(A) + n(B) + 1 \\
&\geq 2h(A) - 1 + 2h(B) - 1 + 1 \quad (\mathit{By\ IH}) \\
&\geq 2h(A) + 1 (\text{as } h(B) \geq 1) \\
&= 2(h(A)) + 2 + 1 - 2 \\
&= 2(h(A) + 1) - 1 \\
&= 2h(T) - 1
\end{aligned}
$$

   Proof for the case where $h(B) > h(A)$ is similar.

   (b) For every Full Binary Tree T, $i(T) = (n(T) - 1)/2$

   Your proof must use structural induction; otherwise you will receive zero credit.

   Ans. Solution not provided. Please see a TA/instructor.

3. Consider the following program, where a and n are positive integers.

```
Input: a, n

x = a; m = n; y = 1;
while (m > 1) {
   if m is even
        x = x*x;
        m = m/2;
   if m is odd
```

```
        y = x*y;
        x = x*x;
        m = (m-1)/2;
}

Output x*y
```

Let x_i, y_i, and m_i denote the value of the variables x, y, and m at the start of the ith iteration. Using induction show the following

$$\forall i \quad a^n = x_i^{m_i} \times y_i$$

Your proof must use induction. Otherwise you will not receive any credit.

Ans.

Base Case: Before the first iteration of the loop: We have $x_1 = a, y_1 = 1$ and $m_1 = n$. Thus $x_1^{m_1} \times y_1 = a^m$.

Induction Hypothesis: Assume that $x_k^{m_k} \times y_k = a^m$.

We will show that $x_{k+1}^{m_{k+1}} \times y_{k+1} = a^m$. Let us consider how x_k, y_k and m_k change during the kth iteration. We will consider two cases.

Case1: m_k is even; thus $m_k = 2\ell$ for some integer ℓ. In this case x become $x \times x$ and m become $m/2$ and y is unchanged. Thus $x_{k+1} = x_k^2$ and $m_{k+1} = m_k/2 = \ell$, and $y_{k+1} = y_k$. Thus

$$\begin{aligned} x_{k+1}^{m_{k+1}} \times y_{k+1} &= (x_k^2)^\ell \times y_k \\ &= x_k^{2\ell} \times y_k \\ &= x_k^{m_k} \times y_k \\ &= a^n (\text{By Induction Hypothesis}) \end{aligned}$$

Case 2: m_k is odd; thus $m_k = 2\ell + 1$. In this case, during the kth iteration, y becomes $x \times y$, x becomes x^2, and m becomes $(m-1)/2$. Thus. $y_{k+1} = x_k y_k$, $x_{k+1} = x_k^2$, $m_{k+1} = \ell$. Thus

$$\begin{aligned} x_{k+1}^{m_{k+1}} \times y_{k+1} &= (x_k^2)^\ell \times x_k y_k \\ &= x_k^{2\ell} \times x_k y_k \\ &= x_k^{2\ell+1} \times y_k \\ &= x_k^{m_k} \times y_k \\ &= a^n (\text{By Induction Hypothesis}) \end{aligned}$$

4. Consider the following problem. Given a sorted array a of consisting of distinct integers and an integer T, determine if there exist two integers in the array (possibly the same integer) whose sum equals T. Consider the following algorithm:

```
Input: Array a, Integer T.
left = 0;
right = length of the array;
while (left <=right){

  x = a[left] + a[right];

  if (x==T)
        return true;
  if (x <  T)
     left++;
  if (x > T)
     right--;

}

return false;
```

Show that the above program is correct by proving the following:

At the start of the ith iteration the following conditions hold:

- $left \leq right$
- If there exist indices i and j such that $a[i] + a[j]$ equals T, then $left \leq i \leq j \leq right$.

Ans.

We will first prove the invariant. Let n denote the length of the array. Note that we quit the loop when $left > right$, thus $left \leq right$ always holds.

Base Case: $left = 1$ and $right = n$. Thus if there are two indices $i \leq j$ such that $a[i]+a[j] = T$, then $1 \leq i \leq j \leq n$.

Induction Hypothesis: Assume that the invariant holds at the beginning of the kth iteration of the loop.

We will show that the invariant holds at the beginning of the $(k + 1)$st iteration of the loop.

Let us consider what happens during the kth iteration of the loop. If $x == T$ holds, we quit the loop and there is no $(k + 1)$st iteration.

Consider two cases. Let $left_k$ and $right_k$ denote the value of $left$ and $right$ before the kth iteration of the loop. Define $left_{k+1}$ and $right_{k+1}$ similarly.

Case 1: $x < T$. Suppose that there exist $i \leq j$ such that $a[i] + a[j] = T$. By induction hypothesis, we have

$$left_k \leq i \leq j \leq right_k$$

Since $a[left_k] + a[right_k] < T$, we will argue that $left_k \neq i$. Suppose that $left_k = i$. Then the maximum value of $a[i] + a[j]$ is at most $[left_k] + a[right_k]$ since the array is sorted.

However, we know that $a[left_k] + a[right_k] < T$. Thus $left_k \neq i$. Combining this with the induction hypothesis, we have $left_k < i$. In this case $left$ becomes $left + 1$ and $right$ remains unchanged. Thus $left_{k+1} = left_k + 1$ and $right_{k+1} = right_k$. Since $left_k < i$, we conclude that $left_{k+1} = left_k + 1 \leq i$. Thus we have $left_{k+1} \leq i \leq j \leq right_{k+1}$.

Case 2: $x > T$. Suppose that there exist $i \leq j$ such that $a[i] + a[j] = T$. By induction hypothesis, we have

$$left_k \leq i \leq j \leq right_k$$

Since $a[left_k] + a[right_k] > T$, we will argue that $right_k \neq j$. Suppose that $right_k = j$. Then the minimum value of $a[i] + a[j]$ is at most $[left_k] + a[right_k]$ since the array is sorted. However, we know that $a[left_k] + a[right_k] > T$. Thus $right_k \neq j$. Combining this with the induction hypothesis, we have $right_k > j$. In this case $right$ becomes $right - 1$ and $left$ remains unchanged. Thus $right_{k+1} = right_k - 1$ and $left_{k+1} = left_k$. Since $right_k > j$, we conclude that $right_{k+1} = right_k - 1 \geq j$. Thus we have $left_{k+1} \leq i \leq j \leq right_{k+1}$.