CIS3190 – Assignment 4 (Reflection Report) Rabia Qureshi (1046427)

My last assignment for this course was based on implementing an ALGOL-60 algorithm for calculating e to a given number of digits in four different languages. This reflection report will provide a synopsis on my experience implementing the algorithm in each of those languages. Before I began coding, I tried fully understanding the algorithm. Since ALGOL-60 was unfamiliar to me, I investigated ALGOL-60 syntax and structure requirements as I followed the algorithm.

I implemented the algorithm in Python first because I find Python code is easiest to write and understand. One benefit of coding in Python was that it supported type inference, so variable initialization would take care of types for me without the use of any reserved keywords. For instance, while C would require something like int index = 0, Python understands the variable index is an integer when one initializes it with an integer value: index = 0. I also believe one of Python's best features is being able to return multiple values from one function in the simplest way: numDigits, filename = getUserInput(). All other languages I used for this assignment would require me to pass those variables in as pointers (C) or by specifying intent attributes or modes (Ada, Fortran). Like every other language, Python had some limitations. Python relies on whitespace to show which code snippet is part of a block of code – I believe using curly braces as are used in C or using end statements (Ada, Fortran) are less risky ways of showing code blocks as they are more explicit.

I implemented the code in C next, because of all four languages, I am most familiar with C. The only major disadvantage I found of coding in C was that reading from standard input and writing to files was less intuitive, as keywords and format specifiers such as printf, scanf, %d, and %s must be used instead of simpler functions such as Python's print() and input().

I then completed my Fortran program. Since I had already implemented the algorithm twice, completing the code in Fortran did not feel like much of a challenge. One aspect of Fortran that took me some time to get used to was how array elements are accessed: arr(1) versus the more commonly used arr[1]. I found I/O and formatting most confusing in Fortran. For example, write(*, '(2I5.3)') means the formatting is repeated twice, a 5-character integer is used in the output, and three digits are displayed. I found this unintuitive.

I worked on my Ada code last and found it easy to complete as I believe its structure and syntax is quite similar to that of Fortran. Though there are many similarities between Fortran and Ada, something I like more in Ada than Fortran is how variables used in Ada subprograms are defined. In Fortran, they're passed in and shown as parameters without their intent attribute. Intent is shown after the implicit none statement in the subroutine, which adds more lines of code to the program and lengthens it unnecessarily. In Ada, however, all parameters are defined with their modes (one of in, out, in out) in the function parameter list.

Small differences between each of the languages seemed to be what caused me most trouble. For instance, C uses a semicolon to denote the end of line, whereas Python uses no character. Second, array indices are defaulted to start at different values for some of the languages (C and Python indices start from 0, Fortran array indices start from 1). Fortran and Ada provide the option to specify the upper and lower bounds for array indices, which made it easier to follow the algorithm provided without making any changes. One of the biggest challenges for me was ensuring the loops in each of my programs would not try accessing out of bound elements. This is because the upper bound specified in some for loops was exclusive, while it was inclusive in other languages. For instance, the statement for i in range(1, numDec + 1) means the last time the for loop will run is when i=numDec, because the second argument of range is exclusive.

Despite the challenges I faced working on this assignment, I enjoyed working with the four languages, somewhat because of the order I chose to work in. Since I am most experienced working with languages with a structure similar to that of Python and C, I chose to do implement the algorithm in those languages first. I chose Fortran next because I found it easier than Ada. Fortran and Ada are similar in structure as well, which is another reason why I worked on their code together.