Laboratório de Linguagens de Programação
Prof. Fabrício Rodrigues Inácio

Trabalho Prático IV

1. Objetivo

O objetivo desse trabalho prático é permitir que os alunos pratiquem os conceitos de programação lógica. Os alunos deverão desenvolver algumas aplicações usando a linguagem de programação Prolog (https://swi-prolog.org).

2. Instruções

Para cada um dos problemas a seguir, desenvolva um programa em Prolog que os resolva. Não se pode usar nenhuma função pré-definidas da biblioteca de Prolog ou qualquer outra de terceiros a não ser aquelas vistas em aula (nos slides), mas é permitido reimplementar alguma função auxiliar caso julgue necessário.

1) Obter o número de elementos de uma lista.

```
Ex.: ?- nelementos([1, 2, 3], S).
     S = 3 .
```

2) Obter o maior valor de uma lista de inteiros.

```
Ex.: ?- maior([4, 5, 2, 3, 1], M).
     M = 5 .
```

3) Obter o valor médio de uma lista de inteiros.

```
Ex.: ?- medio([4, 5, 2, 3, 1], M).
     M = 3.0 .
```

4) Inserir um elemento no fim da lista.

```
Ex.: ?- inserirfim(4, [1, 2, 3], L).
     L = [1,2,3,4] .
```

5) Obter o último elemento de uma lista.

```
Ex.: ?- ultimo([1, 2, 3, 4], U).
     U = 4 .
```

6) Verificar se um elemento X é adjacente a um elemento Y.

```
Ex.: ?- adjacente(3, 4, [1, 2, 3, 4, 5, 6]).
     True.
```

7) Gerar uma lista com os elementos de uma faixa (inclusive).

```
Ex.: ?- gerar(5, 10, L).
     L = [5, 6, 7, 8, 9, 10] .
```

8) Reverter uma lista. Dica: use o predicado concatenar.

```
Ex.: ?- reverter([1, 2, 3], L).
     L = [3, 2, 1] .
```

9) Incrementar em uma unidade cada elemento de uma lista de inteiros.

```
Ex.: ?- incrementar([5, 6, 7, 8], L).
     L = [6, 7, 8, 9] .
```

10) Linearizar uma lista de inteiros. Dica: use o predicado concatenar.

```
Ex.: ?- linearizar([[1,2,3], [4,5], [6], [7,8]], L).
         L = [1, 2, 3, 4, 5, 6, 7, 8] .
```

11) Anexar uma lista em outra.

```
Ex.: ?- anexar([1, 2, 3], [4, 5, 6], L).
     L = [1, 2, 3, 4, 5, 6] .
```

12) Remover de uma lista um elemento (todas as suas ocorrências).

```
Ex.: ?- remover(3, [1,3,2,3,4], L).
     L = [1, 2, 4] .
```

13) Rotacionar uma lista uma posição.

```
Ex.: ?- rotacionar([1, 2, 3, 4, 5], L).
     L = [2, 3, 4, 5, 1] .
```

14) Rotacionar uma lista *n* posições.

```
Ex.: ?- rotacionarn(2, [1, 2, 3, 4, 5], L).
     L = [3, 4, 5, 1, 2] .
```

15) Ordenar uma lista de inteiros.

```
Ex.: ?- ordenar([3, 1, 2], L).
     L = [1, 2, 3] .
```

Cada programa deve estar em um arquivo com o nome da função do item seguido de *.pl*. Por exemplo, para o primeiro item o arquivo deve ser nomeado *nelementos.pl*, enquanto para o segundo *maior.pl* e assim por diante. **Se não estiver no padrão, o programa será desconsiderado.**

3. Avaliação

O trabalho deve ser feito em grupo de até dois alunos, sendo esse limite superior estrito. O trabalho será avaliado em 15 pontos, onde essa nota será multiplicada por um fator entre 0.0 e 1.0 para compor a nota de cada aluno individualmente. Esse fator poderá estar condicionado a apresentações presenciais a critério do professor.

Trabalhos copiados, parcialmente ou integralmente, serão avaliados com nota **NEGATIVA** (nesse caso -15 pontos), sem direito a contestação. Você é responsável pela segurança de seu código, não podendo alegar que outro grupo o utilizou sem o seu consentimento.