UNISINOS - UNIVERSIDADE DO VALE DO RIO DOS SINOS

Curso: *Informática* **Disciplina:** *Compiladores 1* **Horário: TUTORIA**
Prof. Fernando Osório **E-mail: osorio@exatas.unisinos.br** **Data:14/12/2006**

Nome do Aluno: ***Rodrigo Lemieszek Vera Cruz***
Nro. de Matrícula: ***492095***

ASSINATURA: _____

PROVA - GRAU C

Implemente um programa de acordo com o solicitado abaixo, usando geradores automáticos de programas analisadores/tradutores (lex/flex, yacc/bison, ou javacc) ou mesmo implementando "a mão" o seu programa.

1. Faça um programa ***que analise arquivos no formato ARFF***, de acordo com o especificado a seguir. A linguagem ARFF (Attribute-Relation File Format) é uma linguagem textual adotada para descrever arquivos de dados usados pela ferramenta de Inteligência Artificial e Aprendizado de Máquina (Machine Learning) denominada de WEKA. A descrição dos arquivos ARFF usados pelo WEKA pode ser encontrada na Internet em:

   http://www.cs.waikato.ac.nz/~ml/weka/arff.html ou
   http://weka.sourceforge.net/wekadoc/index.php/en:ARFF_%283.4.6%29

   O professor irá fornecer os PDFs com esta descrição mais detalhada, assim como arquivos com exemplos de dados no formato ARFF, que terão que "passar" pelo seu analisador léxico/sintático.

   Os arquivos ARFF (formato simples, como o da versão 4.6 – indicado acima) possuem um cabeçalho do tipo:

```
% Linhas de comentário
% Começam por um "%" e vão até o final da linha
% Devem ser ignoradas
 %
@RELATION iris

@ATTRIBUTE sepallength  NUMERIC
@ATTRIBUTE sepalwidth   NUMERIC
@ATTRIBUTE petallength  NUMERIC
@ATTRIBUTE petalwidth   NUMERIC
@ATTRIBUTE class        {Iris-setosa,Iris-versicolor,Iris-virginica}
```

   Onde este cabeçalho é seguido de uma sessão de dados:

```
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.1,2.6,5.6,1.4,Iris-virginica
```

   Note a relação de cada "**ATTRIBUTE**" com a respectiva coluna (em ordem) que aparece na sessão "**DATA**", assim como o seu conteúdo (numérico, string, constantes), conforme descrição do cabeçalho. No exemplo acima, temos 4 valores numéricos seguidos de um valor qualitativo (classe).

A análise léxica deve considerar o léxico básico dos arquivos ARFF conforme a documentação. A análise sintática também deve considerar a estrutura básica destes arquivos, sendo que o analisador deverá funcionar para os arquivos exemplos fornecidos pelo professor.

Verificações a serem feitas:

* Na sessão "data" devem ser validados os seguintes itens:
  - Quantidade de itens (separados por vírgulas) deve bater com a quantidade de "attributes" declarados;
  - O tipo do atributo no "data" deve bater com o tipo declarado no cabeçalho;
  - Os atributos do tipo qualitativos devem ter seus valores verificados (valor válido);

* O máximo de atributos que o programa deverá considerar será de 20 atributos.

* O máximo de valores qualitativos diferentes aceitos em um dado atributo é de 5 valores. Exemplo: @ATTRIBUTE class {not_recom, recommend, very_recom, priority, spec_prior}

Saída do analisador:

* Quantidade total e atributos de entrada
* Quantidade total de atributos numéricos de entrada
* Quantidade total de atributos qualitativos (nominais) de entrada
* Existência de um atributo de saída explícito (classe = classificador) ou não (nro. = regressão)
* Número total de exemplos disponíveis no arquivo ("data")

Tradução: Gerar uma saída TODA com valores numéricos, ou seja, com os atributos qualitativos convertidos para o valor de seu índice na seqüência da enumeração dos valores que pode assimir (exemplo: seg=1, ter=2, qua=3, ...). As strings devem ser substituídas pelo valor "0.0". Os dados devem ser gerados na saída separados por espaços em branco (no lugar das vírgulas).

A solução desta questão deverá ser entregue em arquivos denominados pgc.l e pgc.y

Bom trabalho!

Atenção:

- ⇨ Caso você não consiga seguir exatamente o que foi proposto, indique claramente o que você conseguiu fazer e quais as limitações de sua implementação, ressaltando limitações adicionais que por ventura sejam impostas as definições indicadas nesta prova pelo professor.
  - ⇨ Primeiro tente fazer o analisador LÉXICO;
  - ⇨ Depois tente fazer o analisador SINTÁTICO;
  - ⇨ Faça a criação da tabela de símbolos e a análise semântica (validação dos atributos);
  - ⇨ Por último faça a geração da saída com a exibição das estatísticas finais.
- ⇨ Lembre-se de colocar o seu nome e o número de matrícula como comentários nas primeiras linhas do programa fonte, e indicações sobre: compilador/gerador de analisadores usados e dicas de como compilar seu programa e obter o executável. Lembre-se também que você deve entregar para o professor a folha da prova identificada com a sua assinatura no cabeçalho.
- ⇨ Envie os programas fonte da prova e os exemplos de arquivos a serem usados com seu programa por e-mail (webmail), comprimidos (.zip/tar.gz) para o professor em *attach* de uma mensagem. Use o seguinte título na mensagem: "*Subject*: Prova GA Tutoria".
  E-mail: fosorio@unisinos.br com cópia (cc:) também para fosorio@gmail.com
- ⇨ **NÃO ADICIONE O EXECUTÁVEL AO ARQUIVO ZIP - ENVIE EM UM EMAIL SEPARADO SE FOR O CASO!!! O ARQUIVO ZIP DEVERÁ CONTER APENAS ARQUIVOS TEXTO!!!**
- ⇨ Todo uso indevido de *correio eletrônico* (envio/recepção de mensagens) e/ou troca indevida de arquivos durante a prova que for registrado, poderá anular a sua prova. Usar o mail somente para o **envio final** dos programas **ao final da prova**.
- ⇨ Prova INDIVIDUAL e COM CONSULTA AO MATERIAL PESSOAL (não é permitido emprestar material ao colega). Prova com duração até o final do período.