I have some concerns about proving that all of arithmetic can come from the Peano Axioms. Perhaps it is possible to define the functions mentioned below in more clever ways that I am unaware of, and my concerns are completely pointless. I also realize this could be construed as proving a universal claim by example, but instead, think of it as proving one curiosity by example. I think that is valid.

I would like to begin with the idea that an axiom is simply an allowed semantic idea, and a semantic idea is something that can be expressed with symbols. I will use the English language to express semantic ideas as best as I can, and mathematical notation to show how they can be conveyed. An example of this is the ninth axiom:

9. For every natural number n, if n is in K, then $S(n)$ is in K, and K contains every natural number.

The semantics of this statement allow us to begin using the $S(x)$ notation, or the +1 notation meaningfully. Now, the first eight axioms set us up to use the = sign, eliminate all negative numbers, allow the use of the number zero, and things like that. From these semantics, and these semantics *alone*, we can write a formula:

$$y = S(x)$$

This simply adds one to the value x and assigns it to the variable y. These semantics can be replicated with other symbols, and are not dependent on the notation system itself. However, I believe that many of the steps involved in extrapolating out the Peano axioms to the extent of arithmetic that it often is are merely features of our common notation system. Consider the decrementing function:

$$S(x) = y$$

It is commonly accepted that this is a valid way to subtract one, but is it? It uses the left side of the function, a construct we created outside of the axioms, in order to subtract one value. We could replicate our prior simple successor function by writing $S(x)$, or $x + 1$, or `X plus one`. We cannot replicate this $S(x) = y$ function in any way, using the semantics we were already given, outside of our personal notation system. Similarly, pattern matching is a feature of our notation system. Consider an example:

$$sign(x) = \begin{cases} 0 & 0 \\ S(x) & 1 \end{cases}$$

This idea of branching apart the input into multiple functions is an abuse of the notation system in order to derive a mathematical operation that the axioms do not allow. The semantics involved in this operation do not follow from the semantics given by the axioms, they follow from the symbols that we assigned to the semantics being re-given new semantics and then extrapolated upon. The thoughts that led me to these hypotheses are below.

Let's imagine a partial function. One such function, over $\mathbb{N}$, is the sign function. We define the function as such:

$$sign(x) = \begin{cases} 0 & 0 \\ S(x) & 1 \end{cases}$$

It is rather obvious that this function relies on pattern matching for its power. Consider the concept that you do not wish to write the pattern matching into your function, because maybe some student believes it is far too powerful for primitive recursion. One way to rewrite the function would be:

$$sign(x) = \frac{x}{x}$$

This takes advantage of the idea that undefined behavior can be used to our advantage. This function is undefined when the number is zero, and defined when the number is one. From this, we can define all sorts of functions that, instead of pattern matching, utilize intentional undefined behavior to express themselves as one function without pattern matching. Consider subtraction, which should be undefined when y is greater than x. We defined floored subtraction in class as the following.

$$-(x, y) = \begin{cases} x, 0 & x \\ S(x), y & DECR(-(x, y)) \end{cases}$$

Without pattern matching, expressed as:

$$-(x, y) = \frac{DECR(-(x, y))}{x} \cdot x$$

Without decimal values, though, this approach will fail on the division. In order to avoid this problem, we can first write a new function using our $sign(x)$ function, $gr(x)$ or $\geq (x)$.

$$ge(x) = sign(S(-(x, y)))$$

We may now consider a more succinct approach.

$$-(x, y) = DECR(-(x, y)) \cdot sign(x) \cdot sign(y)$$

But now $ge(x)$ and $-(x, y)$ are defined in terms of each other, and this function relies on the notion that you must stop recursing when you hit an undefined value, and then treat that undefined value as zero.

The above are merely thoughts I had while contemplating whether or not it would be possible to eliminate pattern matching and remain able to construct the same functions.

- Pattern matching truly adds power. It cannot be replicated in non-pattern-matched functions without changing the axioms to include treating undefined behavior as zero.

- Also, while unmentioned here, I believe altering the left side of the function with the successor function is adding power.

From these hypotheses (which I have not gone through the effort to prove, although maybe I should), it feels like using pattern matching and left-hand modification are simply abuses of our notation system being used to add power to the Peano Axioms.