Suppose we wish to add a ‘select’ statement to Core as follows: a select statement has the structure

\[
\text{select } \ b_1 \rightarrow \ S_1 \ | \ | \ b_2 \rightarrow \ S_2 \ | \ | \ ... \ | \ | \ b_n \rightarrow \ S_n \ \text{end;}
\]

where \( b_1, \ldots, b_n \) are boolean expressions (i.e. \( \langle \text{cond} \rangle \)'s), and \( S_1, \ldots, S_n \) are \( \langle \text{stmt seq} \rangle \)'s; and the number of \( b \)'s is equal to the number of \( S \)'s. We execute the select as follows: if \( b_1 \) evaluates to true, execute \( S_1 \) and you are done; if \( b_1 \) evaluates to false, and \( b_2 \) to true, execute \( S_2 \) and you are done; \ldots; if \( b_1, \ldots, b_{(n-1)} \) evaluate to false and \( b_n \) to true, execute \( S_n \); if \( b_1, \ldots, b_n \) all evaluate to false, simply go to the next statement that follows the select (i.e., the select, in this case, is a ‘no-op’). Note that the “\( | \ | \)” is a new terminal symbol that is used to separate each \( \langle \text{cond}, \langle \text{stmt seq} \rangle \rangle \) pair from the next pair.

Now for the problems:

1. (8 points). Add the select statement to Core by modifying the BNF grammar appropriately. (Use “pure BNF”, i.e., what we have been using in class. If you were to use “extended” BNF, answering the next question will be much harder.)

2. (12 points). Using the notation from the class notes (copies of slides), write down the Execute-select-statement procedure and any other procedures you need to add to Core’s interpreter to implement the select statement. Don’t worry about the Parse-select-statement procedure.

You may assume the array representation of the parse tree when answering this question. You may also assume procedures such as Execute-stmt-seq or functions like Eval-Cond etc. already exist.

Important Note: The problem is not asking you to show how you can achieve the effect of the select statement by using multiple if statements or anything like that. You are being asked to introduce a new statement into the CORE language so that the CORE programmer can write, as part of his or her CORE program, statements such as:

\[
\text{select } \ (X > 0) \rightarrow \ Y = 10; \ | \ | \ (Y > 0) \rightarrow \ X = 10; \ \text{end;}
\]