All 1511 students should take a shot at the first two problems. 2110 students, and CS 1511 students shooting for an A, should also take a shot at the last two problems.

13. (2 points) Express the following famous number theoretic conjectures about the integers using first order logic. You can use the arithmetic operations: addition, subtraction, multiplication, integer division (so no remainder), modulus, and exponentiation. You can also use standard comparators: less than, equal, ... You are welcome to search the web to find the statements of these conjectures if you don't know them (they are all on wikipedia). All quantifications should be universal or existential, and be over the integers (so you can't quantify over functions, sets, sequences ....). As a motivating example, here would be a reasonable answer for Fermat's Last Theorem, which is no longer a conjecture (`https://en.wikipedia.org/wiki/Fermat's_Last_Theorem`).

$$\neg \exists x \exists y \exists z \exists n \; x \geq 1 \wedge y \geq 1 \wedge z \geq 1 \wedge n \geq 3 \wedge x^n + y^n = z^n$$

   (a) Goldbach's conjecture
   (b) Erodös-Straus conjecture
   (c) Legendre's conjecture
   (d) Twin prime conjecture

14. (4 points) The purpose of this problem is to write (more or less) the parts of the Godel sentence for number theory that we discussed in lecture, for the following one tape Turing Machine M.

```
tape alphabet = {0, 1, space}
start state = q_0
states = Q = { q_0, q_1, q_h}
halt state = q_h. So the machine stops running if it ever reaches state q_h.
Transitions:
(q_0, 0) -> (q_0, 1, right)
(q_0, 1) -> (q_1, 0, right)
(q_0, space) -> (q_0, space, stay). So this is an infinite loop.
(q_1, 0) -> (q_1, 1, right)
(q_1, 1) -> (q_0, 0, right)
(q_1, space) -> (q_h, space, stay)
```

   (a) Describe the input strings I on which M halts.
   (b) Show one valid computation history H for the input $I = 10110$ for Turing machine M. Note that the computation history will have the form $\#C_0\#C_1\#C_2\#...\#C_k\#$ where k is the number of steps that M runs on I and C_i is a configuration of M on I after i steps. Note that there technically there is more than one valid computation history depending on how many spaces at the right of the tape are included in each configuration.

(c) Explain how to interpret H as an number written in octal (base 8).

(d) Give a number theoretic statement written in first order logic that expresses that the first symbol in H is a # symbol. Hint: Use the macro $PLACE(H, k) = H \bmod 8^{k+1}/8^k$, were / is integer division that excludes the remainder.

(e) Give a number theoretic statement written in first order logic that expresses that the symbol q_h appears between the last two # symbols in H.

(f) Give a number theoretic statement written in first order logic that expresses that the symbols q_a and q_1 do not appear between the last two # symbols in H.

(g) Give a number theoretic statement written in first order logic that expresses that the initial configuration is $q_0 10110$.

(h) Give a number theoretic statement written in first order logic that expresses that the initial configuration starts with $q_0 10110$ followed by an arbitrary number of spaces.

15. (6 points) The purpose of this problem is to write part of the Godel sentence for number theory that we did not discuss in lecture, namely that each configure logical follows from the prior configuration, for the following one tape Turing Machine M.

```
tape alphabet = {0, 1, space}
start state = q_0
states = Q = { q_0, q_1, q_h}
halt state = q_h. So the machine stops running if it ever reaches state q_h.
Transitions:
(q_0, 0) -> (q_0, 1, right)
(q_0, 1) -> (q_1, 0, right)
(q_0, space) -> (q_0, space, stay). So this is an infinite loop.
(q_1, 0) -> (q_1, 1, right)
(q_1, 1) -> (q_0, 0, right)
(q_1, space) -> (q_h, space, stay)
```

(a) Give a number theoretic statement written in first order logic that expresses that the fact that exactly one of the symbols q_0, q_1 and q_h appears between any two consecutive # symbols in H.

(b) Give a number theoretic statement written in first order logic that expresses that the fact that if H contains a substring S of the form $\#\alpha q_1 1 \beta\#$ where α and β are strings that do not contain the # symbols, then the string $\alpha q_0 0 \beta\#$ follows S in H. So this is saying that the transition in H is correct when the finite state is q_1 and the read/write head is pointing to a 1.

You are welcome to use macros, but of course you have to define whatever macros you use.

16. (8 points) Consider first order logical sentences of arithmetic where you are only allowed to use the arithmetic operators = and +. Without loss of generality we may assume that the only logical operators are AND, represented by $\wedge$ and NOT, represented by $\neg$, and that all quantifiers appear first. So you might have a formula like:

$$\forall x \exists y \forall z \exists w \ (x + y = z) \wedge \neg(y + z = w + x)$$

Our goal here is to show that there is an algorithm to accept exactly the language of true formulas. So the proof Godel's Incompleteness Theorem doesn't doesn't work if addition is the only allowed mathematical operation. We will use the following strategy.

(a) Show that the following language is regular (decided by a finite automaton). The language is a string of the following 8 symbols (so each line is a different symbol)

```
[000]
[001]
[010]
[011]
[100]
[101]
[110]
[111]
```

Note the pattern here is that the third bit is equal to the sum of the first two bits modulo 2. So the 5 symbol input

$$[101][001][110][011][011]$$

is in the language because $10100 + 00111 = 11011$. Note that in this 5 symbol encoding of this summation equation, the k^{th} symbol is determined by the k^{th} bit in each of the three numbers in the equation.

(b) Explain how to construct a finite state machine that accepts tuples (x, y, z, w) properly encoded that satisfy $\neg(y + z = w + x)$ from finite state machine.

(c) Explain how to construct a finite state machine that accepts tuples (x, y, z, w) properly encoded that satisfy $(x + y = z) \wedge \neg(y + z = w + x)$.

(d) Explain how to construct a finite state machine that accepts tuples (x, y, z) properly encoded with the property that $\exists w \ (x + y = z) \wedge \neg(y + z = w + x)$.

(e) Explain how to construct a finite state machine that accepts tuples (x, y) properly encoded with the property that $\forall z \exists w \ (x + y = z) \wedge \neg(y + z = w + x)$.

Note that the same idea can be used to construct a finite state machine that accepts strings x properly encoded with the property that $\forall y \forall z \exists w \ (x + y = z) \wedge \neg(y + z = w + x)$.

(f) Give an algorithm that, given as input a finite automata S, can determine whether there is any string that S accepts.

(g) Give an algorithm that decides whether a given first order sentence in number theory, that only involves arithmetic operations $+$ and $=$ and has the quantifiers appearing first, is true. More or less all the main ideas are contained in the previous subproblems. You more or less just need a couple of sentences to put everything together.