Types and Programming Languages (15-814)
Assignment Desiderata & Advice

Ryan Kavanagh

Fall 2018

This document describes what we are looking for when we ask you to write a proof or write a program in your assignments.

1 On Proofs

A proof is first and foremost a means of *communicating* a *mathematical argument*. This means we look for both *clear* communication and a *correct* argument, and your grade is based on both of these components.

There is significant interplay between clarity and correctness when writing a proof. A proof's correctness depends on each step of the deduction being justified. By writing clearly and explicitly justifying your steps, you force yourself to think carefully about your argument, thereby increasing the odds of giving a correct proof. For example, in an induction proof, if you justify a deduction simply "by the induction hypothesis" without specifying to what you applied the induction hypothesis, you risk applying it where you have no right to do so. Conversely, it is often easier to clearly communicate a correct proof than it is an incorrect proof. If you clearly understand your argument, then you can describe its every step. In contrast, if your argument has logical leaps or unjustified inferences, you will find it difficult to write it so that the reader understands every step.

We expect your proofs to be "abundantly clear", meaning "the proof cannot be made significantly more clear by being explicit or verbose" [1]. To help you write clear and correct proofs, we enumerate some desiderata below for you to keep in mind when writing your proofs. These lists are based in part on the rubrics from other courses [1, 4]. You may find it helpful to look at these to understand what others look for when grading proofs. You may also find it helpful to read general guides on how to write mathematics [2, 3] or the following primer on mathematical proofs, complete with a list of common mistakes in proof-writing [5].

1.1 Communication

1. In a proof by induction, you must explicitly state over what you are inducting.
2. In a proof by cases, you must explicitly state over what you are doing case analysis.
3. If you use an induction hypothesis, it must be abundantly clear what it is.
4. It must be abundantly clear what you must show in each case of a proof by induction or by case analysis.
5. If you deduce something by inversion, it must be clear to what you are applying inversion. State either the name of the rule or the conclusion to which you applied inversion.
6. If you use a result from class or a previous assignment, you must explicitly restate it.
7. If you apply a result, it must be abundantly clear that you can do so. For example, if you apply a theorem, it must be abundantly clear that its hypotheses hold.
8. You must clearly define any notation you introduce.
9. If you claim something holds without loss of generality, the justification for this claim must be abundantly clear.
10. More generally, the justification for each assertion must be abundantly clear.
11. Your writing must be clear. Your sentences should be grammatical and coherent. When pronouns are used, their antecedents should be abundantly clear.
12. If you reformulate the statement of the problem, it must be abundantly clear why the reformulation is equivalent to the original statement.
13. If you provide a counter-example, it must be abundantly clear why it is a genuine counter-example.

1.2 Correctness

For your proof to be correct, we expect you to prove the result that was asked of you and for each step in your proof to be justified. In practice, this means:

1. Each step of your proof must be justified and each inference must be logically sound.
2. The conclusion of your proof must be the desired result.
3. In a proof by induction or by cases, all cases must be considered.
4. If you apply a result, then you must be justified in doing so. For example, if you apply a theorem, its hypotheses must hold.
5. If you claim that a result follows by a symmetric argument, the argument required must actually be symmetric to one already given.
6. If you deduce something by inversion, you must be justified in doing so.

1.3 Elegance

A small portion of your grade reflects the elegance of your proof. Your proof should be as simple as possible while remaining correct. You should not introduce needless mathematical machinery or results.

1.4 Example proofs

We suggest following the proof styles found in the lecture notes or in the homework solutions. The style found in the lecture notes is particularly suited to proofs by induction or by cases over inductively defined judgments. Here is an example:

Theorem 1 (Preservation)**.** If $\cdot \vdash e : \tau$ and $e \mapsto e'$ then $\cdot \vdash e' : \tau$

Proof. By induction on the derivation of $e \mapsto e'$.

Case $(\mapsto/l)$ **with** $\tau = \tau_1 + \tau_2$**:** In this case, $e = l \cdot e_1$ and $e' = l \cdot e_1'$ for some e_1 and e_1'. We must show that $\cdot \vdash l \cdot e_1' : \tau_1 + \tau_2$.

1. $l \cdot e_1 \mapsto l \cdot e_1'$	Case rule conclusion
2. $e_1 \mapsto e_1'$	Case rule hypothesis
3. $\cdot \vdash l \cdot e_1 : \tau_1 + \tau_2$	Case assumption
4. $e_1 \mapsto e_1'$ and $\cdot \vdash e_1 : \tau_1$ implies $\cdot \vdash e_1' : \tau_1$	IH statement
5. $\cdot \vdash e_1 : \tau_1$	By inversion on 3, rule (I-+)
6. $\cdot \vdash e_1' : \tau_1$	By IH 4 on 2 and 5
7. $\cdot \vdash l \cdot e_1' : \tau_1 + \tau_2$	By rule (I-+) on 6

Case . . .: . . . □

This proof was typeset with the following LaTeX code:

```
By induction on the derivation of $e \step e'$.

\begin{description}
\item[Case $({\mapsto}/l)$ with $\tau = \tau_1 + \tau_2$:]
  In this case, $e = \inl{e_1}$ and $e' = \inl{e_1'}$ for some $e_1$
  and $e_1'$.
  We must show that $\cdot \vdash \inl{e_1'} : \tau_1 + \tau_2$.
  \begin{enumerate}
  \item $\inl{e_1} \step \inl{e_1'}$
    \hfill Case rule conclusion
  \item $e_1 \step e_1'$ \hfill Case rule hypothesis \label{item:3}
  \item $\cdot \vdash l \cdot e_1 : \tau_1 + \tau_2$
    \hfill Case assumption \label{item:1}
  \item $e_1 \step e_1'$ and $\cdot \vdash e_1 : \tau_1$
    implies $\cdot \vdash e_1' : \tau_1$
    \hfill IH statement \label{item:5}
  \item $\cdot \vdash e_1 : \tau_1$
    \hfill By inversion on \ref{item:1}, rule \rn{I-+} \label{item:4}
  \item $\cdot \vdash e_1' : \tau_1$
    \hfill By IH \ref{item:5} on \ref{item:3} and \ref{item:4}
    \label{item:2}
  \item $\cdot \vdash l \cdot e_1' : \tau_1 + \tau_2$
    \hfill By rule \rn{I-+} on \ref{item:2}
  \end{enumerate}
\item[Case \ldots:] \ldots\qedhere
\end{description}
```

You could alternatively write your proof in paragraph form. Here is an example:

Proof. By induction on the derivation of $e \mapsto e'$.

Case ($\mapsto / l$) with $\tau = \tau_1 + \tau_2$. In this case, $e = l \cdot e_1$ and $e' = l \cdot e_1'$ for some e_1 and e_1'. We must show that $\cdot \vdash l \cdot e_1' : \tau_1 + \tau_2$. By inversion on the assumption $\cdot \vdash l \cdot e_1 : \tau_1 + \tau_2$ with the rule (I-+), we get $\cdot \vdash e_1 : \tau_1$. By applying the induction hypothesis to this and the assumption $e_1 \mapsto e_1'$, we get that $\cdot \vdash e_1' : \tau_1$. By then applying the rule (I-+), we conclude $\cdot \vdash l \cdot e_1' : \tau_1 + \tau_2$.

Case $\ldots$: $\ldots$ □

2 On Programs

As with proofs, when we ask you to define programs satisfying certain conditions, your solution must be clear and clearly correct. This means that it must satisfy the specification set out in the task statement and that it must work on all corner cases. It also means that it must be written so that this is clearly the case. If we

do not ask you to prove your implementation is correct, then you do not need to do so.

If the task can be solved by giving a simple expression, then it is sufficient to do so. For example, if we ask you to define a function f of type $\tau \to \tau$ **list** that takes an argument of type τ and produces a list of length one containing that element, it is sufficient to write

$$f = \lambda x.\mathbf{fold}(cons \cdot \langle x, nil \cdot \langle\rangle\rangle).$$

When the task is non-trivial, we encourage you do the following.

1. Define helper functions. Give them an appropriate name and state their intended type. If it is not immediately clear what the function does, write a brief sentence explaining its purpose.

2. Name other sub-expressions. Give them an appropriate name and state their intended type. For example, the above definition of f could be made clearer by saying:

   Let $\texttt{Nil} = nil \cdot \langle\rangle : \tau$ **list** denote the empty list. Then take

   $$f = \lambda x.\mathbf{fold}(cons \cdot \langle x, \texttt{Nil}\rangle).$$

3. Use whitespace to help suggestively structure your definitions. You can introduce newlines using "\\", as well as increasing amounts of space in math mode using "\,", "\:", "\;", "\ " (backslash followed by a space), "\quad", and "\qquad". You can align expressions using the `align*` environment, where `&` denotes an alignment point. For example,

   ```
   \begin{align*}
     a &= \bscase{e}{x}{\text{some long expression to the end
                               of the line}\\
       &\qquad\qquad}{y}{\text{some other long expression}}
   \end{align*}
   ```

   produces

   $$\begin{aligned} a = \mathbf{case}\ e\ \{&l \cdot x \Rightarrow \text{some long expression to the end of the line} \\ &\mid r \cdot y \Rightarrow \text{some other long expression}\} \end{aligned}$$

In all cases, please use the macros we provide in the handout. This not only saves you time, but the consistency in notation makes it easier for us to quickly grade solutions.

References

[1] 15-312 Course Staff. *Proof-Writing Criteria & Guidelines*. 2017. URL: https://www.cs.cmu.edu/~rwh/courses/ppl/handouts/312-proofs.pdf.

[2] Paul R. Halmos. "How to Write Mathematics". In: *L'Enseignement Mathématique* 16 (1970), pp. 123–152. URL: https://www.di.ens.fr/~bouillar/Stages/Halmos-How-To-Write.pdf.

[3] Kevin Houston. *How to Write Mathematics*. 2009. URL: http://www.kevinhouston.net/pdf/htwm.pdf.

[4] Katherine E. Stange. *Rubric for Grading Student Solutions*. URL: http://math.colorado.edu/~kstange/grading-rubric.pdf.

[5] Jenny Wilson. *A Primer on Mathematical Proof*. URL: http://www.math.lsa.umich.edu/~jchw/PrimerOnProof.pdf.