Development of skills for my EP

Izaak van Dongen

June 4, 2018

1 Problems and how they were overcome

Problem	What I did to overcome it
Little to no knowledge of the basic usage of LaTeX	In order to develop my basic knowledge of LaTeX, I mainly used the online resources `https://www.latex-project.org/about/` and `https://www.sharelatex.com/`
I wanted to customise my document to take up more of the page	By default, LaTeX uses really large margins, to the point where I found it kind of ugly, and most of all a waste of paper. I found that this could be solved with a package called 'savetrees'. Another option I found was to set the margin myself by loading the geometry package.
I wanted to include code I'd written in my LaTeXdocument	I found out about the listings package and found a suitable configuration online, that I added to somewhat with other sources. I also figured out that I could have LaTeXdirectly read from my source files rather than having to include them in my .tex file.
I had to create a bibliography, using LaTeX, or, as I now know, BibTeX.	I did some research on how bibliographies worked in LaTeX. I had already heard of BibTeX citations, and in fact most of the sources I'd used up to this point had provided an option to "export BibTeX citation". Because of this, I could narrow my search terms somewhat, and soon found these pages: `http://www.bibtex.org/Using/` and `https://www.sharelatex.com/learn/Bibliography_management_with_bibtex`. With these, I'd soon compiled my own .bib file with extensive records of all of my sources.
Creating specifically a **Harvard-referenced** bibliography, customised to my liking.	As this seemed to be a requirement of the EP qualification, I needed to modify my BibTeX workflow to the point where everything was Harvard-referenced with author-year citations. I found that this was apparently quite difficult to do with vanilla BibTeX so I decided to use natbib with the agsm style…

Loading the natbib package caused LaTeX to crash while compiling the output PDF, with a seemingly gibberish error	After much Googling to no avail, I applied an approach I'd picked up from programming PostScript (another language that compiled to PDF for my purposes). When I first started programming PostScript, I didn't use an interpreter but rather Preview, so I didn't have access to error messages. The craft to finding bugs was to section off parts of the source with comments and then slowly release them again until you found the bug. By doing this I found that loading savetrees was causing the crash. I didn't end up finding out what the actual conflict was being caused by, but instead used the alternative fullpage package.
Creating nice-looking, effective tables in LaTeX.	For this project I had to create a number of tables, for various documents such as my source evaluation, this document, my diary and at some point I played around with some tables in my dissertation. These tables generally needed to be very long, which standard LaTeX“tabular” mode didn't support. I found out about the longtabs package, together with generic tabulation and also had to increase my arrayfill to make a spacing between my table cells that I found satisfying.
Creating a unique-feeling document with LaTeX	LaTeX has many virtues, but it has a particularly distinctive look, at first, which is mainly down to the Computer Modern font. In order to set my project apart a bit, I spent some time experimenting with different fonts and how to load them properly in LaTeX. I ended up with what you're seeing now, and am overall quite pleased. I also added many other package for small visual tweaks. See `preamble_compr.pdf` for before/after comparison of my LaTeX setup.
Effectively communicating the maths and computing related parts of my project.	These are subjects where I have relatively more expertise than a lot of people. Because of this, I find it easy to gloss over things that really are quite important to explain, and just in general to take it all a bit too fast. After discussing with my EP tutor, who said to think of it like explaining things to my grandma, I asked my grandma for what she would find helpful. Because of this, I decided to add more visual explanations for things. You can see this in the assortment of figures in my dissertation. I also realised that I needed to be more obvious about explaining what my code does, seeing as it constitutes such a large portion of my project. Because of this, I started extensively commenting all of my source code, not just giving a general indication of what the program does, but really trying to give as much of an explanation of as much of the code as I can.
Bugs in the code!	Frequently thoughout this project I would have problems running various bits of code. This was especially stressfull as my dissertation is comprised of bits of code in itself. When this happened I would have to adapt, and use various techniques to try and fix things. After searching for any error messages or stack dumps generated by buggy code, I would just start removing bits of code until it worked again. This would tell me which bits were broken, and from there it's generally a lot easier. This general problem was also helped by starting to more rigorously test the Python programs I wrote, with the unit testing idiom. By ensuring that each component works as it should, it's a lot easier to be confident in your program. It also makes it a lot easier to track down which component is causing the problems if it does break again.