Pascal Evaluation

Riley Fiske – October 11th, 2023

***Preface – Previous Experience***

Before working on this project, I had never worked with Pascal before, to the point where I had to Google what the file extension for a Pascal program was before even starting. My only knowledge of the language was from in class where we talked about its utility as a learning language in the 90’s.

Note: when running quicksort.exe, it will not remain open since it prints out the sorted array then immediately quits since there’s nothing holding the program open. Uncommenting the line “readln(randomintstring)” will hold the program until something is typed before closing.

***Readability***

From my brief time working with Pascal, it is a quite readable language. It uses many keywords from English to declare things in a simplistic way. For example, when declaring an array, you refer to it as an “array [size] of integer”, which is how you might describe it verbally in spoken language. It is quite clear where statements belong with the use of “begin” and “end” statements acting like the curly braces {} from Java, as well as semicolons to end lines so that it is very clear where one statement ends and the next begins.

The data types included in Pascal are intuitive and have common names that are quite descriptive for what they mean. The special words in Pascal are also indicative of what they do: procedures and functions are called and executed, begin and end encapsulate a set of commands between them, etc. The use of Pascal as a teaching language makes sense given its qualities that make it very readable to beginning programmers.

***Writability***

The writability of Pascal is also quite good and intuitive for a programmer. It writes in a fashion very similar to Java, with {} replaced with “begin” and “end”, = assignment operator replaced with :=, which I read as a shorthand for “variablename : type = value”, where the variable is implicitly typed since it was previously declared, then lists its value, and running a program from a main method among many more similarities. One way it is written that differs from Java is requiring variables to be declared outside the main method, similarly to Fortran. This adds a little clunkiness for the programmer who either needs to know absolutely every variable before beginning to write the main method or having to hop around the program to add variables in certain places rather than wherever they’re currently typing.

One finicky thing I observed when writing Pascal was the assignment operator :=. It is a little annoying to type since one character requires shift to be pressed and the other doesn’t, and this led to several typos.

***Reliability***

Pascal felt overall like a reliable language. Whenever I had a type error, the error description was very helpful in indicating where the error was, the type it received, and the type(s) it was expecting. Pascal does have exception handling with the [“try…except” statement](https://www.freepascal.org/docs-html/ref/refse119.html), however, I did not have to utilize this for the sorting algorithm implemented, similarly for aliases.

***Cost***

Pascal is not a core programming language anymore, and thus would require education costs to utilize it on a large-scale project. However, when it was the go-to learning language in the 90’s, this cost factor was likely much lower since it was used everywhere, and more people knew it to teach it.

***Quick Comparison to Java***

As was mentioned in both the readability and writability sections, Pascal felt a lot like programming in Java, just with a slightly different and older flavor. Many of the same constructs and formatting that exist in Pascal are now used in Java. The brackets for encapsulating sections, ending lines with semicolons, and having a main method are three ways in which this felt familiar.

***Parameter Passing by Value vs. by Reference***

Pascal gives you the flexibility of passing parameters into functions either as a [reference or as the value](https://stackoverflow.com/questions/54604999/pass-by-reference-in-pascal#:~:text=Pass%20by%20value%20means%20a,any%20modification%20modifies%20the%20original.). When you pass a parameter in as a value, that value is modified locally within the method, then any changes are discarded, maintaining the previous value of the variable outside the function. When you pass a parameter in as a reference, this value is modified within the method and these changes apply to the variable outside of the function as well. Depending on what the programmer wants to do with this variable, this flexibility can be extremely helpful to be able to modify multiple values within a method as opposed to calling multiple methods with multiple return types.

Java has a similar idea, but it is less clearly cut for the programmer. Primitive data types that are passed into a method are treated as a value and modified locally within the method. User-defined data types, namely objects, maintain their changes outside of the method, passed like a reference in Pascal.

***Screenshot of Pascal Quicksort Algorithm Execution:***

