Problems:

Consider the interpreter for language Simple3 (available on the website as SIMPLE3INTERPRETER.zip), modify the interpreter in the following ways:

1. The interpreter currently implements static scoping. Change the implementation so that it supports dynamic scoping. Show that your implementation works by demonstrating that the modified interpreter interprets the following program according to the definition of dynamic scoping:

```
declare step = 10;
declare inc(x) {
    return x+step;
}
// start a local scope
{
    declare step = 2;
    put inc(5);
}
```

2. The existing interpreter implements positional parameter correspondence. Design and implement keyword correspondence for function parameters for both function call statements and function calls in expression. Demonstrate that your implementation works with following program:

```
declare idiv(dividend,divisor) {
    return dividend/divisor;
}
put idiv(divisor->2,divident->3)
```
3. Most C-like programming languages support a ‘break’ keyword that allows the user to break out of loops:

```c
while(<exp>) {
    ...
    if (<exp>)
        break;
    ...
}
```

The intended meaning is that break immediately terminates the surrounding loop and execution continues right after the loop. A break statement that is not surrounded by a loop should terminate the execution of the program.

Implement the ‘break’ statement and demonstrate that it works with the following examples:

```c
// this program prints out the integers 1 thru 9 twice.
declare j = 1
while (j <= 2) {
    declare i = 0
    while(1) {
        i = i + 1
        if (i == 10)
            break
        put i
    }
    j = j + 1
}

and

// this program prints out the value 10 and then stops.
declare i = 10
put i
break
i = i + 1
put i
```

**Hint:** Take a look at the implementation of the ‘return’ statement, the ‘break’ statement can be implemented in a similar way.

Hand in your source code and proof that your program works. The easiest way to show that your program works is to include a screen capture of the window that is executing your parser/grammar. Good software engineering/coding practices count towards your grade.