Gabbi Forsythe

Dr. Rybarczyk

CS441-01

25 October 2019

Assignment #3

Enumerations in C++ and Java

According to what we learned in class, an enumeration is a data type "in which all of the possible values, which are named constants are provided, or enumerated, in the definition". Enumerations are implicitly assigned integer values (0, 1, …), but any integer value can be explicitly assigned to the values in the enumeration's definition. However, as is the case with most data types, enumerations are not available in all programming languages, nor are they implemented the same way in every language.

For this assignment, I used enumerations in both C++ and Java. These are very different syntactically and semantically. In Java, enumerations are specific instances of the Enum class, while in C++, there is no enumeration class. This makes enumerations in Java significantly more complex, both syntactically and semantically. Because enumerations are a subclass, they are allowed to have constructors and methods, in addition to the named integer values. This is in large contrast to C++ where enumeration values can be assigned integer values, and that is about all they can do.

Another helpful way to analyze the differences between enumerations in C++ and Java is to use the four criteria that were mentioned in class (Readability, Writability, Reliability, and Cost). In my opinion, enumerations are both more readable and writable in C++. Because they do not have the same abilities of a class, the enumerations only contain the information you need (the names and the integer values), and you do not need methods to access this information. However, enumerations in Java are more reliable because they are a class. Classes provide more restrictions which leave less room for error. Also, Java does not allow arithmetic operations on enumerations (C++ does), which adds to the reliability of the program. As far as cost goes, it took significantly more time for me to figure out how to implement the enumerations in Java, and therefore did not feel worth using them. Using these criteria as the basis for my opinion, I prefer C++ over Java when it comes to enumerations. In C++, they are overall easier to understand, and *then therefore*, easier to use.

Pointers and Arrays in C++ and Java

For this project, I reimplemented the C++ program from Assignment #1 to use pointers explicitly. Pointers allow the programmer to interact with specific memory locations. They can also be used to access the heap (dynamic memory). In C++, pointers can be used to place arrays on the heap. Other languages, such as Java, do not have pointers, but provide other ways to access dynamic memory.

In C++, pointers provide a way to put arrays on the heap (using a pointer and the *new* keyword), allowing the programmer to control memory usage. While pointers do not exist in Java, arrays can be placed on the heap using the *new* keyword. This approach does not provide the full control of the memory, it does allow for more control over where the data is stored (heap vs. stack). Another similarity between pointers in C++ and Arrays in Java, is that both are reference types. Arrays in Java are pass-by-reference, rather than pass-by-value, and pointers in C++ are references to locations in memory.

It is important to analyze arrays and pointers using the four criteria mentioned in class: Readability, Writability, Reliability, and Cost. In my opinion, Arrays are much easier to read and write then pointers. This is a biased statement because I had never used pointers before this assignment, but I believe the reason I had never used them before is because they were difficult to read, and therefore difficult to write. Another aspect to think about is reliability. Pointers allow for memory leaks, so if a programmer is inexperienced in using pointers (ex. me), the program becomes significantly less reliable. On the other hand, if you are using arrays in Java, you could run into problems with the automatic garbage collection. As far as cost goes, the cost of figuring out pointers was not worth it. It took significantly less time to build the programs using arrays, because they are like a faithful friend to me, while pointers are like strangers. In summary, I prefer arrays over pointers because they are easier to use, easier to read, and more reliable.