Ficha de Trabalho nº 6

Sistema Pericial: Diagnóstico médico

Sintaxe das regras DRL

1. Implementação do SP em Drools - diagnóstico - versão 1

Neste exercício vamos implementar um Sistema Pericial que, dado um conjunto de sintomas associados a um conjunto de pacientes, consiga identificar o diagnóstico correto.

1.1 Crie o Projeto Drools

- Inicie o Eclipse e faça: `File – New Project – Drools Project`
- Atribua o nome ao projeto (Diagnostico1)

1.2 Crie duas Classes:

- **Nome da Classe:** Paciente
- **Campos:** id (String), nome (String), idade (int), diagnostico (String);
- Acrescente o construtor; getters e setters

- **Nome da Classe:** Sintoma
- **Campos**: idPaciente (String), sintoma (String);
- Acrescente o construtor; getters e setters

As duas classes estão ligadas através dos campos **id** e **idPaciente**. Desta forma saberemos que sintomas pertencem a um Paciente específico.

1.3 Insira factos na memória de trabalho

Na função DroolsTest.java insira os seguintes factos. O campo *diagnostico* fica vazio e será preenchido pelas regras quando for encontrado o diagnóstico apropriado.:

Paciente P1: ("001","Ana Melo",12,"");
Paciente P2: ("002","Rui Costa",13,"");
Paciente P3: ("003","Joana Martins",85,"");
Paciente P4: ("004","Pedro Torres",53,"");
Paciente P5: ("005","Ana Gomes",93,"");

Sintoma S1: ("001", "febre");
Sintoma S2: ("001", "manchas");
Sintoma S3: ("002", "febre");
Sintoma S4: ("003", "febre");
Sintoma S5: ("003", "dores");
Sintoma S6: ("004", "febre");
Sintoma S7: ("004", "dores");
Sintoma S8: ("004", "manchas");

Desta informação podemos ver que os sintomas S1 e S2 pertencem ao paciente P1, o sintoma S3 ao paciente P2, os sintomas S4 e S5 ao paciente P3 e os sintomas S6, S7 e S8 ao Paciente P4. O Paciente P5 não tem sintomas associados.

1.4 Implemente as seguintes regras

a) Regra 1: "exantema 1"

Se um determinado Paciente com 15 ou mais anos tiver **febre, manchas** e **dores**, atualizar o seu diagnóstico (atributo *diagnostico*) para "exantema 1" usando o método "setDiagnostico". Imprimir também o diagnóstico no componente RHS (**then**) da regra. ***>> O paciente <<nome>> foi diagnosticado com Exantema 1.***

Teste esta regra: deverá funcionar bem.

```
>>  O Paciente Pedro Torres foi diagnosticado com Exantema 1
```

Agora, atualize também o novo diagnóstico na memória de trabalho do Drools, usando: ***update($p);***
em que a variável $p é a variável que usou para carregar as instâncias da classe Paciente

Teste esta regra: deve imprimir "***O Paciente Pedro Torres foi diagnosticado com Exantema 1***".
Para evitar que este paciente venha a receber outros diagnósticos de doenças com menos sintomas, acrescente no LHS da regra um teste ao valor do diagnóstico, como por exemplo *&& diagnostico==""*.

b) Seguindo processo análogo, crie mais as seguintes regras:

Regra 2: "exantema 2"
Se um determinado Paciente com menos de 15 anos tiver **febre, manchas** actualizar e imprimir o seu diagnóstico para "exantema 2"

Regra 3: "gripe"
Se um determinado Paciente tiver **febre** e **dores** actualizar e imprimir o seu diagnóstico como "gripe"

Regra 4: "resfriado"
Se um determinado Paciente apenas apresentar **febre,** actualizar e imprimir o seu diagnóstico como "resfriado"

c)

Regra 5: "não tem sintomas"
Se para um determinado Paciente não existirem sintomas na memória de trabalho, actualizar e imprimir o seu diagnóstico como "não tem sintomas"

Para testar esta condição, tem de utilizar o comando ***forall***, de forma análoga à seguinte:

```
$p:Paciente(diagnostico=="")
 forall($s:Sintoma(idPaciente!=$p.getId()))
```

Teste o sistema pericial, deve obter o seguinte output:

```
O Paciente Pedro Torres foi diagnosticado com Exantema 1
O Paciente Ana Melo foi diagnosticado com Exantema 2
O Paciente Joana Martins foi diagnosticado com Gripe
O Paciente Rui Costa foi diagnosticado com Resfriado
O Paciente Ana Gomes não tem sintomas associados
```

d) Repare que as regras criadas devem disparar pela ordem com que foram criadas, isto é, com maior prioridade para as mais restritivas: os doentes que apresentarem mais sintomas devem ser diagnosticados primeiro, pois caso contrário, por exemplo, "febre" + "dores" + "manchas" poderia disparar a regra "*resfriado*" (que só exige "febre") e depois disso - porque o diagnóstico passaria a ser diferente de "" - nunca mais poderia ser atualizado para o diagnóstico correto, "*exantema 1*".

Estas situações ocorrem sempre que a cobertura das regras se sobrepõe, isto é, quando uma ou algumas regras disparam com exemplos que fazem também disparar outras, mais restritivas. Para o evitar, a cada regra tem de se atribuir a prioridade, através do comando **salience**. A maior prioridade é indicada por **salience 100**, e a menor por **salience 0**.

Atribua as prioridades adequadas às regras criadas. Experimente alterar a sua ordem e teste o programa. Os resultados deverão ser sempre os mesmos, e corretos.

e) Crie mais uma regra para imprimir o ID, nome e diagnóstico de cada paciente. Naturalmente que esta regra só deve disparar depois de todas as anteriores.
Coloque em comentário todas as impressões executadas pelas regras anteriores. Teste o programa. Deverá funcionar corretamente.

2. Implementação do SP em Drools - diagnóstico - versão 2

Vamos agora resolver o problema da atribuição do diagnóstico de outra forma. A ideia consiste em criar um nova classe na qual são inseridas as associações entre pacientes e respetivos diagnósticos, e no final imprimir o conteúdo desta classe. Desta forma é possível:

1) evitar o update da classe Paciente
2) remover da memória de factos os pacientes à medida que forem diagnosticados, evitando assim o disparo das regras de menor prioridade e menos restritivas.

Nota: Para remover factos usa-se o comando **retract**.

a) Crie um novo projeto de nome Diagnostico_V2. Crie agora **3** Classes:

- **Nome da Classe:** Paciente
- **Campos**: id (String), nome (String), idade (int);
- **Acrescente** o construtor; getters e setters

- **Nome da Classe:** Sintoma
- **Campos:** idPaciente (String), sintoma (String);
- **Acrescente** o construtor; getters e setters

- **Nome da Classe:** Diagnostico
- Campos: idPaciente (String), nomePaciente (String), designacao (String);
- **Acrescente** o construtor, getters e setters

b) Reescreva todas as regras anteriores, com as seguintes modificações:

i) Retire o teste a `diagnostico ==""` na condição LHS das regras
ii) Retire as instruções da componente RHS da regras

```
$p.setDiagnostico("...");
update($p);
```

iii) No RHS (conclusão da regra) acrescente a inserção do diagnóstico respetivo na classe **Diagnostico**. Para isso utilize o comando **insert**, de forma análoga ao seguinte:

```
insert (new Diagnostico($p.getId(), $p.getNome(), "Exantema 1"));
```

em que $p é a variável usada para obter a lista de pacientes no LHS da regra;

iv) Em cada regra, e depois deste ***insert***, remova da lista de factos os pacientes já diagnosticados, com o comando ***retract***, de forma análoga ao seguinte:

```
retract ($p);
```

em que $p é a variável usada para obter a lista de pacientes no LHS da regra;

v) Modifique a regra de impressão para agora utilizar a classe **Diagnostico** (em vez de Paciente) para imprimir os resultados.

vi) Insira na memoria de trabalho os mesmos factos do exemplo anterior|:

```
Paciente p1 = new Paciente("001","Ana Melo",12);
Paciente p2 = new Paciente("002","Rui Costa",13);
Paciente p3 = new Paciente("003","Joana Martins",85);
Paciente p4 = new Paciente("004","Pedro Torres",53);
Paciente p5 = new Paciente("005","Ana Gomes",93);

Sintoma s1 = new Sintoma("001", "febre");
Sintoma s2 = new Sintoma("001", "manchas");
Sintoma s3 = new Sintoma("002", "febre");
Sintoma s4 = new Sintoma("003", "febre");
Sintoma s5 = new Sintoma("003", "dores");
Sintoma s6 = new Sintoma("004", "febre");
Sintoma s7 = new Sintoma("004", "dores");
Sintoma s8 = new Sintoma("004", "manchas");
kSession.insert(p1);
....
```

vii) Teste o programa. Deverá funcionar corretamente.

```
Paciente com id 005 Ana Gomes com diagnostico: sem sintomas definido
Paciente com id 002 Rui Costa com diagnostico: Resfriado
Paciente com id 003 Joana Martins com diagnostico: Gripe
Paciente com id 001 Ana Melo com diagnostico: Exantema 2
Paciente com id 004 Pedro Torres com diagnostico: Exantema 1
```

viii) Teste o programa utilizando mais pacientes e factos. Para isso utilize também uma regra para os inserir na memória de factos, de forma análoga ao seguinte:

```
rule "Inserir mais factos"
salience 110
when
then
        insert (new Paciente("006","Carlos Lopes",22));
        insert (new Paciente("007","Maria Ferreira",25));
        insert (new Paciente("008","Nuno Gomes",60));
        insert (new Paciente("009","Julio Lopes",58));
        insert (new Paciente("010","Rui Saraiva",45));
        insert (new Sintoma("006", "febre"));
        insert (new Sintoma("006", "dores"));
        insert (new Sintoma("006", "manchas"));
        insert (new Sintoma("007", "febre"));
        insert (new Sintoma("008", "dores"));
        insert (new Sintoma("010", "manchas"));
end
```

ix) Nos *inserts* de viii) repare que os pacientes 008 e 010 têm sintomas que não correspondem a nenhuma doença. Uma hipótese de contemplar esta situação consiste em acrescentar a regra

```
rule "com sintomas mas não correspondem a nenhum diagnóstico"
salience 10
when
      $p1:Paciente()
then
      insert (new Diagnostico($p1.getId(), $p1.getNome(), "Com sintomas, mas
não correspondem a nenhum diagnóstico"));
      retract ($p1)
end
```

Ela deve cobrir apenas os doentes **que restam** depois de todos terem sido removidos através dos diversos retracts das regras anteriores, e por isso a sua *salience* deve ser inferior à das outras regras de diagnóstico (no exemplo, *salience = 10*, restantes regras *salience=50* ou mais)

Incluindo esta regra os resultados devem ser:

- `Paciente com id 008 Nuno Gomes com diagnostico: Com sintomas, mas não correspondem a nenhum diagnóstico`
- `Paciente com id 010 Rui Saraiva com diagnostico: Com sintomas, mas não correspondem a nenhum diagnóstico`
- `Paciente com id 009 Julio Lopes com diagnostico: sem sintomas definido`
- `Paciente com id 005 Ana Gomes com diagnostico: sem sintomas definido`
- `Paciente com id 007 Maria Ferreira com diagnostico: Resfriado`
- `Paciente com id 002 Rui Costa com diagnostico: Resfriado`
- `Paciente com id 003 Joana Martins com diagnostico: Gripe`
- `Paciente com id 001 Ana Melo com diagnostico: Exantema 2`
- `Paciente com id 006 Carlos Lopes com diagnostico: Exantema 1`
- `Paciente com id 004 Pedro Torres com diagnostico: Exantema 1`

3. Implementação do SP em Drools - diagnóstico. versão 3

Vamos agora implementar a versão final do programa, com a possibilidade de recolher factos indicados pelo utilizador, realizando em seguida o diagnóstico de um paciente único. Para isso:

a) Crie um novo projeto de nome Diagnostico_v3, contendo as 3 classes usadas na versão anterior.

b) Mantenha as regras de diagnóstico usadas na versão anterior, à exceção da regra "inserir mais factos", criada em **B-viii**), que deve ser eliminada.

c) Para recolher sintomas dados pelo utilizador, utilize uma regra com sintaxes do tipo:

```
rule "Entrada de Dados"
salience 110
    when

    then
        System.out.println("Nome do Paciente ?");
        Scanner input1 = new Scanner(System.in);
        String Nome = input1.nextLine();
        String r1 ;

        System.out.println("Idade ?");
        Scanner input2 = new Scanner(System.in);
        int Idade = input2.nextInt();

        insert (new Paciente("001", Nome, Idade));

        System.out.println("Tem dores ?");
        r1 = input1.nextLine();

             if (r1.equals("S")) {
                    insert (new Sintoma("001","dores"));
                    }

        System.out.println("Tem manchas ?");
        r1 = input1.nextLine();

             ... // COMPLETAR
end
```

Acrescente a instrução: `import java.util.Scanner;`

Sugestão: depois de criar esta regra, poderá criar outra para efeitos de debugg, destinada apenas a imprimir todos os factos conhecidos e verificar se foram correctamente inseridos na memória de trabalho do Drools

d) Na aplicação Drools.Test.Java remova todas as inserções de factos, mantendo apenas a activação das regras, kSession.fireAllRules();

e) Teste o programa, não esquecendo as situações em que a combinação de sintomas não corresponde a nenhum diagnóstico, e em que o paciente não tem nenhum sintoma.