**Design Document**

**Synopsis:**

When I started this assignment I first had to learn Fortran Legacy code. I started off with looking at Fortran IV examples but I felt like it was not sticking in my head I was not able to understand it properly. I moved to learning Fortran 90/95 first and it had mechanics I was able to visually understand easier to start learning Fortran. Fortran 90/95 also had some mechanics that are similar to Lua programming (Ex. If blocks use the same format in lua and Fortran 90/95). After I understand Fortran 90/95 I was then able to learn Fortran IV so I could start my assignment.

Next I started on understanding the Fortran IV code that the professor wants converted to Fortran 90/95, it took me a while to start to grasp and understand what is going on. Especially the use of Hollerith constants and GO TO statements that were no longer used in Fortran 90 took me a while to be able to understand how to convert properly.

Once I got a basic understanding of the assignment code, I started working on the fortran 90 conversion from the top to bottom of the IV code. I started doing research and noticed Hollerith constants are notoriously unstable and should not be used and these variables were converted to strings/characters. Once that changed happened, this drastically changed the entire outcome of the converted code.

When I converted all the variables I started to work on the overall loop of the program, I took reference from the flow chart given in the assignment documentation and I tried to closely follow that as much as I could. I started by using a do while loop to continuously loop until the user types “exit” to leave the program, and after that I started to check the hierarchy of the characters in the input string just like the Fortran IV code did. I noticed the Fortran IV code uses GO TO statements and continue/stop statements which should not be used any longer and I converted those into do while, if, for statements.

Once I got the input string and hierarchy of the string characters, I started to convert that string into reverse-polish notation from the algebraic notation. Fixing the easy problems first like uppercase to lower case and the format of the loops helped me get closer to the final goal of finishing the assignment and got all the easy stuff out of the way. I started on a blank page when I started on the assignment to convert everything to Fortran 90/95 and continued to work on that, instead of changing to any other scheme.

A lot of features were removed/changed on the way from going to Fortran IV to Fortran 90/95. Some of these features that were removed/changed were GO TO statements, continue / break statements, Hollerith constants, .EQ. to ==, etc. These features were the core mechanics of the Fortran IV code and changing this ultimately changed the final result of the Fortran 90 code and this will have drastic differences. For example the updated code is longer compared to the Fortran IV code and a different programmer was used on the project compared to the original project (programmer that only knows C/Java vs a programmer that might only know Fortran) and the way I approach problems would be different compared to the original author.

**Identify Legacy Features:**

• Fortran IV used .EQ. and in Fortran 90/95 it was converted to == which most modern day languages use this new format to test if something is equals. This applies to the other testing cases such as; /= , >=, <=, etc.

•Fortran IV to represent integer uses “Inter num” as their format while Fortran 90/95 changed it to “integer :: num”

•Fortran IV uses GO TO statements to go to different parts of the program, while Fortran 90/95 phased out GO TO which is shown to be bad programming style and changed it to for, if, while loops etc to iterate through code. GO TO is very hard to debug and phasing that out to use proper loops is helpful.

•Fortran IV uses continue, stop as a normal programming technique and this is also shown to be bad programming techniques and that has been phased out in Fortran 90/95

•Fortran IV uses if “do stuff” as their if statement while Fortran 90/95 uses If “statement” then ... code ... end if as a normal if loop block. This is easier to read and understand compared to the old if block style in Fortran IV.

• Fortran IV uses code that has “1H” and this is known as Hollerith constants, this is from the older Fortran era’s and is also notoriously error prone. This is no longer used in Fortran 90/95 and could just use simple characters/strings (for our example in the project)

**Q & A:**

• **Would it have been easier to re-write the program from scratch in a language such as C?**

It depends on the background of the programmer, since being a student in the 2000s we start learning with C programming while legacy programmers might have started with Fortran. Since my background is mostly C it would have been easier for me to re-write the program from scratch instead of converting legacy code.

• **What were the greatest problems faced during the re-engineering process?**

The greatest problem faced was that I had to first learn Fortran IV to be able to convert the legacy code to Fortran 90/95. Next I also had to learn Fortran 90/95 to see the changes compared to the different versions. I was not able to understand the legacy code till I learned this new language, and learning it quickly doesn’t help me make better/optimized code to be converted to Fortran 90.

**• Is your program shorter or longer? Why?**

My program is longer than the Fortran IV legacy code, this is because standards change and also I am not the same programmer that wrote the original code. The programmer that wrote the original code would be able to easily optimize it and convert it over easier, while a programmer that did not write it would have a more difficult time. Also the Fortran IV code relies on GO TO statements and this alone will output less code than Fortran 90/95 if/for/do while loops that have conditional checks.

**• Is there a better way of writing the program?**

When the code is converted from Fortran IV to 90/95 then yes there is a better way of writing the program as standards change and functions change. For example in the Fortran IV code it uses “1H” and this is known as Hollerith constants from the older Fortran era’s and this is notoriously error prone. Changing everything to strings/char’s would be a better way to write this program as the first step.

**My Experience with Fortran:**

Fortran was definitely a new experience for me, University has only really taught us mostly C and a little bit of Java / Perl . At first it was hard to grasp and understand the new concepts, it was actually very hard to understand Fortran IV at the start and I had to look at Fortran 90/95 examples first because it was much easier to understand at the start.

I did a bit of Lua coding on my spare time during the summer breaks, and I learned that Fortran 90/95 and Lua use very similar style if block’s standards and this helped me quickly feel more comfortable in the language. Once I was able to grasp Fortran 90/95 I was able to work backwards from there and start to understand Fortran IV better.

Fortran is very learnable with a programming background, it just may take people a bit to wrap their head around any new language.