King Saud University
College of Computer and Information Sciences
Computer Science Department
CSC 340: Programming Language and Compilation
Programming Assignment 2 & 3: Using JFLex & BYACC/J

For this assignment, you will use JFLex for the lexical part of the task and BYacc/J for writing a parser for a simple language described as:

P → D ; P | D
D → def id(ARGS) = E;
ARGS → id , ARGS | id
E → int | id| if E1 OP E2 then E3 else E4 | ***do*** *E1 until E2*
*| E1+E2 | E1-E2 |E1*E2|E1/E2 |**E1^E2**| id(E1, …, En)*
*OP → == | > | < | >= | <= | **<>***

Where <> is equivalent to != in Java; and ^ is the power operation, for instance 2^3 is 8.

Here is a simple program for computing Fibonacci numbers, written in the language

```
def fib(x) = if x = 1 then 0 else
             if x = 2 then 1 else
               fib(x - 1) + fib(x - 2)
```

Part I: Lexical Analysis of Mython

You will use JFLex to capture the tokens of the language. To do this, you need to define a constant value associated with each token, so that you can return this value every time your parser needs a new token.

- **Keywords**: def, if, then, else, **do**, until.
- **Punctuation elements**: ( ) : , ;
- **Operators**: = , +, - ,*, /, ^, == , > , < , <= , >= , <>
- **Identifiers**: String of alphanumeric (and _) starting with an alphabetic character
- **Literals**: integer only

Hint:
There are lots of online resources on JFlex including tutorials on youtube. See for example https://www.youtube.com/watch?v=IV1Rwq7ERR4

- **Due Date for part Sunday 20 March.**

**** you are requested to submit your code and Screenshot for sample output***
**** Please do not forget to write your name and your attendance number***

Part II: The parser

1) Rewrite the grammar to eliminate any ambiguity and make sure that the associativity of the operations + , – , *, and /are left to right. **However, the associativity of the power operation ^ is from right to left**.

2) Write a parser of the rewritten grammar using BYACC/J. Once you have built your parser, you should be able to process programs written in the language described above. When BYACC finds input that doesn't match the grammar, it automatically terminates with the message 'Syntax error'. You will need to write an error routine (yyerror) that also prints out the line number before this termination, and a main method that read a program from a file.

The above program should compile and have no error

Hint: Also, there are lots of online resources on BYACC/J (see for example http://byaccj.sourceforge.net/) and video tutorials.

Due Date for part 2 Wednesday 06 April

**** you are requested to submit your code and Screenshot for sample output***

**** Please do not forget to write your name and your attendance number***