MATH 999 – Assignment 9

Last-Name, First-Name
`email@domain.com`

Last-Name, First-Name
`email@domin.com`

October 24, 2015

2.IV. Provide some examples of proofs.

Here is a proof with a contradiction symbol.
Theorem. *If Q is both true and false then P is true.*

Proof. Suppose P is false.

However, Q is both true and false.

Therefore we have a contradiction and thus P must be true. $\bigtimes$

Notice that the theorem environment (the AMS standard approach to writing theorems) *thm* creates a large whitespace. Personally I don't like how this looks so I often actually just write **Theorem:** instead of using an environment.

2.XI. Write a proof with several lemmas.

Typically a person writes lemmas before getting to the proof itself in order to avoid confusion about what is being proven. However by using indentation it is possible to write a lemma inside the proof body (as a sort of nested subproof) without creating confusion. As an added bonus it eliminates any confusion inflicted on the reader when they are unable to figure out the reason for needing the lemma until getting to the proof itself.

The rule of thumb I use is that any lemma that stands on its own and any lemma that many proofs rely on should be written on its own outside the proof. Otherwise, any lemma that solely exists to support a specific proof should be written within the proof body (unless the lemma is huge or something). I find this akin to writing a program where some functions are defined inside the main functions and others are defined elsewhere. The analogy could be pushed further if one acknowledges the Curry-Howard correspondence.

First we prove lemma 1.

Lemma 1: Lemma proof statement.

Proof. Proof Body. **1.Write out this proof** $\square$

Next we prove lemma 2.

Lemma 1: Lemma proof statement.

Proof. Proof Body. **2.Write out this proof** $\square$

Now we can move onto our main result.

Theorem: Main result proof statement.

Proof. We need to show that a number of things are satisfied.

We can do this with lemmas.

First we prove sublemma 1.

> **Sublemma 1:** Sublemma proof statement.
>
> *Proof.* Proof body. **3.Write out this proof** ⨳

Next we prove sublemma 2.

> **Sublemma 2:** Sublemma proof statement.
>
> *Proof.* Proof body. **4.Write out this proof** $\square$

Finally, by taking all of our lemmas together with the reference [1] we can see the result is proven. $\square$

2.XII. Format a proof with cases.

Cases are also very difficult to handle without using indentation via the addmargin command. They are the reason I began using the *addmargin* environment.

Theorem. Proof statement.

Proof. We can break the problem into cases.

Case 1: $(x = 0)$

Suppose $x = 0$.

Do things.

Thus P is true.

Case 2: $(x = 1)$

Suppose $x = 1$.

Do things.

Thus P is true.

Case 3: $(x \neq 0$ and $x \neq 1)$

Suppose $x \neq 0$ and $x \neq 1$.

Do things.

Thus P is true.

Since P is true in each case then P must always be true. $\square$

2.XII. Format a proof by induction.

Proofs by induction can be difficult and tedious to typeset and as such often just get handwaved away. I haven't quite settled on a standard way of writing them but this style seems to work.

Proof. Proof Statement.

Basis: Suppose $n = 0$.

Do things.

Therefore the basis is proven.

Inductive step: Let $k \in \mathbb{N}$ be an arbitrary integer. Suppose the following statement

Inductive hypothesis: $f(k) \leq f(k+1)$.

in order to prove the following statement

Inductive claim: $f((k+1)) \leq f((k+1)+1)$.

Do things.

This proves the inductive claim and completes the inductive step.

The basis together with the inductive step prove our original claim by induction thus completing the proof. $\square$

3.II. Format an equation array.

Here are two different ways of writing an equation array (there exist many more).

The *IEEEeqnarray* version looks a little better and has slightly better functionality when it comes to line referencing and labels but it is more cumbersome to write than the *align* version.

Note: I'm using the *enumitem* package for the *enumerate* environment. It lets me manually define the numbering format and even give a crazy explicit label like this.

–a–> This is written using the align environment.

$$\begin{aligned}
4xyzw &= 2 \cdot 2tu \\
&\leq 2 \cdot (t^2 + u^2) && \text{(a remark in parentheses)} \\
&= 2 \cdot ((xy)^2 + (zw)^2) \\
&= 2 \cdot (x^2y^2 + z^2w^2) && \text{a remark without parentheses} \\
&= 2x^2y^2 + 2z^2w^2 \\
&\leq ((x^2)^2 + (y^2)^2) + ((z^2)^2) + (w^2)^2) \\
&= x^4 + y^4 + z^4 + w^4
\end{aligned}$$

–b–> This is the same math written using the IEEEeqnarray environment.

$$\begin{aligned}
4xyzw &= 2 \cdot 2tu \\
&\leq 2 \cdot (t^2 + u^2) && \text{(a remark in parentheses)} \\
&= 2 \cdot ((xy)^2 + (zw)^2) \\
&= 2 \cdot (x^2y^2 + z^2w^2) && \text{a remark without parentheses} \\
&= 2x^2y^2 + 2z^2w^2 \\
&\leq ((x^2)^2 + (y^2)^2) + ((z^2)^2) + (w^2)^2) \\
&= x^4 + y^4 + z^4 + w^4
\end{aligned}$$

3.II Bonus. Write a more complex equation.

Here is function with a case in it. It's possible to define these using cases but using the IEEEeqnarraybox is more flexible, looks pretty nice, and is really easy to copy and paste.

$$F(x_2) = \begin{cases} \#\mathscr{P}_{x_2} & \text{if } x_2 \in \text{Total}, \\ x_2 & \text{if } x_2 \notin \text{Total}. \end{cases}$$

Here is another function so you can see which parts change depending on the function.

$$F_n = \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-2} + F_{n-1} & \text{if } n \geq 2. \end{cases}$$

4.I. Draw some commutative diagrams.

Here are some examples of commutative diagrams. They're made using *tikz* with the tikzlibrary *cd*.

Simple: First a simple diagram is shown.

$$\begin{tikzcd} A \arrow[r] \arrow[dr] & B \arrow[d] \arrow[dr] & \\ & C \arrow[r] & D \end{tikzcd}$$

Complex: Next a more complex diagram is shown.

$$
\begin{tikzcd}
& f^*E_V \arrow[rr] \arrow[dl] \arrow[dd, shift left] \arrow[dd, shift right] & & E_V \arrow[dl] \arrow[dd] \\
f^*E \arrow[rr] \arrow[dd] & & E \arrow[dd] & \\
& U \arrow[rr, bend left] \arrow[rr, bend right] \arrow[dl] & & V \arrow[dl] \\
M \arrow[rr, "\varphi"] & & N &
\end{tikzcd}
$$

ToDo

P.

1. Write out this proof 2
2. Write out this proof 2
3. Write out this proof 2
4. Write out this proof 2

References

[1] Lynn Arthur Steen and J. Arthur Seebach Jr., *Counterexamples in Topology*, Vol. 18, Springer, 1978.

[2] Abhijit Dasgupta, *Set Theory: With an Introduction to Real Point Sets*, Springer Science & Business Media, 2014.

[3] Edmund Landau, *Foundations of Analysis*, Vol. 79, American Mathematical Soc., Providence, 1966.

[4] Patrick Suppes, *Axiomatic Set Theory*, Dover Publications, Inc., New York, 1972.

[5] Martin Davis, Ron Sigal, and Elaine J Weyuker, *Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science*, Academic Press, 1994.

[6] James R Munkres, *Topology: A First Course*, Vol. 23, Prentice-Hall Englewood Cliffs, NJ, 1975.

[7] MSKfdaswplwq (http://math.stackexchange.com/users/42887/mskfdaswplwq), *Show that the infinite intersection of nested non-empty closed subsets of a compact space is not empty*, Mathematics Stack Exchange. URL: `http://math.stackexchange.com/q/337397` (version: 2013-03-21).

[8] Steven R Lay, *Analysis: with an introduction to proof*, Pearson Education, 2014.

[9] Walter Rudin, *Principles of Mathematical Analysis*, Vol. 3, McGraw-Hill New York, 1964.

[10] H. Jerome Keisler, *Foundations of infinitesimal calculus*, Vol. 20, Prindle, Weber & Schmidt, Boston, 1976.

[11] Michael O'Connor, *An Introduction to Smooth Infinitesimal Analysis*, ArXiv (2008), available at `http://arxiv.org/abs/0805.3307`.

[12] Andrej Bauer, *Intuitionistic Mathematics and Realizability in the Physical World*, A computable universe: understanding and exploring nature as computation, World Scientific, 2013, pp. 143–158, available at `http://math.andrej.com/wp-content/uploads/2014/03/real-world-realizability.pdf`.