```
# -*- coding: UTF-8 -*-

##
## Data:  22-04-2011
##

from Logic.Expressoes import Variavel
from Logic.Interpretadores import ExpressionInterpreter
from Util.Util import cls, stop, creditos

# Banco de variaveis definidas pelo usuario
variaveis = {}

##
## Funcoes do programinha
##

def gerenciador():
    ''' Interface de manuseio das variaveis logicas da aplicacao '''

    retornar = False

    while not retornar:
        cls()
        print " (<) MainMenu [R]                              [A] Ajuda"
        print
        print " Banco de Variaveis + ------------------------------ +"
        print
        print "   [1] Adicionar                                      "
        print
        print "   [2] Remover                                        "
        print
        print "   [3] Listar Variaveis                               "
        print
        print " + ------------------------------------------------- +"
        print

        opcao = raw_input(" >> ")

        # Adiciona variaveis no dicionario global de variaveis logicas
        if opcao == '1':
            cls()
            print " Entre com as variaveis:   (Deixe um campo vazio para sair)"
            name = None
            while name != '':
                name = raw_input(" >> ")
                if name != '':
                    input = raw_input(" (T)rue ou (F)alse: ")
                    value = False if input in ['F', 'f'] else True
                    variaveis[name] = Variavel(name, value)
                    print

        # Remove variaveis do dicionario global de variaveis logicas
```

```
        elif opcao == '2':
            cls()
            print " Digite o nome da variavel a ser removida:"
            name = raw_input(" >> ")
            if name in variaveis.keys():
                del variaveis[name]
                print
                print " [OK] Variavel removida..."
            else:
                print
                print " [ERRO] Variavel inexistente!..."

            stop()

        # Lista as variaveis logicas do sistema
        elif opcao == '3':
            cls()
            print " Variaveis Armazenadas + ------------------------- +"
            print
            if variaveis.values() == []:
                print "   [VAZIO] Nenhuma variavel armazenada"
            else:
                for var in variaveis.values():
                    print "   ", ("[" + str(var) + "]"), "=", var.eval()
            print
            print " + ---------------------------------------------- +"
            print
            stop()

        # Topico de ajuda
        elif opcao in ['A', 'a']:
            cls()
            print
            print " REGRAS PARA VARIAVEIS + ------------------------------ +"
            print
            print "  As variaveis do LogSent podem ser quaisquer caractere"
            print " alfanumerico, contanto que nao tenha em sua composicao "
            print " nenhum dos caracteres dos operadores logicos ou parenteses."
            print
            print " Ex: p"
            print "     suff3r"
            print
            print " + ---------------------------------------------------- +"
            print
            stop()

        # Muda o valor da flag RETORNAR para True, para sair do loop
        else:
            retornar = True


def avaliador():
    ''' Interface de avaliacao de expressoes '''
```

```
    retornar = False

    while not retornar:
        cls()
        print " (<) Main Menu [R]                              [A] Ajuda"
        print
        print " Avaliacao de Expressoes + ------------------------- +"
        print
        string = raw_input(" Expressao:\n\n >> ")

        # Avalia a expressao e exibe o resultado
        if string not in ['R', 'r', 'A', 'a']:

            try:
                # Instancia o avaliador e avalia a expressao
                avaliador = ExpressionInterpreter()
                expression = avaliador.eval(string, variaveis)

                # Recupera os logs de cada processo para serem imprimidos
                processos = avaliador.processesLog

            except Exception, t:
                print
                print " [ERRO] Problemas ao eval esta expressao!"
                print " [", t, "]"
                print
                stop()
                return

            # "Apaga" a tela e imprime os resultados
            cls()
            print
            print " Variaveis + ------------------------------- +"
            print
            for var in variaveis.values():
                print "   ", ("[" + str(var) + "]"), "=", var.eval()
            print
            print " Avaliacao Procedural + ---------------------- +"
            for i, processo in enumerate(processos):
                print "\n [", (i + 1), "]\n", processo
            print "\n" * 2
            print " Resultado  ---  [", expression, "]  --- "
            print
            print " :: [", str(expression.eval() ) ,"] ::"
            print
            stop()

        # Muda o valor da flag RETORNAR para True, para sair do loop
        elif string in ['R', 'r']:
            retornar = True

        # Topico de ajuda
```

```
        elif string in ['A', 'a']:
            cls()
            print
            print " OPERADORES + ---------------------------------------- +"
            print
            print "   !   (not)"
            print "   &   (and)"
            print "   |   (or)"
            print "   *   (xor)"
            print "   ->  (implica)"
            print "   <-> (bi implica)"
            print
            print " EXPRESSOES + ---------------------------------------- +"
            print
            print "   O LogSent opera com expressoes binarias, ou seja, o usua-"
            print "  rio deve explicitar as precedencias usando parenteses."
            print
            print "  CUIDADOS: - Apenas variaveis que estao no banco podem ser "
            print "              usadas nas expressoes."
            print
            print "            - Formule bem a sua expressao para que nao haja "
            print "            erros no resultado, ja que o LogSent nao tem um "
            print "            analisador sintatico."
            print
            print "  Exemplos de proposicoes validas:"
            print
            print "    (p -> q) &  r"
            print
            print "    !t <-> ((p & q) -> a)"
            print
            print " + ------------------------------------------------------ +"
            print
            stop()

##
## Funcao principal do programa
##

def main():
    sair = False
    opcoes = {  '1' : gerenciador,
                '2' : avaliador,
                '3' : creditos }

    while not sair:
        cls()
        print " LogSent [Eval]"
        print
        print " + Main Menu + --------------- +"
        print
        print "  [1] Banco de Variaveis        "
        print
        print "  [2] Avaliar Expressoes        "
```

```
        print
        print "  [3] Creditos                  "
        print
        print "  [S] Sair                      "
        print
        print " + ---------------------- v 1.0 +"
        print

        opcao = raw_input(" >> ")

        if opcao in ['1', '2', '3']:
            acao = opcoes[opcao]
            acao()

        elif opcao in ['S', 's']:
            sair = True

        else:
            print
            print " [ERRO] Opcao invalida!"
            stop()
```