UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO (UFERSA)
CENTRO DE CIÊNCIAS EXATAS E NATURAIS (CCEN)
DEPARTAMENTO DE COMPUTAÇÃO
DISCIPLINA: COMPILADORES
ATIVIDADE: TRABALHO PRÁTICO SOBRE ANÁLISE SINTÁTICA

OBJETIVO: Construir um analisador sintático para verificar formas adequadas de se descrever classes de uma ontologia em OWL Manchester Syntax.

DESCRIÇÃO: A Web Ontology Language (OWL) é uma linguagem declarativa para especificação de ontologias - grafos conceituais que conectam conceitos interrelacionados de um determinado domínio de conhecimento. Ontologias são compostas de classes, propriedades que ligam uma classe à outra (*object properties*), propriedades que ligam uma classe a um tipo de dado (*data properties*), restrições (cardinalidades e quantificadores) e indivíduos (equivalentes às instâncias em Programação Orientada a Objetos). Há várias formas de se declarar classes em OWL:

1. **Classe primitiva:** é uma classe cujos indivíduos podem herdar suas propriedades, mas indivíduos avulsos que tenham tais propriedades não podem ser classificados como membros dessas classes. No exemplo a seguir, é possível notar que a declaração deste tipo de classe inclui as definições de propriedades abaixo do axioma "**SubClassOf**", ou seja, todos os indivíduos da classe primitiva **Pizza** serão também membros de tudo o que tem alguma base (**PizzaBase**) e tudo o que tem conteúdo calórico de algum valor inteiro. Todas as classes podem conter as seções "**DisjointClasses**" e "**Individuals**" em suas descrições.

   Class: Pizza

   SubClassOf:
   hasBase some PizzaBase,
   hasCaloricContent some xsd:integer

   DisjointClasses:
   Pizza, PizzaBase, PizzaTopping

   Individuals:
   CustomPizza1,
   CustomPizza2

2. **Classe definida:** é uma classe que contém condições necessárias e suficientes em sua descrição. Em outros termos, esse tipo de classe não somente transfere suas características para seus indivíduos por herança, mas também permite inferir que indivíduos avulsos que tenham suas propriedades possam ser classificados como membros dessas classes. Uma classe definida normalmente tem uma seção definida pelo axioma "**EquivalentTo**:" sucedido por um conjunto de descrições. No exemplo abaixo, as classes **CheesyPizza** e **HighCaloriePizza** são definidas dessa forma.

Class: CheesyPizza

EquivalentTo:
Pizza and (hasTopping some CheeseTopping)

Individuals:
CheesyPizza1

Class: HighCaloriePizza

EquivalentTo:
Pizza and (hasCaloricContent some xsd:integer[>= 400])

3. **Classe com axioma de fechamento (closure axiom)**: uma classe com axioma de fechamento contém normalmente uma cláusula que "fecha" ou restringe as imagens de suas relações a um conjunto bem definido de classes ou de expressões. No exemplo abaixo, note que a classe **MargheritaPizza**, como domínio da relação ou "tripla RDF" pode estar conectada a duas outras classes de imagem (**MozzarellaTopping** e **TomatoTopping**) mediante a propriedade **hasTopping**. Entretanto, é necessário tornar explícito para o reasoner (motor de inferência lógica) que esse tipo de pizza só pode ter esses dois tipos de cobertura. Por isso, a expressão "**hasTopping only** (**MozzarellaTopping or TomatoTopping**)" é considerada um "fechamento" da imagem da relação/propriedade hasTopping, que é usada para descrever as relações que **MargheritaPizza** tem com possíveis coberturas (**Toppings**).

   Class: MargheritaPizza

   SubClassOf:
   NamedPizza,
   hasTopping some MozzarellaTopping,
   hasTopping some TomatoTopping,
   hasTopping only (MozzarellaTopping or TomatoTopping)

4. **Classe com descrições aninhadas:** é possível que a imagem de uma propriedade que descreve uma classe não seja necessariamente uma outra classe, mas outra tripla composta de propriedade, quantificador e outra classe. Esses aninhamentos criam estruturas semelhantes a orações coordenadas ou subordinadas, como estudamos na gramática da Língua Portuguesa, por exemplo. Observe como a expressão "**hasSpiciness value Hot**" compreende a imagem (ou objeto) da propriedade "**hasTopping**".

   Class: SpicyPizza

   EquivalentTo:
   Pizza
   and (hasTopping some (hasSpiciness value Hot))

É possível definir classes com aninhamentos de propriedades de forma bem mais complexa, conforme o exemplo abaixo:

Listagem 9 – Representação da Lógica de Descrição da Classe *Regulatory Activity*

```
Class: RegulatoryActivity
   EquivalentTo: ValueActivity
      and ((bundles some
            (CnAObject or CoreObject or PoPObject)) or
               (consumes some CounterObject))
      and ((grants some CnAObject) or
            (transfers some
               (CoreObject or PoPObject)))
      and (isAuthorityOf some Regulator)
      and (hasTransaction some ValueTransaction)
      and (bundles only
            (CnaObject or CoreObject or PoPObject))
      and (consumes only CounterObject)
      and (grants only CnAObject)
      and (isAuthorityOf only Regulator)
      and (transfers only
            (CoreObject or PoPObject))
```

5. **Classe enumerada:** algumas classes podem ser definidas a partir de uma enumeração de suas possíveis instâncias. Por exemplo, uma classe denominada “dias_da_semana” poderia conter apenas sete instâncias, sendo uma para cada dia. Uma outra classe denominada “planetas_do_sistema_solar” poderia conter apenas oito instâncias. Em OWL, há uma forma de se especificar esse tipo de classe. Note que, no exemplo abaixo, a classe **Spiciness** é definida com um conjunto de instâncias (**Hot**, **Medium** e Mild), que aparecem entre chaves.

   Class: Spiciness

   EquivalentTo: {Hot , Medium , Mild}

6. **Classe coberta:** uma variação do exemplo anterior consiste em especificar uma classe como sendo a superposição de suas classes filhas. Ou seja, neste caso teríamos a classe Spiciness como sendo a classe mãe e as classes Hot, Mild e Mild como sendo classes filhas. A implicação lógica nesse caso é de que qualquer indivíduo pertencente à classe mãe precisa também estar dentro de alguma classe filha. Esse tipo de classe poderia ser especificada da seguinte forma:

   Class: Spiciness

   EquivalentTo: Hot or Medium or Mild

DESAFIO: Especificar um analisador sintático somente para a análise de declarações de classes na linguagem OWL, de acordo com o formato Manchester Syntax, de acordo com as seguintes diretrizes:

- Os exemplos acima dão noções de como uma classe pode ser declarada;
- Elaborar uma gramática livre de contexto, sem ambiguidade, fatorada e sem recursividade à esquerda;
- Construir uma tabela de análise preditiva (para implementação manual) ou utilizar um gerador de analisador sintático com base em análise ascendente;
- Simular a leitura de uma especificação de classe como entrada para verificação da consistência da declaração da classe;
- Por "consistência", entende-se que a declaração das classes seguem uma ordem que varia de acordo com o tipo de classe (p.ex.: separação dos nomes das classes por vírgulas nas classes enumeradas, disjunções entre as subclasses de uma classe coberta, separação de cláusulas pela palavra-chave **AND** ou por espaçamento, no caso das classes definidas e primitivas, respectivamente).

FERRAMENTAS:

- Especificação manual usando gramáticas livres de contexto, fatoração, eliminação da recursividade, redução da ambiguidade, especificação de tabela de análise preditiva e simulação de exemplos de entrada;
- Ambientes de desenvolvimento integrado, p.ex.: VS Code;
- Ambientes específicos de modelagem para compiladores, p.ex.: ANTLR ou YASH.

OBSERVAÇÕES IMPORTANTES:

- ⅓ da turma deverá apresentar o trabalho;
- Para os que irão apresentar, 60% da nota corresponde à implementação, enquanto que 40% equivalem à apresentação;
- Poderá ser realizado individualmente ou em grupos de até 2 componentes;
- Data de entrega: 21/03/2024, até às 23h59.

REFERÊNCIAS:

Horridge, M., Drummond, N., Goodwin, J., Rector, A. L., Stevens, R., & Wang, H. (2006, November). The Manchester OWL syntax. In *OWLed* (Vol. 216). Disponível online em: https://ceur-ws.org/Vol-216/submission_9.pdf

Protégé Ontology Editor, disponível para download em: https://protege.stanford.edu/

Pizza Ontology, disponível online em: https://protege.stanford.edu/ontologies/pizza/pizza.owl