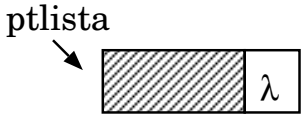
Listas Simplesmente Encadeadas

- Modelo
- Algoritmo de Percurso
- Algoritmo de Busca

Modelo de Listas Simplesmente Encadeadas

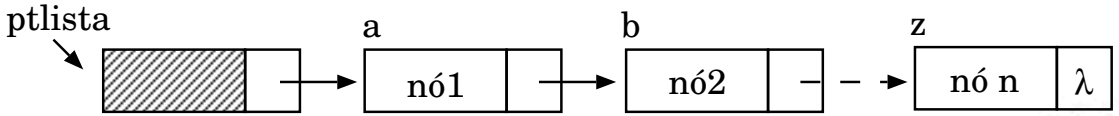
- Cada nó contém 2 campos:
  - INFO (com a informação a respeito do elemento representado)
  - PROX (com um ponteiro para o próximo nó)
- Existe um nó especial, chamado "nó-cabeça", nunca removido, que passa a ser o nó indicado pelo ponteiro de início de lista (PTLISTA).

Lista vazia:



Animação

Lista não vazia:



Algoritmo de Percurso em Lista Simplesmente Encadeada

Idéia:

- PONT inicia apontando para o primeiro nó da lista (aquele que é apontado pelo nó-cabeça PTLISTA)
- O Algoritmo consiste em seguir consecutivamente pelos endereços do campo PROX
- O Algoritmo termina quando PONT = λ

Complexidade: O(n), onde n é o número de nós da lista.

Descrição do Algoritmo de Percurso em Lista Simplesmente Encadeada

Algoritmo: Percurso de lista simplesmente encadeada com nó-cabeça apontado por PTLISTA

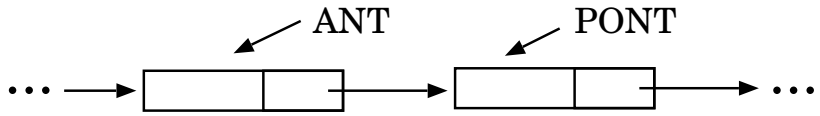
```
PONT := PTLISTA↑.PROX
enquanto  PONT ≠ λ faça
    VISITA( PONT↑.INFO )
    PONT  : =  PONT↑.PROX
```

- Aqui, o procedimento VISITA(PONT↑.INFO) efetua algum processamento no nó sendo atualmente visitado no percurso, como por exemplo impressão, atualização do conteúdo, etc.

Algoritmo de Busca em Lista Simplesmente Encadeada (Ordenada)

Idéia:

- X é a chave procurada.
- PONT retorna ponteiro para o elemento procurado
- ANT retorna ponteiro para o elemento anterior ao procurado



- Obs: Caso a chave X não seja encontrada na lista, PONT aponta para λ e ANT indica o elemento anterior ao último pesquisado.

Descrição do Algoritmo de Busca em Lista Simplesmente Encadeada (Ordenada)

Algoritmo: Busca em uma lista simplesmente encadeada, ordenada.

```
procedimento BUSCA_ENC_ORD(  X,  ANT,  PONT )
    ANT  : =  PTLISTA
    PONT : =  λ
    PTR  : =  PTLISTA↑.PROX  % PTR : ponteiro de percurso
    enquanto  PTR ≠ λ  faça
        se PTR↑.CHAVE < X
           então                    %  atualiza ANT e PTR
               ANT  : =  PTR
               PTR  : =  PTR↑.PROX
           senão
               se PTR↑.CHAVE  =  X
                  então
                     PONT  : =  PTR
               PTR  : =  λ
```

- Complexidade: O(n), onde n é o número de nós da lista.

Exercício

Exercício: Descrever um algoritmo de busca em uma lista simplesmente encadeada, não ordenada.

Tempo: 8 minutos

Solução

Algoritmo: Busca em uma lista simplesmente encadeada, não ordenada.

```
procedimento BUSCA_ENC_NÃO_ORD(  X,  ANT,  PONT )
    ANT  :=  PTLISTA
    PONT :=  λ
    PTR  :=  PTLISTA↑.PROX   %  PTR : ponteiro de percurso
    enquanto  PTR ≠ λ faça
    se PTR↑.CHAVE = X          %  chave encontrada
       então                   %  atualiza ANT e PTR
           PONT := PTR
           PTR  := λ
       senão                   %  atualiza ANT e PTR
           ANT := PTR
           PTR := PTR↑.PROX
```

Exercício

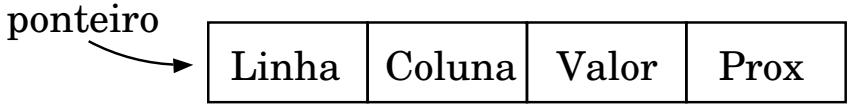
Seja A uma matriz esparsa n x m, isto é, boa parte de seus elementos são nulos ou irrelevantes. Descrever uma estrutura de dados que represente A, e cujo espaço total seja O(k) em vez de O(nm), onde k é o número total de elementos não irrelevantes de A.

- Sugestão: Utilize uma lista simplesmente encadeada

Tempo: 6 minutos

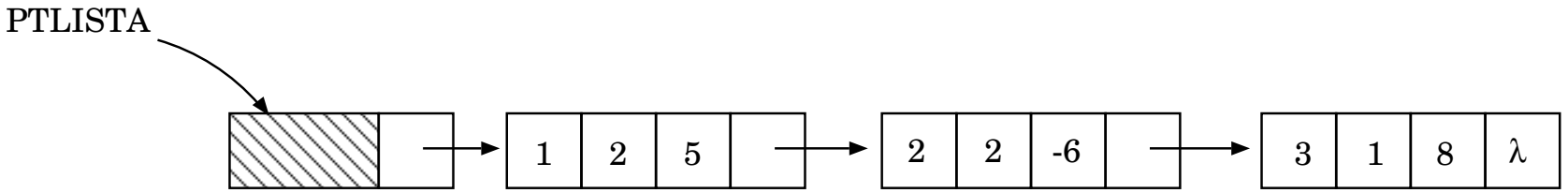
Solução

- Podemos utilizar uma lista simplesmente encadeada onde cada nó contém os índices da linha e da coluna de um elemento.



- Exemplo: Representação da seguinte matriz:

	1	2	3
1	0	5	0
2	0	-6	0
3	8	0	0



Exercício Final

- Desenvolva um algoritmo que imprima os elementos de uma lista simplesmente encadeada em ordem inversa à ordem da lista.

  - Sugestão: use uma PILHA

Uma animação de pilhas utilizando listas encadeadas pode ser vista ao clicar no botão ao lado.

Animação