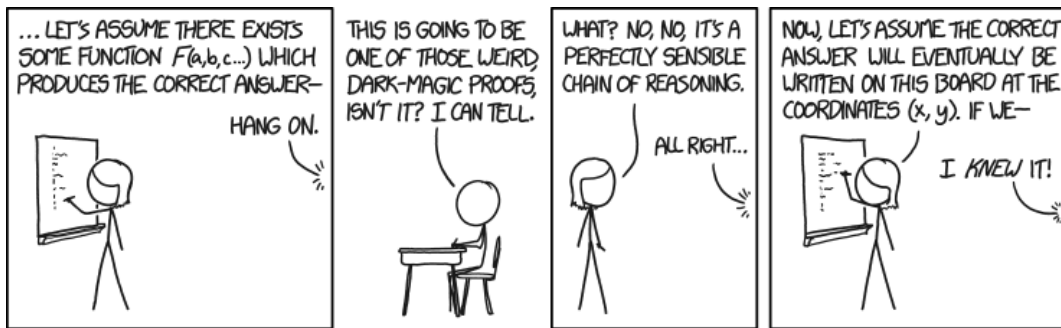


CS51 - Assignment 6

Proofs

Due: October 9th, at 11:59pm



<https://xkcd.com/1724/>

For this assignment, we will:

- gain more practice with Propositional Logic,
- learn about functional completeness,
- write more proofs using loop invariants, and
- gain more practice writing Math and code in $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$.

You are encouraged to work on pen and paper or a board first before you proceed with typing your response in the provided `assignment6.tex`. As always, please keep track of the time you have spent on the assignment and submit your feedback.

Before you come to lab

You are given the following loop:

```
while m>=0 and m<=100:
    m+=1
    n-=1
```

and a potential loop invariant: At the beginning and end of each iteration, $m+n=100$.

On paper, show the *maintenance* part of a loop invariant proof, that is, show that if the condition is true before entering the i -th iteration, then it is true before the $(i+1)$ -th iteration, after completing the work in the loop once.

1 Propositional Logic

p	q	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
		True	$p \vee q$	$p \Leftarrow q$	p	$p \Rightarrow q$	q	$p \Leftrightarrow q$	$p \wedge q$	$p \mid q$	$p \oplus q$	$\neg q$		$\neg p$		$p \Downarrow q$	False
T	T	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
T	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F

Figure 1: The truth tables for 16 different logical connectives.

Figure 1 shows the truth tables for 16 different logical connectives. We have seen many of these already in class: $\neg, \wedge, \vee, \Rightarrow, \Leftarrow, \oplus$. The (slightly) new ones are the Sheffer stroke ($\mid$), which corresponds to the logic gate NAND (the negation of AND), and Peirce's arrow ($\Downarrow$), which corresponds to the logic gate NOR (the negation of OR).

1.1 Part 1

Prove that the set of logical connectives $\{\vee, \wedge, \Rightarrow, \neg\}$ is *functionally complete* (also known as universal or adequately expressive). A set of logical operators is functionally complete if every logical connective can be expressed using some combination of the connectives in the set.

To do so, take each logical connective from Figure 1 and try to find a logically equivalent (results in the same truth/false values for the same truth assignment) proposition that only uses $\{\vee, \wedge, \Rightarrow, \neg\}$.

Hint 1: Which columns correspond to tautologies and unsatisfiable propositions? Which propositions have we seen in class that match these?

Hint 2: You might notice an interesting pattern between the columns 1–8 and 9–16 that you can take advantage of.

1.2 Part 2

Prove that the set $\{\vee, \wedge, \neg\}$ is functionally complete.

Hint: You already have proven this partially through Problem 1. Essentially, all you have to do is show that you can express $\Rightarrow$ using $\{\vee, \wedge, \neg\}$ only. Then, translate any logical equivalent proposition that you created in Problem 1 and that contains $\Rightarrow$ to now only contain $\{\vee, \wedge, \neg\}$.

2 Loop Invariants

Let's assume we want to find the largest number in a non-empty list `lst` and have written the following Python function:

```
def max(lst):
    answer = lst[0]
    for i in range(1, len(lst)):
        if (lst[i] > answer):
            answer = lst[i]
    return answer
```

We will prove that this function works as intended using loop invariants.

1. What is a pre-condition for this program?
2. What is a post-condition for this program?
3. State a loop invariant. Remember, this will be a condition that is true at the beginning and end of every iteration of a loop.
4. Write a loop invariant proof by filling out all the steps for *Initialization*, *Maintenance*, *Post-condition*, and *Termination*.

3 Weak Induction

Using mathematical induction, prove that for each integer $n \geq 1$, $\sum_{i=0}^n i(i+1) = \frac{n(n+1)(n+2)}{3}$.

4 Strong Induction

Leonardo Bonacci, commonly known today simply as Fibonacci, was a famous thirteenth-century Italian mathematician. He popularized the Indo-Arabic numeral system in the Western world and defined the Fibonacci numbers, a sequence that appears frequently in computer science.

We define the Fibonacci numbers by the sequence $f_1 = 1$, $f_2 = 1$, and $f_n = f_{n-1} + f_{n-2}$, for $n \geq 3$. Thus, the first several Fibonacci numbers are 1, 1, 2, 3, 5, 8, 13, 21, 34, 55,

Édouard Lucas was a nineteenth-century French mathematician who used the Fibonacci numbers to define the Lucas numbers. We define the Lucas numbers as $L_1 = 1$, $L_2 = 3$, and $L_n = L_{n-1} + L_{n-2}$, for $n \geq 3$.

Using **strong** induction and the definitions of Lucas and Fibonacci numbers, prove that $L_n = f_n + 2f_{n-1}$ for every integer $n \geq 2$.

5 Feedback

Create a file named `feedback6.txt` that answers the questions:

- How long did you spend on this assignment?
- Any comments or feedback? What things did you find interesting, challenging, or boring?

6 When you're done

Submit your `assignment6.tex`, resulting `assignment6.pdf`, and `feedback6.txt` files online using Gradescope; be sure you're uploading them to the right assignment. Note that you can resubmit the same assignment as many times as you would like up until the deadline.

7 Grading

Problem	points
Prelab work	1
Problem 1 - Part 1	2
Problem 1 - Part 2	1
Problem 2	2
Problem 3	2
Problem 4	2
Total	10