Disciplina: Paradigmas de Programação
Professor: Maicon Rafael Zatelli
Entrega: *Moodle*

Atividade I - Prolog

Atenção: Faça um ZIP com todos os arquivos de solução. Use o nome do arquivo de maneira a entender qual problema você está resolvendo. Por exemplo, problema1.pl, problema2.pl e assim por diante. Faça consultas para testar seu programa. **Inclua no seu código fonte as consultas realizadas e o resultado obtido (em comentário)**. Não use funções prontas do Prolog para resolver os exercícios, exceto se necessário.

Resolva os seguintes problemas na linguagem Prolog:

1. Modifique o arquivo `familia.pl` (disponível no Moodle) de forma a incluir as seguintes regras:
   - `tio(X,Y)`, onde X é o tio de Y.
   - `tia(X,Y)`, onde X é a tia de Y.
   - `primo(X,Y)`, onde X é o primo de Y.
   - `prima(X,Y)`, onde X é a prima de Y.
   - `primos(X,Y)`, onde X é primo ou prima de Y.
   - `bisavo(X,Y)`, onde X é o bisavô de Y.
   - `bisavoh(X,Y)`, onde X é a bisavó de Y.
   - `descendente(X,Y)`, onde X é descendente de Y.
   - `feliz(X)`, onde X é feliz se possui filhos.
   - Faça consultas para testar seu programa. Inclua no seu código fonte as consultas realizadas e o resultado obtido (em comentário).
   - Crie outra regra a sua escolha, descreva seu funcionamento e exemplifique seu uso por meio de uma consulta.

2. Crie um novo programa Prolog para o seguinte cenário.
   - Crie fatos para representar as sentenças abaixo:
     - um estudante estuda uma disciplina, sendo que uma disciplina é formada por 3 letras e 4 dígitos (ex: ine5416).
     - um professor leciona uma disciplina.
   - Crie regras que representem as sentenças abaixo:
     - Um professor ensina um aluno se o professor leciona uma disciplina em que o aluno estuda.
     - Dois estudantes são colegas de classe se estudam a mesma disciplina e não são o mesmo aluno.
     - Crie consultas para as seguintes perguntas abaixo:
       * Quais são todas as disciplinas lecionadas pelo professor x? (substitua x por algum nome de professor no seu programa)
       * Quais são todos os alunos do professor x? (substitua x por algum nome de professor no seu programa)
       * Quais são todos os amigos do estudante y? (substitua y por algum nome de aluno no seu programa)
       * a e b são amigos? (substitua a e b por nomes de alunos no seu programa)
   - Faça outras consultas para testar seu programa. Inclua no seu código fonte as consultas realizadas e o resultado obtido (em comentário).
   - Crie outra regra a sua escolha, descreva seu funcionamento e exemplifique seu uso por meio de uma consulta.

ATENÇÃO: Para as questões abaixo, faça consultas para testar cada regra criada. Inclua no seu código fonte as consultas realizadas e o resultado obtido (em comentário).

3. Crie uma regra `divisivel(N,K)` para dizer se um número N é divisível por K.

4. Crie uma regra `triangulo(X,Y,Z)` que receba três valores X, Y e Z e indique se havendo varetas com esses valores em comprimento pode-se construir um triângulo. Exemplo, com varetas de comprimento 4, 8 e 9 posso construir um triângulo, porém com varetas de comprimento 10, 5 e 4 não posso construir um triângulo.

5. Crie uma regra `eqSegundoGrau(A,B,C,ValorX)` que resolva uma equação de segundo grau da forma $ax^2 + bx + c$ utilizando a fórmula de Bhaskara.

6. Crie uma regra `potencia(X,Y,Resultado)`, onde *Resultado* é X^Y.

7. Crie uma regra `absoluto(N,X)` que receba um número N, negativo ou positivo, e retorne seu valor absoluto X.

8. Crie uma regra `areaTriangulo(B,A,Area)` que receba a base e a altura de um triângulo e calcule a área do mesmo.

9. Crie uma regra `xor(X,Y)` que receba dois valores booleanos X e Y e indique se a operação X xor Y é verdadeira. Construa a regra apenas usando os operadores **,** (and), **;** (or) e **not**.

10. Crie uma regra `aprovado(A,B,C)` que receba três notas de um aluno (A, B, C), calcule a média e indique se o aluno foi aprovado ou reprovado. Para um aluno ser aprovado, ele deve possuir nota igual ou superior a 6.

11. Crie uma regra `fib(N,K)` que compute o N-ésimo número de Fibonacci K.

12. Crie uma regra distancia3D(ponto(X1,Y1,Z1),ponto(X2,Y2,Z2), Dist) que dados dois pontos no espaço 3D, (x1, y1, z1) e (x2, y2, z2), compute a distância (Dist) entre eles.

13. Crie uma regra `maior(A,B,C,X)` que receba 3 valores numéricos (A, B, C) e retorne o maior deles (X).

14. Crie uma regra `operacao(Op,X,Y,Resultado)` que receba três parâmetros Operador, X e Y, e retorne o resultado da operação matemática X *Operador* Y. Os operadores possíveis de informar são +, -, *, /.

15. Crie uma regra mdc(X,Y,Resultado) que receba dois números X e Y e retorne o máximo divisor comum (DICA: pesquise sobre o Algoritmo de Euclides).

16. Crie uma regra mdc(X,Y,Z,Resultado) que receba três números X, Y e Z e retorne o máximo divisor comum (DICA: apenas modifique a regra anterior).

17. Crie uma regra `mmc(X,Y,Resultado)` que receba dois números X e Y e retorne o mínimo múltiplo comum (DICA: use a regra do máximo divisor comum já criada).

18. Crie uma regra `coprimos(X,Y)` que receba dois números X e Y e indique se eles são coprimos. Dois números são ditos coprimos se o máximo divisor comum entre eles é 1.

19. Crie uma regra `totienteEuler(N,K)` que receba um número n e retorne (em K) o resultado da função totiente de Euler ($\phi(n)$). A função totiente de Euler é dada pelo número de inteiros positivos r a partir de 1 e menores que n, ou seja $1 <= r < n$, que são coprimos de n. Por exemplo, se $n = 10$, então os coprimos de 10 de 1 até 10-1 são $\{1, 3, 7, 9\}$ e a função deve retornar $\phi(n) = 4$. Além disso, $\phi(1) = 1$.

20. Crie uma regra `primo(N)` que receba um número N e retorne se o mesmo é primo.