Pascal Translation Reflection

CIS*3190 Assignment 4

Geofferson Camp (0658817)

April 8th, 2016

When I began translating the 8 - Queens Pascal program I decided to start with Ada. I assumed C would be easiest as I was most familiar with it, so I left it until the end and figured it would be a nice way to wrap things up. I started with Ada, because, after reading through the Pascal program a few times I decided Ada and Pascal had the most number of similarities. For the most part, this approach worked well. Translating to C didn't go as smoothly as I had hoped but because I was already so familiar with the algorithm in general, by the time I got to C the task was very manageable.

Translating the 8 - Queens Pascal program into Ada and Fortran was straight forward. All three languages seem to share very similar characteristics such as variable array indices, and program structure such as function, subroutine, and variable positioning. The requirement that caused the most issues was writing to a file in Fortran. Problems arose in Fortran because, unlike in Ada, I had not implemented file output during previous assignments. I was not familiar with the output formatting options; it did not take long to have the program printing to a file, but formatting the output took significantly longer.

The most challenging part of the assignment was understanding the 8 - Queens algorithm. The combination of non-descriptive variable names, recursion, and confusing array bounds, lead to this result. The actual programming was easier to accomplish because all three languages function similar enough that since I already had written in a program in Ada, Fortran, and C, writing one more that I already had the algorithm for was simple. Luckily, the only part of the program I had to design, the printing of the output, had similar implementations in all three languages and the same algorithm could be used in all cases.

Converting the program into C was, very briefly, more challenging than translating it to Ada and Fortran. As I began to write out the C code I realised the arrays would not be able to start or finish at the same values as the other three languages. This meant the math that would have been used to access the correct array addresses would not work correctly. The solution I used to get around this issue was to offset the result of the previously mentioned math by the same amount I had to offset the beginning of each array in order to force an initial address of 0.

I think the language chosen to write a program in, or be translated to, should depend on many factors. I believe choosing a language based solely on how easy it would translate would be irresponsible. More importantly, programmers should make this decision based on the purpose and functionality of the program and compare that to strengths and weaknesses of languages in consideration. However, I do think it is important to consider the scale of the project and functional requirements of the program. For example, the 8 - Queens program is fairly simple and all of its aspects were handled efficiently in all three languages it was converted to, with the exception of C where some “hacks” were implemented in order to return desired results. In cases such as this I would choose whichever language I was most familiar with, as the long term negative factors are negligible and language familiarity would increase production significantly. I also think it is possible that languages with similar characteristic, in other words, easily translatable, would have similar strengths and weaknesses, and therefore make logical choices to translate between, as-per the criteria I outlined above.