Justin Espiritu
Assessment
CS 331
Dr. Thomas Naps

Assessment

Language: F#
Problem: Hexagonal Number

F#, also known as Function C#, is a strongly typed functional language. Like Java, this means all data, values, are predefined. If I create an a string it cannot be used in as an Integer unless I parse or cast it to one beforehand. Now F# does make assume types like Java, well somewhat. F# is quite a bit more strict than Java in this sense. For example, in F# I cannot just simply just take a float and multiply it by an int. However if I create a function, like I did in my assessment with checkIfHexigonal takes two parameters but none of the types are declared. How F# works in this sense it that whatever is returned in the function it will automatically assume a needed type. If you are using Ionide for Visual Studio Code you can actually see this with the above functions as it will state something like int -> bool. A feature that I found helpful when making my code assessment much easier. This is something that I'm not able to do in Java, the compiler would complain if I don't declare types at all for functions. Even though I can't do something simple such as "4.0 * 4" in F# I do like the fact that I can create a function without declaring their needed types.

Now, as I said before, F# is a functional language also. This is what makes it F# such a powerful language. You are able to create finite and infinite sequences right out of the gate. F# has built in sequence expression which allow a user to create a simple sequence like so: seq { 1 .. 15} which will return a sequence 1 through 15. However, there is a lot more going on with the sequence expression. When I print out a seq in the terminal it might only show the first 5 of the sequence for example "[1; 2; 3; 4; …]". You might be thinking, where are the other numbers in the seq. This is what makes F# so powerful in this sense is that it does lazy evaluation when creating sequences, meaning it will only evaluate the rest of the sequence when it is needed. If

Justin Espiritu
Assessment
CS 331
Dr. Thomas Naps

I wanted to do something like this in Java I would have to do something very similar to what we did in Assignment 8 by creating a multiple functions that will allow me to call-by-need.

I found that using F# for this assessment was actually quite easy. I was actually able to setup an environment for me to use in class while the requirements for this assessment were being explained. I think the biggest challenge I had when doing this assessment was getting used to the strict strongly type language. I mean I already dislike strongly typed languages, Javascript is one of my favorite languages. When I was trying to create the hexigonalEquation function I found it to be extremely annoying that I had to make sure all types match for my program to compile. Specifically it was with using the square root function in my program. Not only did I have to make sure that the parameter being put in was a float and nothing else. The equation that I put inside as parameters, every single number in that equation had to be a float as well. Luckily F# being a strongly typed language told me where the issue was right away. Making it quite simple to fix the issue.

As I said before I found this assessment to be quite trivial. I think what helped make this assignment easier for me was the experience I've gotten from working at the Management Information Office (MIO). When I started working at MIO I had no experience with HTML, CSS, Javascript, AngularJS, SQL, etc. After working there for a year I can say that I am comfortable using all those languages. MIO really helped me understand how to look at a new language and understand how it functions. I think the most helpful thing I've learned from MIO was following the data. Because javascript is not a compiled language I had to put debuggers and walk through my program to find issues which helped me understand the language much better. That's probably the reason why I like Javascript so much.