Constructive Logic (15-317), Fall 2020
Assignment 1: Say hi to logic!

Instructor: Karl Crary
TAs: Avery Cowan, Matthew McQuaid, Long Pham, Ariel Uy

Due: Friday, September 11, 2020, 11:59 pm

Welcome to 15-317, Fall 2020 edition! In this homework assignment, you will practice some basic principles you'll need for the rest of the course. As a special exception to the usual rules, for this homework assignment, you may collaborate with other students in the class on the answers to all of the questions as long as you do your write-up individually.

We recommend that you typeset your solutions using Latex. This is not required, but illegible proofs are not correct.

The assignments in this course must be submitted electronically through Autolab and Gradescope. Written homework PDFs will go to Gradescope, and code[1] will go to Autolab. Links to the course's Autolab page and Gradescope can be found on the course homepage. For this homework, submit two files: `hw1.pdf` (your written solutions) on Gradescope, and `hw1.tut` on Autolab.

One and one is one

Derive the following judgments using the inference rules given in lecture (be sure to name each rule when you use it).

Task 1 (1 point)**.**

$$(A \wedge (A \supset B)) \supset B \textit{ true}$$

Task 2 (2 points)**.**

$$(A \wedge ((A \wedge A) \supset B)) \supset B \textit{ true}$$

Task 3 (2 points)**.**

$$(A \wedge (A \supset B)) \supset (B \wedge B) \textit{ true}$$

Doing constructive mathematics

An integer a is said to *divide* an integer b, written $a \mid b$, if there exists an integer k such that $b = ak$. We write $a \nmid b$ for $\neg(a \mid b)$.

Task 4 (1 point)**.** Give an (informal) constructive proof of the following proposition: for all integers a, b, and c, if $a \mid b$ and $b \mid c$, then $a \mid c$.

Task 5 (1 point)**.** Give an (informal) non-constructive proof of the following proposition: for all integers a, b, and c, one of the following is true:

- $a \nmid b$ or $b \nmid c$;
- $a \mid c$.

Hint: Use the proposition from the previous task.

[1] either a single code file or a `tar` of multiple files, depending on assignment

Task 6 (1 point)**.** In the previous task, you likely used the following instance of the law of the excluded middle

$$a \mid b \vee a \nmid b.$$

Though constructivism rejects the law of the excluded middle as a general axiom, it will still be the case that for some specific propositions A we have

$$(A \vee \neg A) \textit{ true}.$$

In particular, this will be the case whenever A is decidable, that is, whenever there exists an effective procedure for deciding whether or not A is true. Give a brief and informal constructive justification for concluding $a \mid b$ when $\neg(a \nmid b)$.

The *Fundamental Theorem of Arithmetic* states that every integer greater than 1 either is prime or factors as the product of primes numbers, and moreover, that this factorisation is unique up to reordering of factors.

Task 7 (1 point)**.** Is the following proof that $3 \nmid 10$ constructive? Justify your answer.

> *Proof.* Assume to the contrary that $3 \mid 10$. Then there exists a k such that $10 = 3k$. By the fundamental theorem of arithmetic, k has some unique prime factorisation $k = \prod_{i=1}^{n} p_i$. So 10 factors into primes as $10 = 3\prod_{i=1}^{n} p_i$. But we also know that 10 factors into primes as $10 = 2 \times 5$. The existence of two distinct prime factorisations for 10 contradicts the uniqueness guaranteed by the fundamental theorem of arithmetic. We thus conclude that $3 \nmid 10$. □

Say hi to Tutch!

In this homework you will be introduced to the proof checker *Tutch*. If you were ever wondering whether using that inference rule was quite right or not, wonder no more! Tutch can check the correctness of your natural deduction proofs. You can use Tutch on the Andrew machines via the command:
`/afs/andrew/course/15/317/bin/tutch <file>`
Make sure in this case to keep anything that could be considered a solution in your `private` folder.

Task 8 (1 point)**.** Type the following proof in a file named `hw1.tut`, check it with `tutch` and submit it on Autolab:

```
proof andComm: A & B => B & A =
begin
  [ A & B;
    A;
    B;
    B & A ];
  A & B => B & A
end;
```

You can also run `tutch -r ./hw1.req hw1.tut` to compare your solution against the requirements file `hw1.req` provided with this assignment.