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

Atividade II - 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. Crie uma regra `posicao(X,L,P)` que receba um elemento X, uma lista L e retorne a posição P do elemento X na lista L. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

2. Crie uma regra `inserirElementoPosicao(X,P,L1,L2)` que receba um elemento X, uma posição P, e uma lista L1 e retorne uma lista L2 onde X é inserido na posição P. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

3. Crie uma regra `numerosParaPalavras(L1,L2)` que receba uma lista L1 contendo os números de 0 até 9 e retorne uma lista L2 que contenha a mesma lista de números de 0 até 9, mas escritos como palavras, ou seja, zero, um, dois, tres (sem acento) e assim por diante. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

4. Crie uma regra `soma(L,X)`, a qual recebe uma lista L de inteiros e retorna a soma de todos os elementos X da lista. Retorne 0 caso a lista for vazia. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

5. Crie uma regra `media(L,X)`, a qual recebe uma lista L de inteiros e retorna a média de todos os elementos X da lista. Retorne 0 caso a lista for vazia. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

6. Crie uma regra `menor(L,X)`, a qual recebe uma lista L de inteiros e retorna o menor elemento X da lista. Retorne 0 caso a lista for vazia. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

7. Crie uma regra `palindrome(L)`, a qual recebe uma lista L e retorna se ela é uma palíndrome. Uma lista é uma palíndrome se os itens da esquerda para a direita estão na mesma ordem da direita para a esquerda. Ex: [1,2,3,4,3,2,1] é uma palíndrome. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

8. Crie uma regra `diferencaMaiorMenor(L,X)`, a qual recebe uma lista L de inteiros e retorna a diferença entre o maior e o menor elemento X da lista. Retorne 0 caso a lista for vazia. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

9. Crie uma regra `ocorrencias(L,X,N)`, a qual recebe uma lista L, um elemento X e retorna o número de vezes N em que o elemento está presente na lista. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

10. Crie uma regra `inverte(L1,L2)`, a qual recebe uma lista L1 como parâmetro e deve retornar a mesma invertida L2. **Não utilize** nenhuma função pronta do Prolog para realizar esta tarefa.

11. Crie uma regra `primeiros(N,L1,L2)`, a qual recebe um número de elementos N, uma lista L1, e retorna uma lista L2. Esta função deve retornar uma lista com os N primeiros elementos informados no primeiro parâmetro. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

12. Crie uma regra `apagar(N,L1,L2)`, a qual recebe, um número de elementos N, uma lista L1, e retorna uma lista L2. Esta função deve remover da lista os N primeiros elementos fornecidos como parâmetro. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

13. Crie uma regra `dividir(L1,L2,L3)`, a qual recebe uma lista como entrada L1 e deve dividí-la em duas listas L2 e L3 com a mesma quantidade de elementos (exceto quando L1 tiver quantidade ímpar). Por exemplo, dividir([1,2,3,4,5],L2,L3) deve retornar L2 = [1, 3, 5], L3 = [2, 4]. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

14. Crie uma regra `uniao(S1,S2,S3)`, a qual recebe dois conjuntos S1 e S2 e retorna em S3 a união de S1 e S2. Por exemplo, uniao([1,2,3,4],[1,2,5,6],S3) deve retornar S3 = [3, 4, 1, 2, 5, 6]. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

15. Crie uma regra `diferenca(S1,S2,S3)`, a qual recebe dois conjuntos S1 e S2 e retorna em S3 a diferença de S1 e S2. Por exemplo, diferenca([1,2,3,4],[1,2],S3) deve retornar S3 = [3, 4] e diferenca([1,2],[1,2,3,4],S3) deve retornar S3 = []. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.

16. Crie regras para as operações de intersecção, união e diferença, mas que agora realizem a operação com 3 conjuntos e retornem o resultado no quarto conjunto. **Não utilize** nenhuma função pronta to Prolog para esta tarefa.