HW05

Sam Ly

October 9, 2025

Total points: 21

Required Exercise 1 [5]

Andy Zhang 9/30/25, 11:36 AM

> " Hi Andy, this is also for ex 1 hw05. I really liked your proof, it was super slick and a lot more straight forward than what I came up with. I really like how you explicitly state how you are substituting the equations. But that is also where I would have some constructive criticism. The proof itself is fine, but a latex tip is to use the reference command in order to refer back to previous work in a semantically correct way. Otherwise, I thought it was a great proof! Another thing is that the long string of display style math could have been better suited with a inline math. That way, the sentences flow better visually. Lastly, the wording of the conclusion of the proof seems a bit odd to me. Maybe something like "Therefore, we see that ¡expression¿ must be divisible by m, and that ¡expression¿ is congruent to ¡expression¿." "

Sox 9/30/25, 9:56 PM

> " Hi Sox, this is for HW5 ex 1.
>
> I really liked your proof, and thought it was intuitive and easy to follow. Here are the four pieces of feedback: 1. The structure of proof made a lot of sense. Proving the smaller statements before combining them in the final conclusion sentence made a lot of sense. 2. I like how you included multiple base cases to make the proof more concrete. 3. Some of the math isn't in "complete sentences." For example, The middle block of display style math doesn't end with a period. This is nitpicky, but I didn't really see anything else. 4. Overall, I thought this was a really good proof, but there were just some little formatting things like the "4m" at the start of the proof not being inline math. "

Required Exercise 2 [3]

1. Prove that for all integers $n \geq 2$,

$$(A_1 \cup A_2 \cup ... \cup A_n)^c = A_1^c \cap A_2^c \cap ... \cap A_n^c$$

Proof. First, we define some useful notation for a union and intersection for a large series of sets $A_1, A_2, ..., A_3$:

$$\bigcup_{i=1}^{n} A_i = A_1 \cup A_2 \cup ... \cup A_n$$

$$\bigcap_{i=1}^{n} A_i = A_1 \cap A_2 \cap ... \cap A_n.$$

First, we use the standard De Morgan's Law as the base case $n = 2$, $(A_1 \cup A_2)^c = A_1^c \cap A_2^c$.

Then, as our inductive hypothesis, we assume that

$$\left(\bigcup_{i=1}^{n} A_i\right)^c = \bigcap_{i=1}^{n} A_i^c, \text{ for } n \leq k.$$

Then, we see that for $n+1$,

$$\left(\bigcup_{i=1}^{n+1} A_i\right)^c = \left(\bigcup_{i=1}^{n} A_i \cup A_{n+1}\right)^c$$
$$\left(\bigcup_{i=1}^{n+1} A_i\right)^c = \left(\bigcup_{i=1}^{n} A_i\right)^c \cap A_{n+1}^c.$$

We can use our inductive hypothesis to substitute $\left(\bigcup_{i=1}^{n} A_i\right)^c = \bigcap_{i=1}^{n} A_i^c$ to get,

$$\begin{aligned}\left(\bigcup_{i=1}^{n+1} A_i\right)^c &= \bigcap_{i=1}^{n} A_i^c \cap A_{n+1}^c. \\ &= A_1^c \cap A_2^c \cap ... \cap A_n^c \cap A_{n+1}^c.\end{aligned}$$

Therefore,

$$(A_1 \cup A_2 \cup ... \cup A_n)^c = A_1^c \cap A_2^c \cap ... \cap A_n^c$$

for any integer $n \geq 2$. □

2. Prove that for all integers $n \geq 2$,

$$(A_1 \cap A_2 \cap ... \cap A_n)^c = A_1^c \cup A_2^c \cup ... \cup A_n^c$$

Proof. First, we use the standard De Morgan's Law as the base case $n = 2$, $(A_1 \cap A_2)^c = A_1^c \cup A_2^c$. Then, as our inductive hypothesis, we assume that

$$\left(\bigcap_{i=1}^{n} A_i\right)^c = \bigcup_{i=1}^{n} A_i^c, \text{ for } n \leq k.$$

Then, we see that for $n+1$,

$$\left(\bigcap_{i=1}^{n+1} A_i\right)^c = \left(\bigcap_{i=1}^{n} A_i \cap A_{n+1}\right)^c$$
$$\left(\bigcap_{i=1}^{n+1} A_i\right)^c = \left(\bigcap_{i=1}^{n} A_i\right)^c \cup A_{n+1}^c.$$

We can use our inductive hypothesis to substitute $\left(\bigcap_{i=1}^{n} A_i\right)^c = \bigcup_{i=1}^{n} A_i^c$ to get,

$$\begin{aligned}\left(\bigcap_{i=1}^{n+1} A_i\right)^c &= \bigcup_{i=1}^{n} A_i^c \cup A_{n+1}^c. \\ &= A_1^c \cup A_2^c \cup ... \cup A_n^c \cup A_{n+1}^c.\end{aligned}$$

Therefore,

$$(A_1 \cap A_2 \cap ... \cap A_n)^c = A_1^c \cup A_2^c \cup ... \cup A_n^c$$

for any integer $n \geq 2$. □

Required Exercise 3 [2]

See below.

Choice Exercise 6 [5]

1. **My original guess:** We say
$$p(x) \equiv q(x) \pmod{r(x)}$$
if there exists an integer k such that $p(x) - q(x) = k \times r(x)$.

   Comparison: My original guess was wrong because the actual definition for this relation is that there exists a function $f(x)$ such that $p(x) - q(x) = f(x) \times r(x)$.

   My original guess is actually a special case of the actual definition, since f can be a constant function. However, my original guess is also wrong in a more subtle way because the actual definition of polynomial congruence extends to the real numbers, while my original guess only relates to the natural numbers.

2. [1] Recall that if we're given two (positive) integers n=n and m and their decimal representation, it's easy to check $n \equiv m \pmod{10}$: just make sure that their last digits match up. What is the analgous rule when looking at polynomials $\mod x$?

   First, we notice that decimal is a positional number system, and 10 is one of bases. This makes it easy to check for congruence under $\mod 10$ because for integer n, if the one's place of number is 0, then $n \equiv 0 \pmod{10}$. Analogously, polynomials can themselves be see as a "positional number system" where each degree term is a position. Thus, any time a polynomial $p(x)$ has no constant term (analogous to the one's place), $p(x) \equiv 0 \pmod{x}$. We see this intuitively when we write out examples of $p(x)$. For example,
$$p(x) = x^2 + 3x = x(x+3).$$

3. [1] Prove that $x^2 \equiv -1 \pmod{x^2+1}$.

   We see that $p(x) = x^2$ and $q(x) = -1$, thus
$$x^2 - (-1) = (x^2+1) \times f(x),$$
for some f.

   We see that $f(x) = 1$. Thus, $x^2 \equiv -1 \pmod{x^2+1}$.

4. [1] Notice that you can replace every x^2 with -1 when simplifying a polynomial $\mod x^2+1$. Use this to simplify $x^4 + x^2 + 1 \pmod{x^2+1}$.

Choice Exercise 7 [3]

1. Choose a programming language, determine the logical operators, write a truth table, and record the results here.

```python
print("a b | a OR b")
print("------------")

for a in [True, False]:
    for b in [True, False]:
        print(f"{'T' if a else 'F'} {'T' if b else 'F'} | {'T' if a or b else 'F'}")
# a b | a OR b
# ------------
# T T | T
# T F | T
# F T | T
# F F | F
```

2. We can have a truth table which has three columns recording values, P, Q, and R. This means that there are $2^{2^3} = 256$ different logical statements we can make up to logical equivalence. Write a program that outputs a logical statement for each different outcome.

```python
# This function assumes the order of our input -> [(F,F,F), (F,F,T), (F,T,F), ...(T,T,T)]
def func(desired_output: list[bool]) -> str:
    if len(desired_output) != 8:
        return ""

    out = "False" # quick and dirty

    for index, output in enumerate(desired_output):
        if not output:
            continue

        out += f" or ({'' if (index // 4) % 2 == 0 else '!'}P and {'' if (index // 2) % 2 == 0 else '!'}Q and {'' if index % 2 == 0 else '!'}R)"

    return out


print(func([True] * 8))
# False or (P and Q and R) or (P and Q and !R) or (P and !Q and R) or (P and !Q and !R) or (!P and Q and R) or (!P and Q and !R) or (!P and !Q and R) or (!P and !Q and !R)

print(func([False] * 8))
# False

print(func([n % 2 == 0 for n in range(8)]))
# False or (P and Q and R) or (P and !Q and R) or (!P and Q and R) or (!P and !Q and R)
```

Choice Exercise 8 [3]

I will use the $\neg$ for negation because the bot uses this notation.

1. $((a \Leftrightarrow b) \vee \neg(\neg a \Leftrightarrow a))$

   First, we notice that $\neg a \Leftrightarrow a$ is always false. Thus, $\neg(\neg a \Leftrightarrow a))$ is always true. Therefore, the entire statement is always true.

2. $(a \Leftrightarrow b) \vee \neg\neg(b \vee a)$

   First, we see that the double negation can be simplified to yield $(a \Leftrightarrow b) \vee (b \vee a)$. Then, we see that there are two possible cases:

   $a \not\Leftrightarrow b$ When this is the case, then $b \vee a$ must be true since one of the variables must be true.

   $a \Leftrightarrow b$ When both variables are false or true, this statement is true.

   Thus, there is no combination of a and b that makes the statement false.

3. $(a \Leftrightarrow a) \rightarrow (\neg a \Leftrightarrow \neg a)$.

   First, we notice that $\neg \Leftrightarrow \neg a$ is always true. This means that the overall implication is also alaways true.

Proofs Portfolio

MAT 3100W: Intro to Proofs

Sam Ly

October 6, 2025

1 Introduction

(Leave this blank for now. Here's an outline of course topics for your reference.)

2 Proof techniques

(Here we give examples of some proof techniques.)

2.1 Proof by Induction

As an example of Proof by Induction, we will prove the following.

Proposition 1. *Let F_n be the n-th Fibonacci number, where $F_0 = F_1 = 1$ and $F_n = F_{n-1} + F_{n-2}$. Prove that $F_n \leq 1.9^n$ for all $n \geq 1$.*

Proof. We begin by verifying the relation for small n to create our base cases:

$$n = 1, F_1 = 1 \leq 1.9^1 = 1.9$$

$$n = 2, F_2 = 2 \leq 1.9^2 = 3.61.$$

We then form our inductive hypothesis by assumimg $F_n \leq 1.9^n$ for $1 \leq n \leq k$.

$$F_{k+1} \leq 1.9^{k+1}$$

$$F_k + F_{k-1} \leq 1.9^{k+1}.$$

Using our inductive hypothesis, we see that $F_k \leq 1.9^k$ and $F_{k-1} \leq 1.9^{k-1}$.
So, $F_k + F_{k-1} \leq 1.9^k + 1.9^{k-1}$.
By refactoring $1.9^k + 1.9^{k-1}$, we get:

$$1.9^k + 1.9^{k-1} = 1.9(1.9^{k-1}) + 1.9^{k-1} = 2.9(1.9^{k-1}).$$

Also, 1.9^{k+1} can be rewritten as $1.9^2(1.9^{k-1}) = 3.61(1.9^{k-1})$.
Finally, we see

$$F_{k+1} = F_k + F_{k-1} \leq 2.9(1.9^{k-1}) \leq 3.61(1.9^{k-1}) = 1.9^{k+1}$$

$$F_{k+1} \leq 1.9^{k+1}.$$

Therefore, $F_n \leq 1.9^n$ for all $n \geq 1$. □

Appendix

(The first section, "Course objectives and student learning outcomes" is just here for your reference.)

A Course objectives and student learning outcomes

1. Students will learn to identify the logical structure of mathematical statements and apply appropriate strategies to prove those statements.
2. Students learn methods of proof including direct and indirect proofs (contrapositive, contradiction) and induction.
3. Students learn the basic structures of mathematics, including sets, functions, equivalence relations, and the basics of counting formulas.
4. Students will be able to prove multiply quantified statements.
5. Students will be exposed to well-known proofs, like the irrationality of $\sqrt{2}$ and the uncountability of the reals.

A.1 Expanded course description

- Propositional logic, truth tables, DeMorgan's Laws
- Sets, set operations, Venn diagrams, indexed collections of sets
- Conventions of writing proofs
- Proofs
  - Direct proofs
  - Contrapositive proofs
  - Proof by cases
  - Proof by contradiction
  - Existence and Uniqueness proofs
  - Proof by Induction
- Quantifiers
  - Proving universally and existentially quantified statements
  - Disproving universally and existentially quantified statements
  - Proving and disproving multiply quantified statements
- Number systems and basic mathematical concepts
  - The natural numbers and the integers, divisibility, and modular arithmetic
  - Counting: combinations and permutations, factorials
  - Rational numbers, the irrationality of $\sqrt{2}$
  - Real numbers, absolute value, and inequalities
- Relations and functions
  - Relations, equivalence relations
  - Functions
  - Injections, surjections, bijections

- Cardinality
  - Countable and uncountable sets
  - Countability of the rational numbers, $\mathbb{Q}$
  - Uncountability of the real numbers, $\mathbb{R}$