Interpreter Programming Project
CS 3361 Concepts of Programming Languages

Below is the syntax and semantics for a single Boolean expression followed by a period. Write a program which prompts the user to input a file name which contains the Boolean expression or simply to input the string to be checked (indicate which input method you will use in your comments). Although it may not matter (depending on your implementation), you may assume that no input will be longer than 1000 characters in length. Expressions may contain white spaces and white spaces should be considered to be delimiters (i.e. a white space between : and = of the assignment operator or between the − and > of the implication symbol would be a syntax error). The program should check if the expression in the file is of valid syntax and, if valid and has no undefined variables, output the value of the expression. If there is an undefined variable, the program should give a message that the variable was undefined and continue checking the syntax. If the same undefined variable is used more than once, only one error should be given for that variable. The output should either be an error message(s) or a message that gives the value of the expression. **You must use the techniques taught in the class - this is a recursive descent interpreter.**

Syntax: (note: for ∨ use the uppercase letter "V", for ∧ use the caret symbol, and a valid *var* is a single lower case letters)

			Selection Sets
$\langle B \rangle$	::=	$\langle VA \rangle \langle IT \rangle$.	$\{\#, \sim, T, F, var, (\}$
$\langle VA \rangle$	::=	$\#var := \langle IT \rangle \; ; \langle VA \rangle$	$\{\#\}$
	::=	ε	$\{\sim, T, F, var, (\}$
$\langle IT \rangle$	::=	$\langle CT \rangle \; \langle IT\_Tail \rangle$	$\{\sim, T, F, var, (\}$
$\langle IT\_Tail \rangle$	::=	$-> \; \langle CT \rangle \; \langle IT\_Tail \rangle$	$\{->\}$
	::=	ε	$\{., ;,)\}$
$\langle CT \rangle$	::=	$\langle L \rangle \; \langle CT\_Tail \rangle$	$\{\sim, T, F, var, (\}$
$\langle CT\_Tail \rangle$	::=	$\vee \; \langle L \rangle \; \langle CT\_Tail \rangle$	$\{\vee\}$
	::=	$\wedge \; \langle L \rangle \; \langle CT\_Tail \rangle$	$\{\wedge\}$
	::=	ε	$\{->, ., ;,)\}$
$\langle L \rangle$	::=	$\langle A \rangle$	$\{T, F, var, (\}$
	::=	$\sim \langle L \rangle$	$\{\sim\}$
$\langle A \rangle$	::=	T	$\{T\}$
	::=	F	$\{F\}$
	::=	var	$\{var\}$
	::=	$(\; \langle IT \rangle \;)$	$\{(\}$

Syntactic Domains:
$\langle B \rangle$: Bool_def
$\langle VA \rangle$: Var_def
$\langle IT \rangle$: Imply_term
$\langle IT\_Tail \rangle$: Imply_tail
$\langle CT \rangle$: Connect_term
$\langle CT\_Tail \rangle$: Connect_tail
$\langle L \rangle$: Literal
$\langle A \rangle$: Atom

Semantic Domain:
$\eta : b = \{true, false\}^{\circ}$
$\sigma : st = var \rightarrow b$

Semantic Function Domains:
α : Bool_def $\rightarrow$ b
γ : Var_def $\times$ st $\rightarrow$ st
β : Imply_term $\times$ st$\rightarrow$ b
δ : Connect_term $\times$ st$\rightarrow$ b
$\lambda : b \times$ Imply_tail $\times$ st$\rightarrow$ b
$\mu : b \times$ Connect_tail $\times$ st$\rightarrow$ b
ϕ : Literal $\times$ st$\rightarrow$ b
ψ : Atom $\times$ st$\rightarrow$ b

Semantic Equations:
$\psi(\ll T \gg, \sigma) = true$
$\psi(\ll F \gg, \sigma) = false$
$\psi(\ll var \gg, \sigma) =$ if $\sigma[var] \neq \perp$ then $\sigma[var]$ else $\top$
$\psi(\ll (\langle IT \rangle) \gg, \sigma) = (\beta(\ll \langle IT \rangle \gg, \sigma))$
$\phi(\ll\sim \langle L \rangle \gg, \sigma) =$ if $\phi(\ll \langle L \rangle \gg, \sigma) \neq \top$ then $\neg\phi(\ll \langle L \rangle \gg, \sigma)$ else $\top$
$\phi(\ll \langle A \rangle \gg, \sigma) = \psi(\ll \langle A \rangle \gg, \sigma)$
$\delta(\ll \langle L \rangle \langle CT\_Tail \rangle \gg, \sigma) = \mu(\phi(\ll \langle L \rangle \gg, \sigma), \ll \langle CT\_Tail \rangle \gg, \sigma)$
$\mu(\eta, \ll \vee \langle L \rangle \langle CT\_Tail \rangle \gg, \sigma) =$ if $\eta \neq \top$ and $\phi(\ll \langle L \rangle \gg, \sigma) \neq \top$ then
$\quad \mu(\eta \vee \phi(\ll \langle L \rangle \gg, \sigma), \ll \langle CT\_Tail \rangle \gg, \sigma)$ else $\top$
$\mu(\eta, \ll \wedge \langle L \rangle \langle CT\_Tail \rangle \gg, \sigma) =$ if $\eta \neq \top$ and $\phi(\ll \langle L \rangle \gg, \sigma) \neq \top$ then
$\quad \mu(\eta \wedge \phi(\ll \langle L \rangle \gg, \sigma), \ll \langle CT\_Tail \rangle \gg, \sigma)$ else $\top$
$\mu(\eta, \ll \gg, \sigma) = \eta$
$\beta(\ll \langle CT \rangle \langle IT\_Tail \rangle \gg, \sigma) = \lambda(\delta(\ll \langle CT \rangle \gg, \sigma), \ll \langle IT\_Tail \rangle \gg, \sigma)$
$\lambda(\eta, \ll -> \langle CT \rangle \langle IT\_Tail \rangle \gg, \sigma) =$ if $\eta \neq \top$ and
$\quad \lambda(\delta(\ll \langle CT \rangle \gg, \sigma), \ll \langle IT\_Tail \rangle \gg, \sigma) \neq \top$ then
$\quad \eta \rightarrow \lambda(\delta(\ll \langle CT \rangle \gg, \sigma), \ll \langle IT\_Tail \rangle \gg, \sigma)$
$\lambda(\eta, \ll \gg, \sigma) = \eta$
$\gamma(\ll \#var := \langle IT \rangle; \langle VA \rangle \gg, \sigma) = \gamma(\ll \langle VA \rangle \gg, update(\sigma, \beta(\ll \langle IT \rangle \gg, \sigma)/var))$
$\gamma(\ll \gg, \sigma) = \sigma$
$\alpha(\ll \langle VA \rangle \langle IT \rangle . \gg) = \beta(\ll \langle IT \rangle \gg, \gamma(\ll \langle VA \rangle \gg, \sigma[\,])$