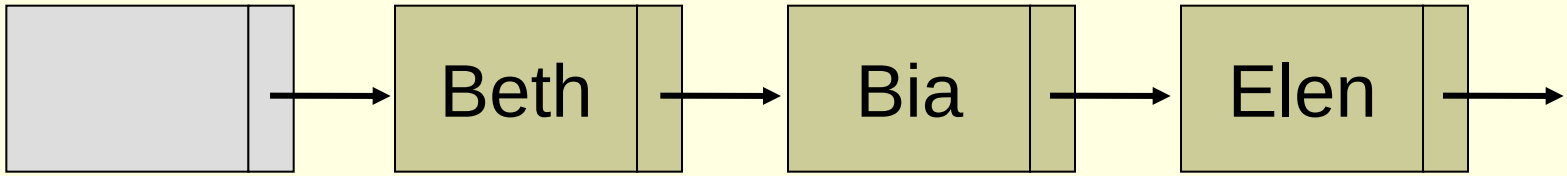
Outras Listas

SCC-202 – Algoritmos e Estruturas de Dados I

Lucas Antiqueira

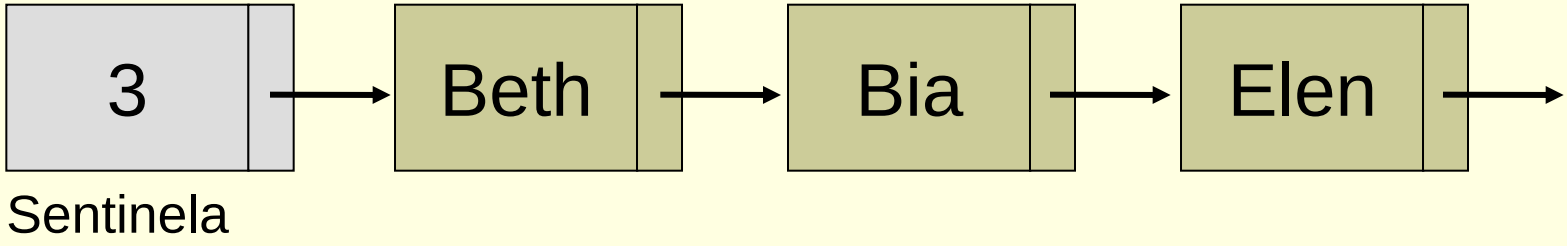
Lista com nó de cabeçalho

- Nó de cabeçalho
  - *Header*, sentinela, etc.
- Para que serve?



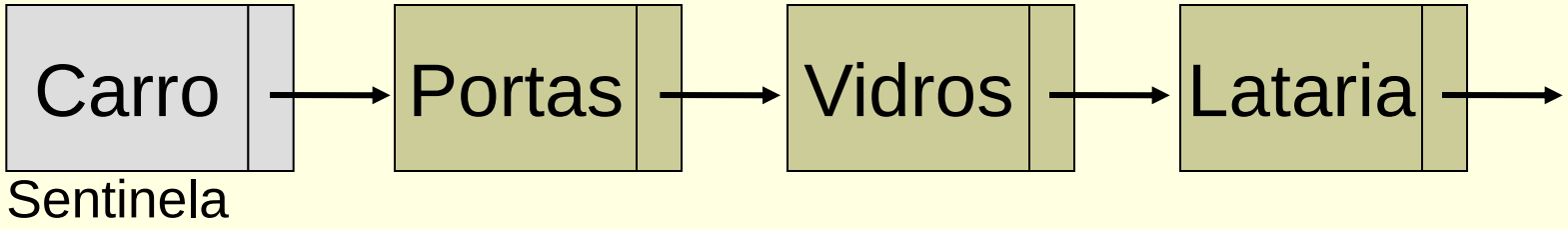
Lista com nó de cabeçalho

- Possibilidades de uso
  - Informação global sobre a lista que possa ser necessária na aplicação
    - P. ex.: Armazenar número de elementos da lista, para que não seja necessário atravessá-la contando seus elementos



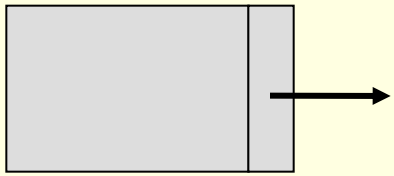
Lista com nó de cabeçalho

- Possibilidades de uso
  - Informação global sobre a lista que possa ser necessária na aplicação
    - Em uma fábrica, guarda-se as peças que compõem cada equipamento produzido, sendo este indicado pelo nó sentinela
    - Informações do vôo correspondente a uma fila de passageiros



Lista com nó de cabeçalho

- Possibilidades de uso
  - Informação global sobre a lista que possa ser necessária na aplicação
    - Lista vazia contém somente o nó sentinela



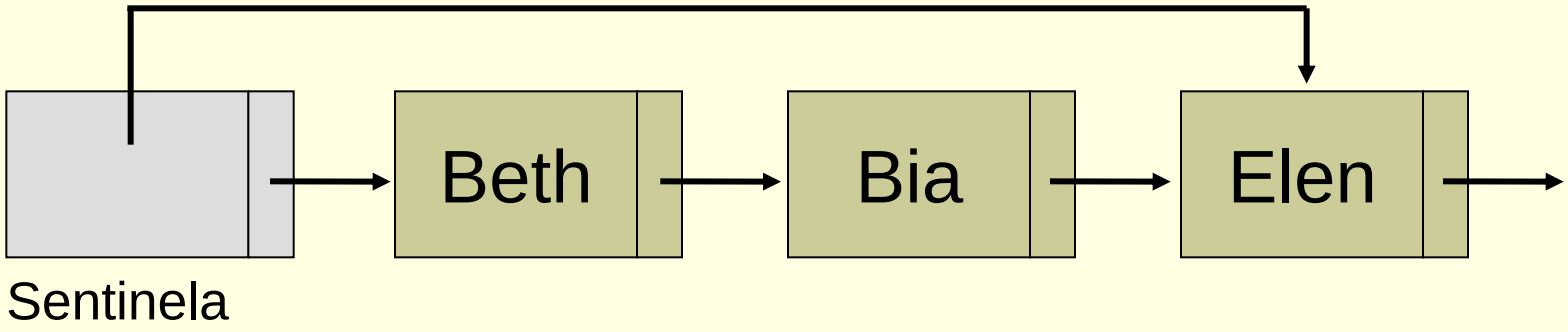
Sentinela

Lista com nó de cabeçalho

- Possibilidades de uso
  - Informação global sobre a lista que possa ser necessária na aplicação
    - Operação de busca de informação pode ser simplificada
    - Operações de inserção e remoção podem se tornar mais caras

Lista com nó de cabeçalho

- Possibilidades de uso
  - Informações para uso da lista como pilha, fila, etc.
    - Exemplo: em vez de um ponteiro de fim da fila, o nó sentinela pode também apontar o fim
      - Sentinela indica o início da fila também



Lista não homogênea

- Lista “genérica”
- Possibilidade de usar uma mesma estrutura para armazenar informações diferentes
  - Inteiro, caracter, estrutura, etc.
- Não é necessário definir blocos diferentes

Lista não homogênea

- Solução 1
  - Definem-se vários campos de informação
  - Usam-se somente os necessários

```
struct no {
        char info1;
        int info2;
        struct no *prox;
}
```

- Desvantagem: memória alocada desnecessariamente

Lista não homogênea

- Solução 2
  - Definem-se vários ponteiros
  - Aloca-se memória conforme necessidade

```
struct no {
        char *info1;
        int *info2;
        struct no *prox;
}
```

Lista não homogênea

- Solução 3
  - Definem-se um ponteiro genérico para qualquer tipo

```
struct no {
        void *info;
        struct no *prox;
}
```

Lista não homogênea

- Solução 4
  - Usa-se um registro/estrutura variante

```
struct no {
        union {
                int ival;
                float fval;
                char cval;
        } elemento;
        int tipo;
        struct no *prox;
}
```

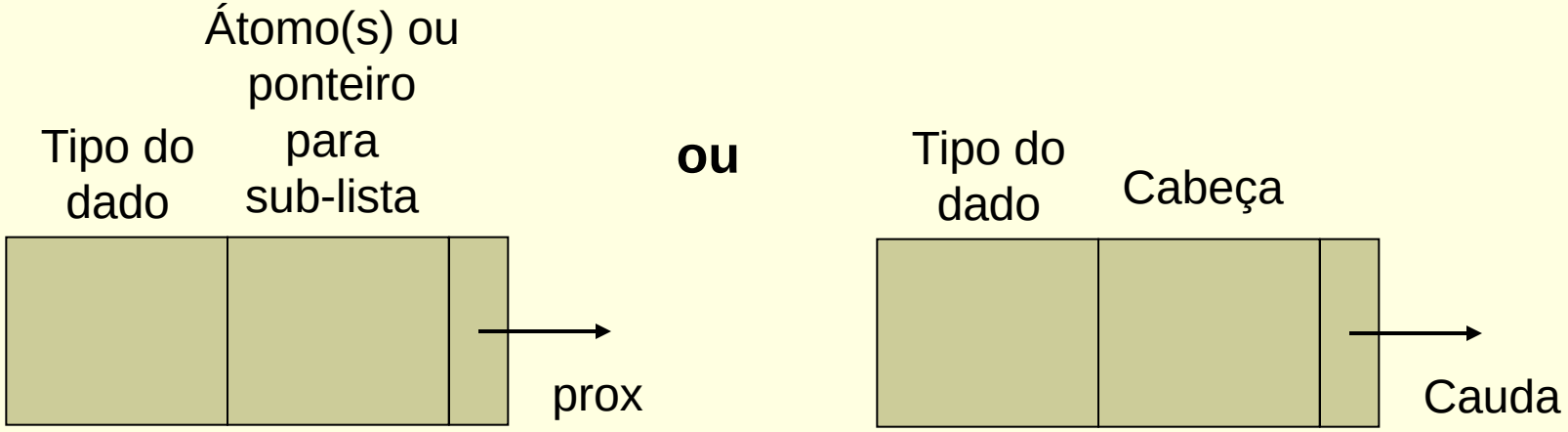
Lista generalizada

- Uma lista generalizada é aquela que pode ter como elemento ou um átomo ou uma outra lista (sub-lista)
  - Átomo: integer, real, char, string, etc.
- Cabeça e cauda
  - Cabeça: primeiro elemento da lista (átomo ou lista)
  - Cauda: o resto (uma outra lista, mesmo que vazia)

Lista generalizada

- Definição formal
  - Uma lista generalizada A é uma seqüência finita de $n \geq 0$ elementos α_1, α_1, ... α_n, em que α_i são átomos ou listas. Os elementos α_i, com $1 \leq i \leq n$, que não são átomos são chamados sub-listas de A.

- Estrutura básica do bloco de memória



Lista generalizada

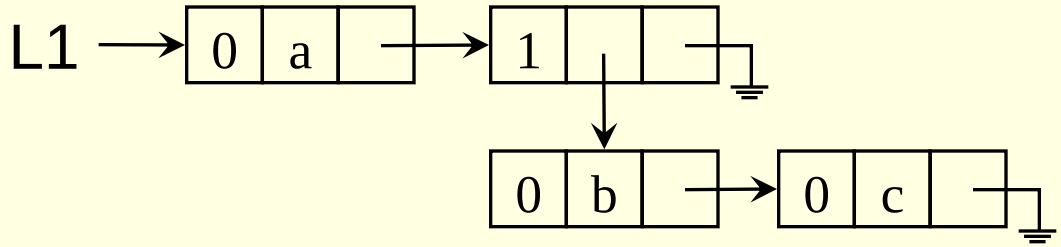
- Suponha que uma lista seja representada por elementos entre parênteses (no estilo da linguagem de programação LISP) ou entre colchetes (no estilo de PROLOG)
  - (a,b,c) ou [a,b,c]
  - (a,(b,c)) ou [a,[b,c]]
  - (a,(b),(c)) ou [a,[b],[c]]
  - (a,b,()) ou [a,b,[]]

- Tipo=0 indica átomo e tipo=1 indica sub-lista

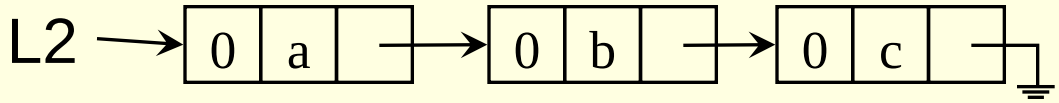
Lista generalizada

- Exemplos de representação

L1 = (a,(b,c))



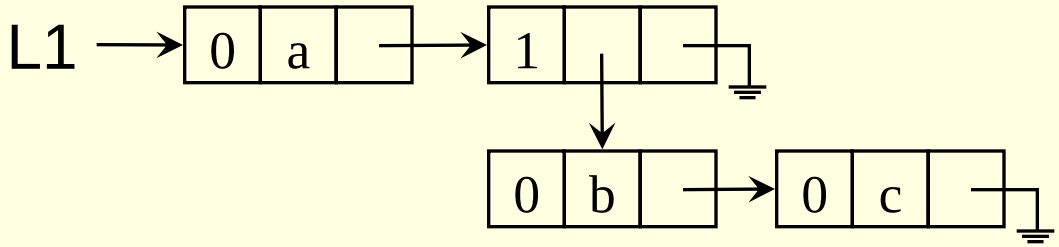
L2 = (a,b,c)



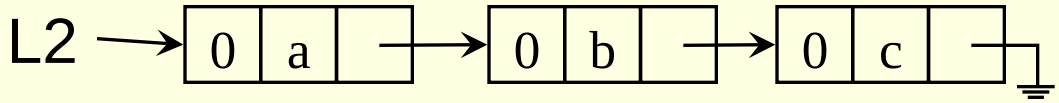
Lista generalizada

- Exemplos de representação

L1 = (a,(b,c))



L2 = (a,b,c)



Cabeça(L1)? Cauda(L1)? Cabeça(Cauda(L1))?

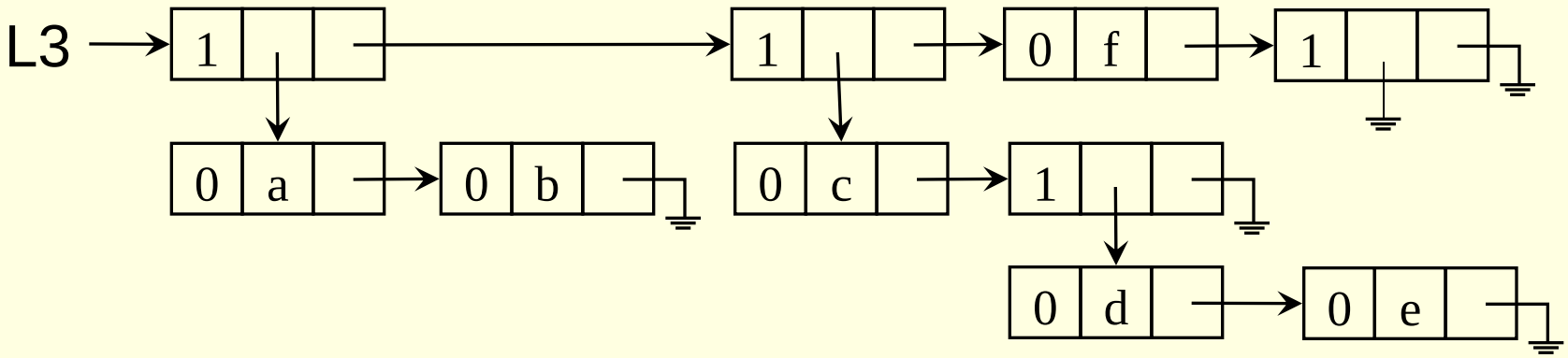
Cabeça(L2)? Cauda(L2)? Cabeça(Cauda(L2))?

Lista generalizada

- Exercício: faça a representação da lista L3 **((a,b),(c,(d,e)),f,())**

Lista generalizada

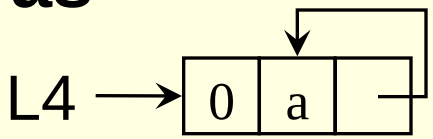
- Exercício: faça a representação da lista L3 **((a,b),(c,(d,e)),f,())**



Lista generalizada

Listas Recursivas

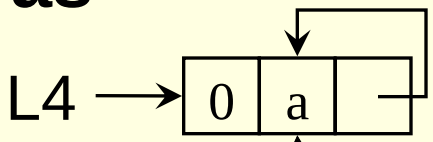
L4 = (a,L4)



Lista generalizada

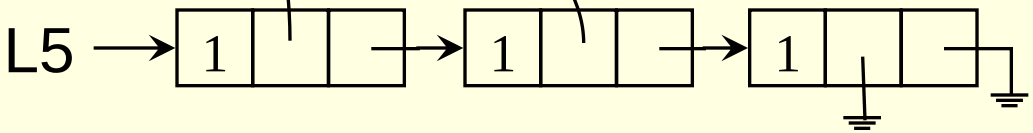
Listas Recursivas

L4 = (a,L4)



Listas Compartilhadas

L5 = (L4,L4,())



Lista generalizada

- Declaração em C
  - Union

```
typedef char elem;

typedef struct bloco {
        union {
                elem atomo;
                struct bloco *sublista;
        } info;
        int tipo;
        struct bloco *prox;
} no;
```

Lista generalizada

- Exercício
  - Implementar uma sub-rotina para verificar se um átomo x está em uma lista generalizada
    - Apenas na lista principal (primeiro nível da lista)

Lista generalizada

- Exercício
  - Implementar uma sub-rotina para verificar se um átomo x está em uma lista generalizada
    - Em qualquer parte dela

Lista generalizada

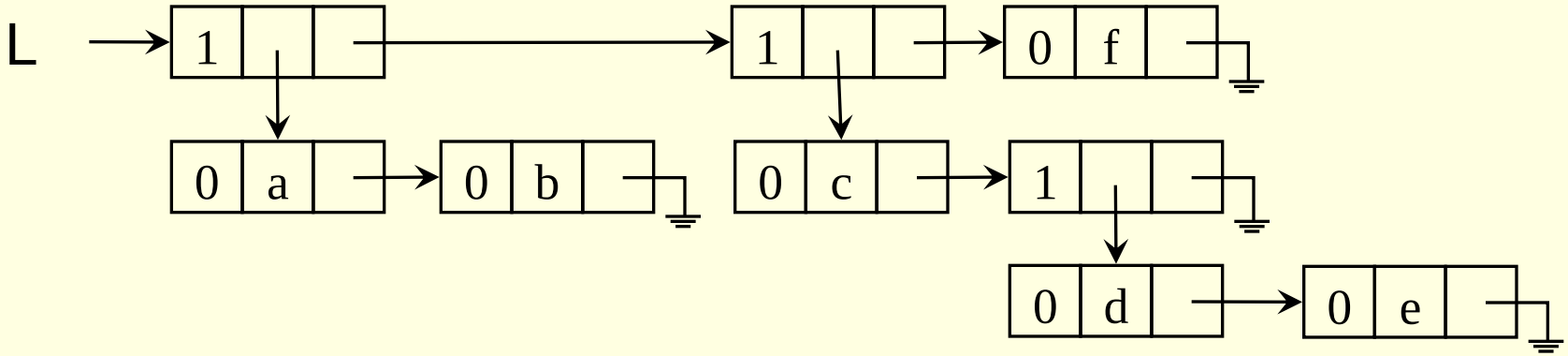
- Exercício
  - Implementar uma sub-rotina para verificar se duas listas generalizadas são completamente iguais

Lista generalizada

- Exercício
  - Implementar uma sub-rotina para verificar se duas listas generalizadas são estruturalmente iguais
    - O conteúdo em si não importa

Lista generalizada

- Exercício
  - Implementar uma sub-rotina que determina a profundidade máxima de uma lista generalizada
    - A=(a,(b)) ➔ prof(A)=2
    - B=(a,b,c) ➔ prof(B)=1
    - C=() ➔ prof(C)=0
    - Por exemplo, para o caso abaixo, a sub-rotina deveria retornar profundidade 3



Lista generalizada e polinômios

- Considere os polinômios:

$P1 = 4x^2y^3z + 3xy + 5$

$P2 = x^{10}y^3z^2 + 2x^8y^2z^2 + x^4y^4z + 6x^3y^4z + 2yz$

$P3 = 3x^2y$

(a) n° de termos: variável
  - P1=3, P2=5, P3=1

(b) n° de variáveis: variável
  - P1=3, P2=3, P3=2

(c) nem todo termo é expresso com todas as variáveis

Lista generalizada e polinômios

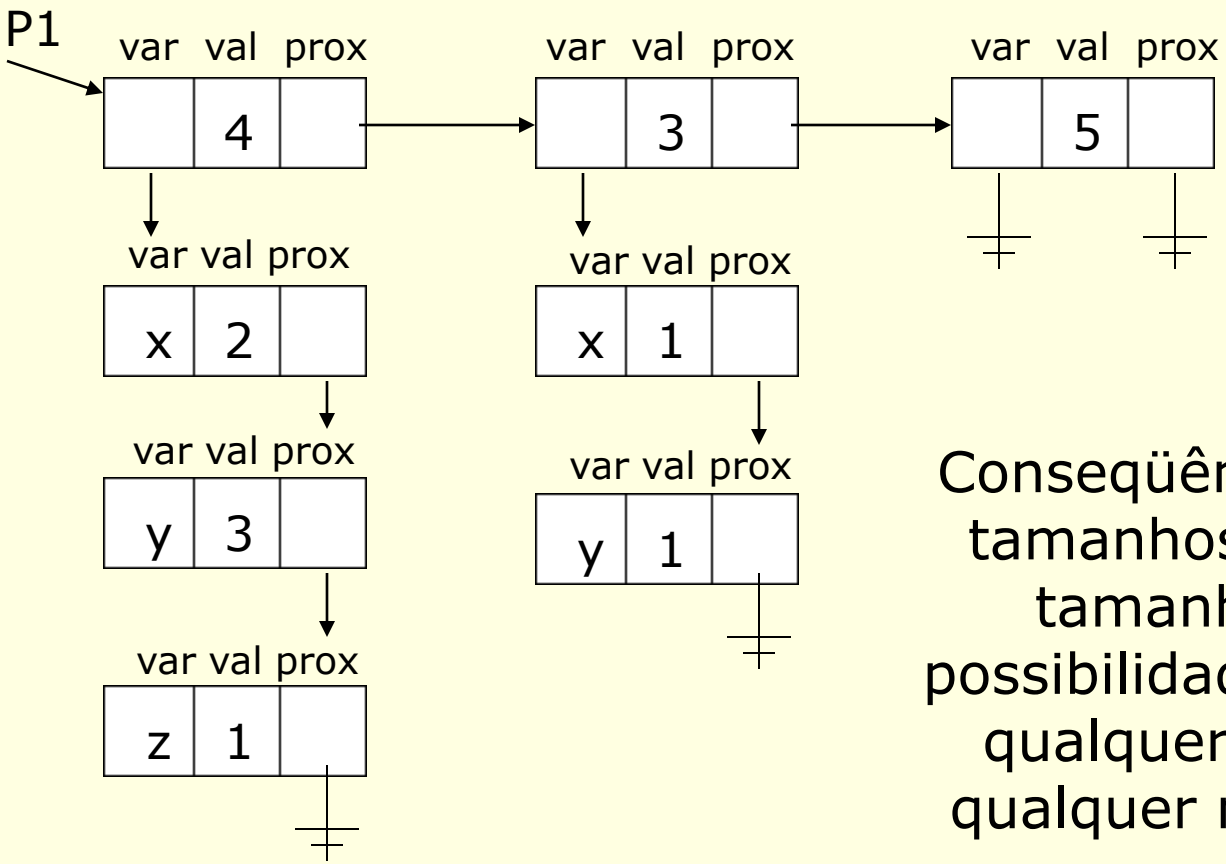
- Objetivos

  - representar de forma a otimizar o uso de memória
  - representação única para todo polinômio

- Solução: lista generalizada

Lista generalizada e polinômios

Ex: P1 = $4x^2y^3z + 3xy + 5$



Conseqüência: registros de tamanhos fixos; listas de tamanhos variáveis; possibilidade de representar qualquer polinômio com qualquer n° de variáveis e qualquer grau

Exercício para casa

- Implementar uma função que receba (a) um polinômio representado via lista generalizada e (b) os valores das variáveis do polinômio, para, a seguir, percorrer essa lista a fim de calcular e retornar o resultado do polinômio.

Créditos

- *Material gentilmente cedido pelo Prof. Thiago A. S. Pardo*