CSCE 222-200, Discrete Structures for Computing, Honors
Fall 2021
Homework 4
Aakash Haran

Instructions:

- The exercises are from the textbook. MAKE SURE YOU HAVE THE CORRECT EDITION! You are encouraged to work extra problems to aid in your learning; remember, the solutions to the odd-numbered problems are in the back of the book.
- Each exercise is worth 5 points.
- Grading will be based on correctness, clarity, and whether your solution is of the appropriate length.
- Always justify your answers.
- Don't forget to acknowledge all sources of assistance on the cover sheet, and write up your solutions on your own.
- *Turn in your pdf file on Canvas by 3:00 PM, Wednesday, October 27.*

LaTeX hints: Read this .tex file for some explanations that are in the comments.

Math formulas are enclosed in $ signs, e.g., `$x + y = z$` becomes $x + y = z$.

Logical operators: $\neg, \wedge, \vee, \oplus, \rightarrow, \leftrightarrow$.

Here is a truth table using the "tabular" environment:

p	$\neg p$
T	F
F	T

**** Delete the instructions and the LaTeX hints in your solution. ****

Exercises for Section 5.2 (pp. 362–365):

4a:
P(18) is true as 2(7) + 1(4) = 18
P(19) is true as 1(7) + 3(4) = 19
P(20) is true as 0(7) + 5(4) = 20
P(21) is true as 3(7) + 0(4) = 21

4b: Assume P(18) to P(k) is true.

4c: Prove that P(k + 1) is true with the inductive hypothesis.

4d: k + 1 = (k - 6) + 7. We know we can use a 7 and that P(k - 6) is true through the inductive hypothesis.

4e: All cases from 18 to k prove case k + 1. This logic repeats to prove all cases where $n \geq 18$.

12: Base case:
$P(1)$ is true. $1 = 2^0$
Inductive Hypothesis:
Assume $P(1)$ to $P(k)$ is true.
Proving $P(k+1)$:
If $k+1$ is even then we know $\frac{k+1}{2}$ is an integer and that $P(\frac{k+1}{2})$ is a true statement as it's in the inductive hypothesis. If $k+1$ is odd then we can split it into k and 1. We know k is true, and that $1 = 2^0$. Hence, $P(k+1)$ is true. $P(n)$ is true for $n \geq 1$.

32: The proof is wrong because it assumes $P(k)$ uses 4 cent stamps or 3 cent stamps which may not necessarily be true.

Exercises for Section 5.3 (pp. 378–381):

12: Using strong induction.
Proving $P(1)$:
$P(1)$ says $0^2 = 0 \times 1$, which is a true statement.
Inductive hypothesis: Assume statements $P(1)$ through $P(k)$ are true.
Attempting to prove $P(k+1)$ which says $\sum_{i=1}^{k+1} f_i^2 = f_{k+1} f_{k+2}$
The left side can be rewritten to make $f_k f_{k+1} + f_{k+1}^2 = f_{k+1} f_{k+2}$
Factoring f_{k+1} from the left side, we get $f_{k+1}(f_k + f_{k+1}) = f_{k+1} f_{k+2}$

From the property of the fibonacci sequence we know that $f_k + f_{k+1}$ is just f_{k+2}. Hence $P(k+1)$ is true. Thus the statement $\forall n P(n)$ is true when $n \geq 1$.

24(c): If x and y are in the set, then $x + y$ and $x \times y$ are in the set.
Basis set: $S = (x, 1)$.

28(c):
Basis step:
$5|0 + 0$ is as true statement as 0 is divisible by 5.
Inductive Hypothesis:
We can assume $5|(a + b)$ is true for all $(a, b) \in S$
Inductive step:
New elements of the set are recursively defined as if $(a, b) \in S$ then $(a+2, b+3)$ and $(a+3, b+2) \in S$. For the recursive elements, to prove $5|a + 2 + b + 3$ we can split it into $5|(a + b) \wedge 5|(2 + 3)$. We know the left part is true from the inductive hypothesis and that the second part is true since 5 is divisible by 5. Hence, we proved the statement.

46: Proving that the property is true for the basis step:
A tree with one node, the root node fits the property as the number of leaves is 1, which is 1 more than the number of internal vertices which is 0.
Assuming the property is true for full trees T_1 and T_2. We can define a new full tree T_3 that is the combination of T_1 and T_2 with a new root joining the two existing roots. If T_1 had n_1 internal vertices and T_2 had n_2 interal vertices then the new tree has $n_1 + n_2 + 1$ internval vertices and the number of leaves is the addition of the existing leaves which is $n_1 + 1 + n_2 + 1$. This value is 1 more than the new number of internval vertices hence the property is true for all full trees.

Exercise Not From the Book:

(A): Prove that the recursive binary search algorithm on slide 69 of Set 7 of the lecture notes is correct. Use strong induction to show that $P(n)$ is true for all integers $n \geq 0$, where $P(n)$ is the statement "Binary-search(i, j, x), where $j - i = n$, returns 0 if x is not in $A[i..j]$ and returns the index of x if x is in $A[i..j]$."
Base case:
The recursive algorithm will correctly terminate when there is only one element and when there are only two elements. The first case will calculate the middle index as 0 and the second will calculate the middle index and adjust to find the target element whether it was the index 0 or 1.
Inductive step:
We can assume that for lengths arrays from 1 to k, the recursive binary search algorithm terminates correctly.
Proving the recursive step:

For a length k + 1, the array can be split into (k+1)/2 if k + 1 is even, and both cases terminate correctly from the inductive hypothesis. If it's not even then we have k and an array of size 1, both of which are cases that work from the inductive hypothesis.
Hence, the recursive binary search algorithm is correct.

Exercises for Section 5.4 (pp. 391–392):

10: Just give the algorithm here; the next exercise asks you to prove its correctness.

function FINDMAX(index, arr):
    if $index == len(arr) - 1$ **then**
        return $arr[index]$
    return $max(arr[index], FINDMAX(index + 1, arr))$

22:
For an array of length 1, $FINDMAX$ returns the element at index 0, hence the recursive function is correct for the base case.
Inductive step: we can assume the function works for an array of length n.
To prove that $FINDMAX(n + 1)$ returns the maximum of $n + 1$ elements, we know that $FINDMAX(n)$ will return the maximum of n elements from the inductive step. The final call of the function returns $max(FINDMAX(n), arr[n])$ which is the maximum element of the set. Hence, the recursive function is correct.

24:

function CALCPOWER(a, n):
    if $n == 1$ **then**
        return $a \times a$
    return $a \times a \times CALCPOWER(a, n - 1)$

42: ** YOUR ANSWER GOES HERE **

Exercise for Section 5.5 (p. 398):

6:
Lemma 1 state that every polygon with at least four sides has at least one interior diagonal.

function TRIANGULATE(polygon):

if polygon has more than 3 sides **then**

draw an interior diagonal to split polygon into two smaller polygons, a and b. TRIANGULATE(a), TRIANGULATE(b)