**Design Document**

**Synopsis:**

Since I already did my first assignment in Fortran IV I knew a good amount of Fortran IV and I was able to quickly get into this assignment at a quicker pace compared to my first assignment. Except in this assignment we now had to learn Ada. Ada is a language that was completely new to me but I was able to grasp it much quicker compared to Fortran and I thought I was good to write my assignment in under an hour of learning Ada.

I started off doing the “easy” things which is getting all my variables set up and it took me a day of error’s and nothing working to learn, that thinking Ada was the same as C. My mistake was trying to write string’s like C and I can influence them any way I want, this was incorrect because an array of string’s in Ada all need to be declared a certain size in parameter and length and once that is declared it needs to stay that size throughout (no bigger / smaller or you get an error!). String’s in Ada also index differently, you can set it and start it anywhere you like, while C starts at 0 index.

Once I spent countless hours doing research I finally was able to create an array of “Unbounded\_Strings” in Ada that allowed me to create different size strings in the dictionary. Next it took me a long time to figure out how to even handle these string’s and get something out of them or even print to the screen! What I learned was they could not be treated as string’s in Ada and needed to be treated as “Unbounded\_Strings”.

Once I figured out how to handle Unbounded\_Strings properly, I was able to finally start writing my program. When I converted all my variables to appropriate names and standards in Ada to complete the task, I started to re-engineer the Fortran assignment, which used GO TO statements throughout their code and if/for/while loops that have different standard’s in Ada. I looked over the Fortran code to see what I should transfer to Ada and what legacy structure’s were no longer used. Since I had to treat my “strings” in a special way in Ada because they are Unbounded, I also had to try to follow the Fortran assignment and this itself got a few structure’s changed in the process when I started writing the assignment in Ada. This process will make the Fortran IV code and Ada code to look different because the way they access their “Dictionary” variable (from hangman code) is different. Also a lot of features were changed / removed on the way from going to Fortran to Ada.

My result is a bigger program, but roughly the same “overall structure” as Fortran. I felt I was able to follow it and re-engineer the hangman code into Ada easily (with a few different standards) and noticed the program was not modularized at all. I decided to modularize a few components such as the printing of the hangman into its own function that I can call when I need to print out any mistakes from the user to be seen.

**Identify Legacy Features:**

• Fortran IV used .EQ. and in Ada it was converted to = which most modern day languages do not use this format, they use “==” to check if something is equals to it. If you use “=” instead it does cause confusion while programming in the language.

•Fortran IV to represent integer uses “Inter num” as their format while Ada uses num : integer; as their format.

•Fortran IV uses GO TO statements to go to different parts of the program, while Ada phased this out and should use If/For/While statements instead!

•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 Ada.

•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 has their libraries integrated into the compiler, but Ada is more like C where you need to call your libraries at the start of the program to be able to do any print statements to the screen  
• Fortran IV can declare an array of characters with max size of 20 and 50 elements inside of it very simple like: CHARACTER\*20 DICT(50), but this process cannot be done in Ada, you need to use unbound strings, and create an array of unbound strings that you need to declare at the start. This process in Ada takes a few more steps compared to Fortran.

• Ada uses ‘;’ to terminate any line of code, while Fortran IV does not use this. ‘;’ to terminate code is a modern day feature used in most software program’s and is something most people are use to now.

**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 Ada. 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 from Fortran to Ada.

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

The greatest problem faced was that I had to first learn Fortran IV from assignment 1 before I could even understand the Hangman code in Fortran. Next I had to understand an entirely new language called Ada that follows an entirely different structure compared to Fortran or any language that I am use to. My first mistake was thinking in “C”, this caused me to lose a day trying to figure out why string’s needed to be declared a specific size and can only be that size (no bigger or smaller) in Ada. Trying to learn 2 legacy compiler’s and complete the assignment the way the teacher wants it was my greatest problem during the re-engineering process.

**• 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 Ada if/for/while loops. When I eliminated all the GO TO, and tried to re-engineer the project to Ada in the short time that I was learning it, it made my project much longer. Also I wrote my “hangman” in a function and drawing it takes almost 100 lines of code.

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

I felt like Fortran code did a good job at writing hangman project, besides the GO TO statements and a few legacy structures. I tried to follow it pretty closely while changing some of the GO TO statement, modernize the if/for loops, get rid of any continue’s/break’s and I felt like my code is very similar to the Fortran code. The one thing I changed was I modularized the drawing of the hangman to make it easier to read and understand.

I believe Fortran did a good job writing the code, I only needed to change / add a few components, and I believe Fortran was on the correct route.

**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.

Next I had to understand Ada and my mistake was thinking Ada treat’s it’s strings and index’s the same as “C”. In Ada the index starts where you want it to start and when you declare a string its size needs to be determined before and the string if you want to add onto it or change it needs to be the same size (not bigger or smaller!). It took me a long time to figure this out and properly wrap my head around it. Once I figured out how to setup my dictionary project properly and understand how everything works in Ada I was able to work on the assignment and complete it.