Adrien Protzel

1. Making logic of programs writing structure with syntax as defined by BNF, and their behavior and meanings with semantics.
2. A BNF can be read as a set of sentences with blank words within the sentences in which you can place certain other partial sentences which will eventually create a whole sentence. For instance, writing a math equation; Start with an expression <exp> which can be a number and finish the expression or an <exp> followed by a + and another <exp>. Where the <exp> can be replaced by any valid sentence in the set. This can help with expanding, modular expressions of math or logic or even an English sentence.

S := <e> | <a> !

<e> := hello <a>

<a> := world | <e>

This simple BNF would read as follows. Start with the vantage which is denoted as S, the symbol := means the sentence is comprised of the following options and the | to segregate the potential options that comprise the sentence. Following are the options that the options make up. For example: sentence S can be either <e> or <a>!, <e> will be the word “hello” followed by an <a>. <a> can be either word “world” or be <e> which could go in a loop and have multiple “hello” words before ending with “ world!”.

1. Initially, to me, the BNF was easier to read because all of the components of a select statement were there in one readable line. However, after looking at it more, I would need to essentially go hunting for the expressions that would fit into the statement. While the flow chart on the other hand, is large but does walk you step by step through each node and option. It’s harder to see an overall look at the structure of the SELECT statement with the flow chart versus the BNf but it is easier to fill out a SELECT using the flowchart.

For positives and negatives, the BNF seems to have more customization and ease of adding new features or more complex expressions. This also makes reading all options and loops more cumbersome. The flowchart is much more restrictive but easy to navigate.