

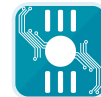


Algoritmos e Estrutura de Dados

Aula 09 – Listas Encadeadas – Parte I

Profa. Alana Oliveira e Prof. Davi Viana

alanaslz@gmail.com e davi.viana@dsi.ufma.br



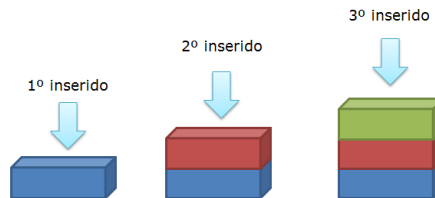
ENGENHARIA DA COMPUTAÇÃO
UNIVERSIDADE FEDERAL DO MARANHÃO



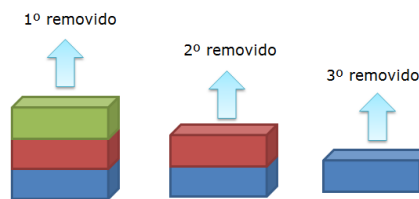
Se não entender, pergunte!

Como funciona uma pilha?

- Empilhar:



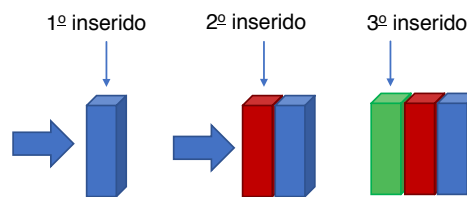
- Desempilhar:



3

Como funciona uma fila?

- Enfileirar:



- Desenfileirar:



4

Estrutura de Dados (ED)

Definição: organização de dados e operações (algoritmos) que podem ser aplicados sobre esses dados como forma de apoio à solução de problemas.

Exemplos de **EDs**:

- Pilhas
- Filas
- Listas lineares 
- Árvores

Profa. Alana Oliveira, ECP / UFMA

5

Definição de Lista em Python

- Uma lista é uma coleção ordenada de zero ou mais itens, de um mesmo tipo ou não.
- Python fornece um tipo de dados embutido que implemente listas como posições contíguas de memória.

Notas:

6.1	2.3	9.4	5.1	8.9	9.8	10	7.0	6.3	4.4
-----	-----	-----	-----	-----	-----	----	-----	-----	-----

Posição: 0 1 2 3 4 5 6 7 8 9

6

E se Python
(ou outra linguagem)
não tivesse tal
implementação?



Pode-se pensar em duas
Estruturas de Dados para
implementar Listas:

- Listas Não-ordenadas
- Listas Ordenadas

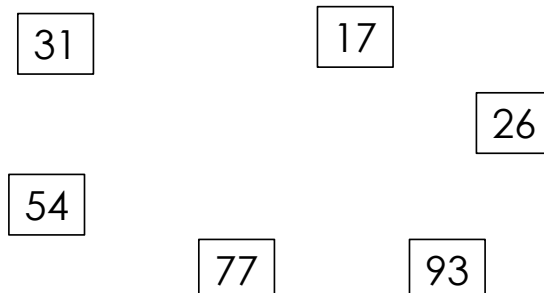
Ambas podem ser implementadas
utilizando-se o conceito de
listas encadeadas

Profa. Alana Oliveira, ECP / UFMA

7

Definição de Lista Encadeada

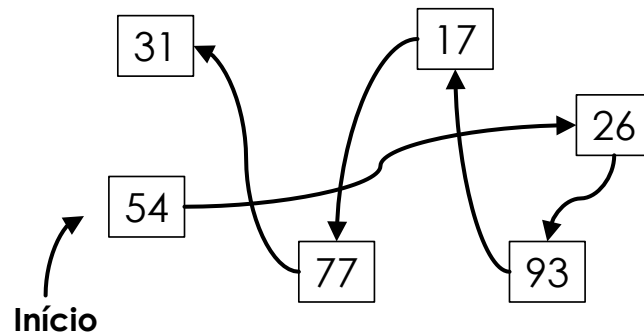
- Em listas encadeadas, há elementos e ligações entre eles, tal que a posição relativa do elementos seja mantida.



8

Definição de Lista Encadeada

- Em listas encadeadas, há elementos e ligações entre eles, tal que a posição relativa do elementos seja mantida.



9

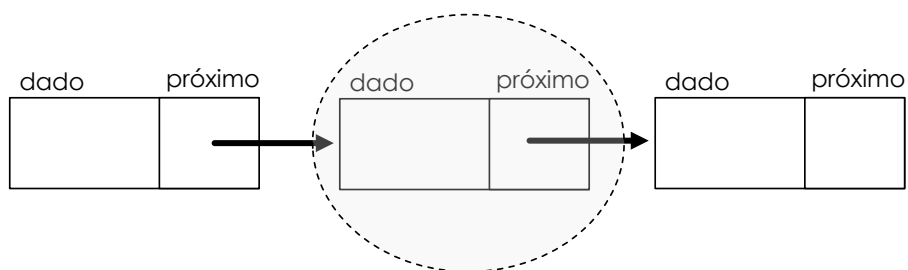
Definição de Lista Encadeada

- Existem três importantes classes que devem ser trabalhadas em listas encadeadas
 - Classe No
 - Classe ListaNaoOrdenada
 - Classe ListaOrdenada – Parte II

10

Lista Encadeada: Classe No

- Cada objeto da classe No possui um valor associado e uma referência para o próximo objeto da classe No.



11

Classe No → em Python

```

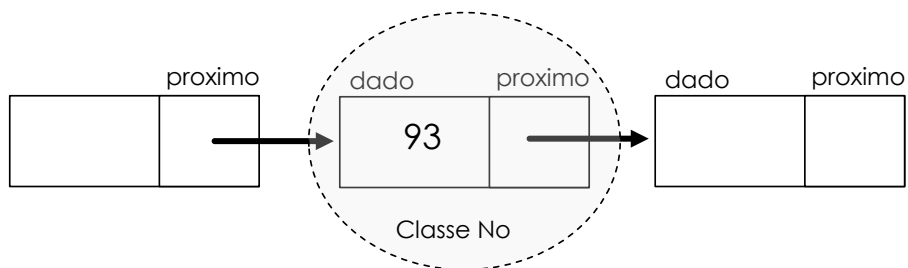
1 class No:
2     def __init__(self, inidado):
3         self.dado = inidado
4         self.proximo = None
5
6     def pegaDado(self):
7         return self.dado
8
9     def pegaProximo(self):
10        return self.proximo
11
12    def botaDado(self, novoDado):
13        self.dado = novoDado
14
15    def botaProximo(self, novoProximo):
16        self.proximo = novoProximo

```

12

Classe No → em Python

```
18 temp = No(93)
19 print temp.pegado()
```



13

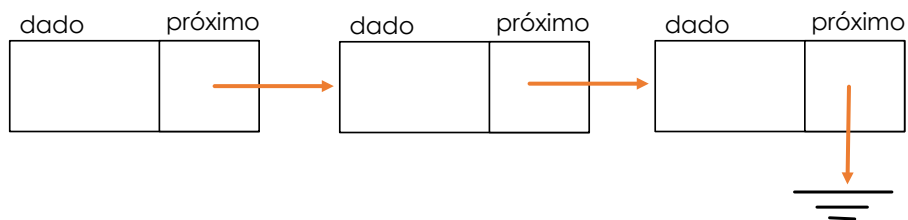
Lista Encadeada: Classe ListaNaoOrdenada

- Uma lista não ordenada consiste de uma coleção de nós, cada um ligado ao outro por referência. Uma vez que o nó inicial é conhecido, os demais podem ser acessados através de sucessivas referências.

14

Lista Encadeada: Classe ListaNaoOrdenada

- Uma lista não ordenada consiste de uma coleção de nós, cada um ligado ao outro por **referência**. Uma vez que o nó inicial é conhecido, os demais podem ser acessados através de sucessivas referências.



15

Classe ListaNaoOrdenada: Implementação

```

1 class ListaNaoOrdenada:
2     def __init__(self):
3         self.inicio = None
4
5 mylist = ListaNaoOrdenada()

```



16

Definição de Lista Encadeada

Classe ListaNaoOrdenada

- As operações comumente utilizadas em listas não ordenadas são:
 - **Vazia**: verifica se a lista está vazia.
 - **Inserir**: insere um novo item na estrutura.
 - **Buscar**: verifica se um determinado item está na estrutura.
 - **Remover**: remove um determinado item da estrutura.
 - **Tamanho**: retorna a quantidade de itens da estrutura.
 - **Imprimir**: imprime os itens da estrutura.

17

Definição de Lista Encadeada

Classe ListaNaoOrdenada: Operações

- Operação **Vazia**:

```
22 def Vazia(self):
23     return self.inicio == None
```



18

Definição de Lista Encadeada

Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```
25 def Inserir(self,item):  
26     temp = No(item)  
27     temp.botaProximo(self.inicio)  
28     self.inicio = temp
```



19

Executando o método “inserir” para o
item = 17

20

Definição de Lista Encadeada

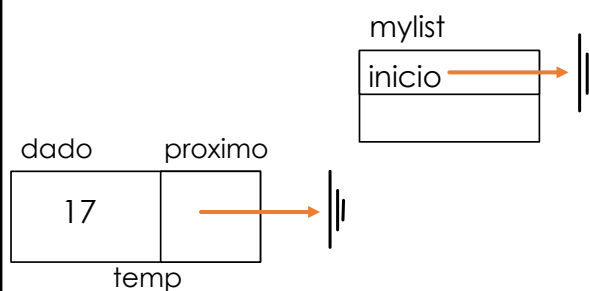
Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```

25  def Inserir(self,item):
26      temp = No(item)
27      temp.botaProximo(self.inicio)
28      self.inicio = temp

```



21

Definição de Lista Encadeada

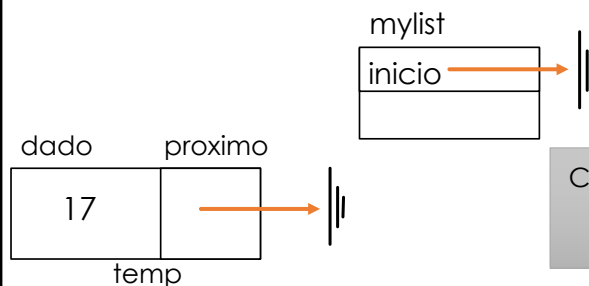
Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```

25  def Inserir(self,item):
26      temp = No(item)
27      temp.botaProximo(self.inicio)
28      self.inicio = temp

```



Como início possui "None"
então o próximo fica o
mesmo "valor"

22

Definição de Lista Encadeada

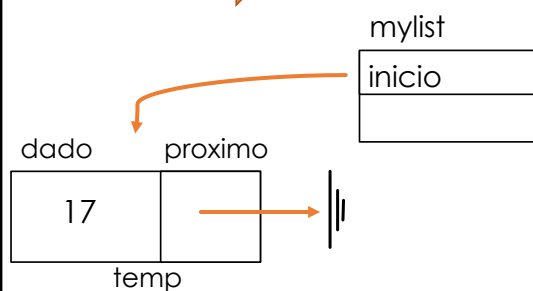
Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```

25     def Inserir(self,item):
26         temp = No(item)
27         temp.botaProximo(self.inicio)
28         self.inicio = temp

```



23

- Executando o método “inserir” mais uma vez, para o item = 28

24

Definição de Lista Encadeada

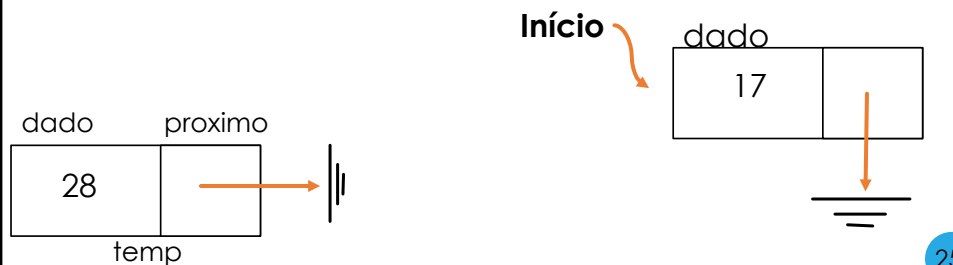
Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```

25     def Inserir(self,item):
26         temp = No(item)
27         temp.botaProximo(self.inicio)
28         self.inicio = temp

```



25

Definição de Lista Encadeada

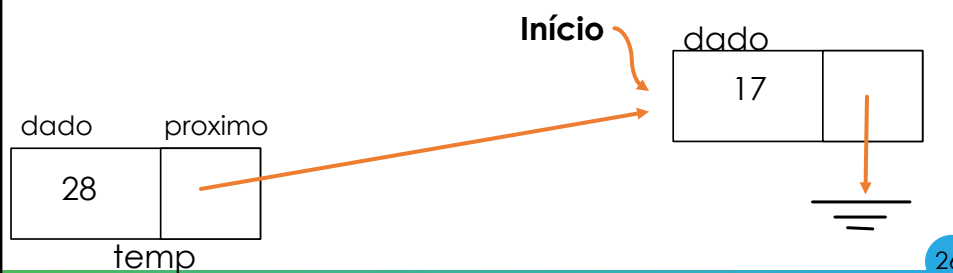
Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```

25     def Inserir(self,item):
26         temp = No(item)
27         temp.botaProximo(self.inicio)
28         self.inicio = temp

```



26

Definição de Lista Encadeada

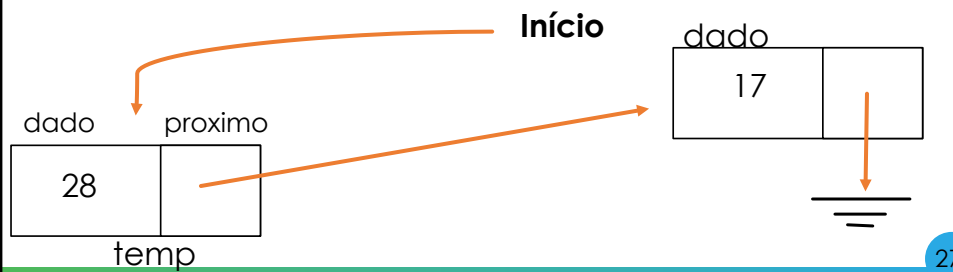
Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```

25     def Inserir(self,item):
26         temp = No(item)
27         temp.botaProximo(self.inicio)
28         self.inicio = temp

```



Executando o método “inserir” mais uma vez, agora para o item 36

Definição de Lista Encadeada

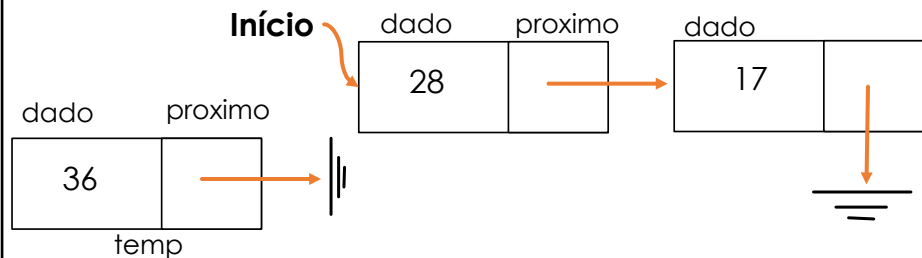
Classe ListaNaoOrdenada: Operações

- Operação Inserir:

```

25     def Inserir(self,item):
26         temp = No(item)
27         temp.botaProximo(self.inicio)
28         self.inicio = temp

```



29

Definição de Lista Encadeada

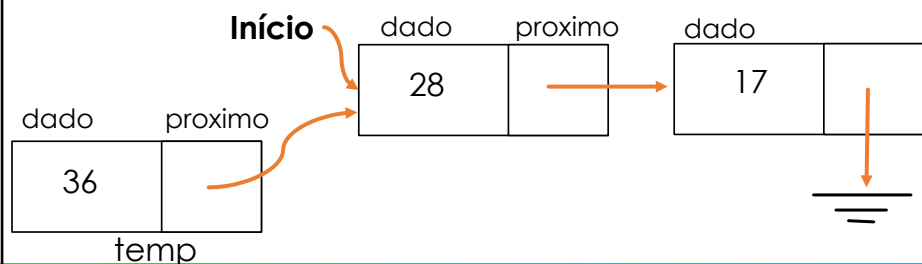
Classe ListaNaoOrdenada: Operações

- Operação Inserir:

```

25     def Inserir(self,item):
26         temp = No(item)
27         temp.botaProximo(self.inicio)
28         self.inicio = temp

```



30

Definição de Lista Encadeada

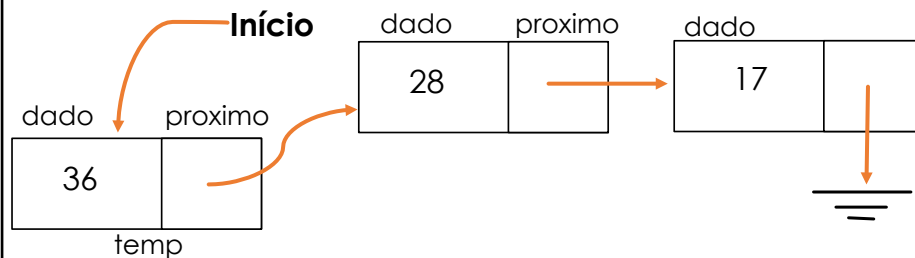
Classe ListaNaoOrdenada: Operações

- Operação **Inserir**:

```

25     def Inserir(self,item):
26         temp = No(item)
27         temp.botaProximo(self.inicio)
28         self.inicio = temp

```

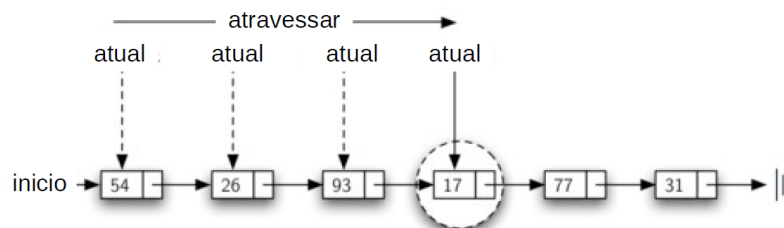


31

Definição de Lista Encadeada

Classe ListaNaoOrdenada: Operações

- Operação **Buscar**:



32

Definição de Lista Encadeada

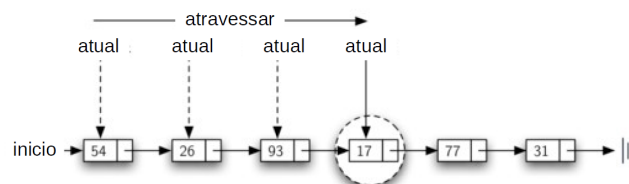
Classe ListaNaoOrdenada: Operações

- Operação **Buscar**:

```

30 def Buscar(self,item):
31     atual = self.inicio
32     encontrou = False
33     while atual != None and not encontrou:
34         if atual.pegado() == item:
35             encontrou = True
36         else:
37             atual = atual.pegProximo()
38     return encontrou

```

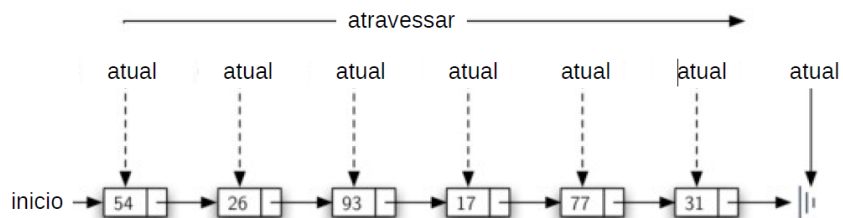


33

Definição de Lista Encadeada

Classe ListaNaoOrdenada: Operações

- Operação **Tamanho**:



34

Definição de Lista Encadeada

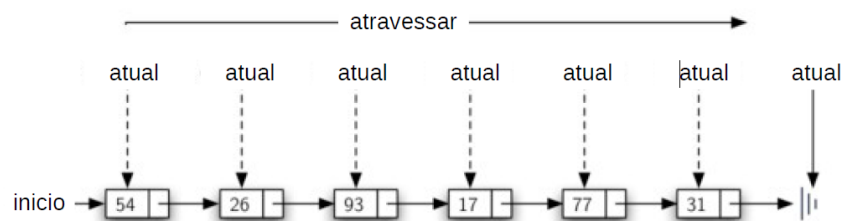
Classe ListaNaoOrdenada: Operações

- Operação **Tamanho**:

```

56     def Tamanho(self):
57         atual = self.inicio
58         conta = 0
59         while atual != None:
60             conta = conta + 1
61             atual = atual.pegProximo()
62         return conta

```

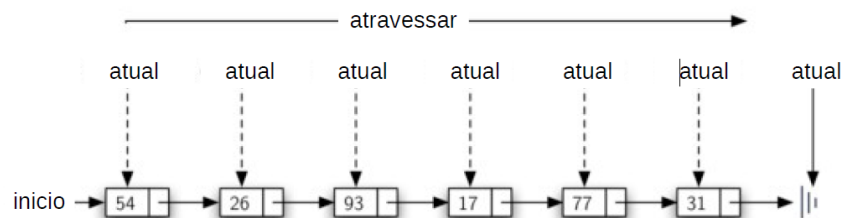


35

Definição de Lista Encadeada

Classe ListaNaoOrdenada: Operações

- Operação **Imprimir**:



36

Definição de Lista Encadeada

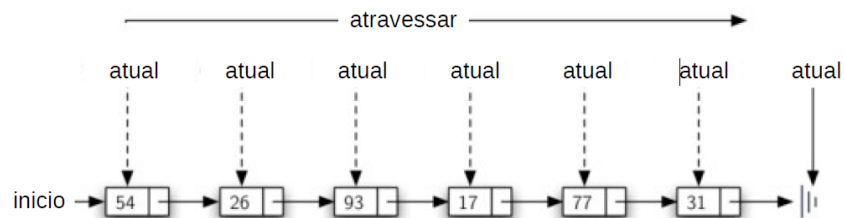
Classe ListaNaoOrdenada: Operações

- Operação **Imprimir**:

```

64     def Imprimir(self):
65         atual = self.inicio
66         while atual != None:
67             print(str(atual.pegado()))
68             atual = atual.pegProximo()

```



37

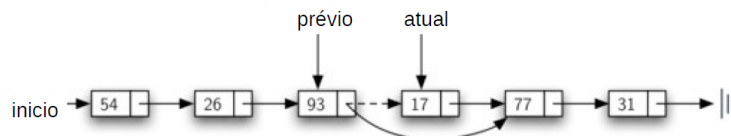
Como seria a remoção de um elemento de uma lista encadeada?

38

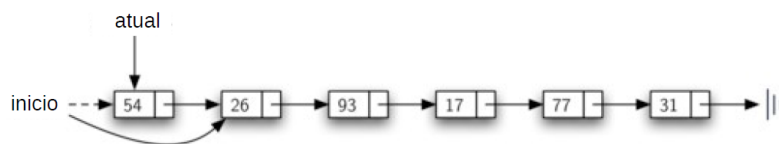
Definição de Lista Encadeada

Classe ListaNaoOrdenada: Operações

- Operação **Remove** – meio da lista:



- Operação **Remove** – início da lista:



39

Faça o código de remoção e
apresente na próxima aula 😊

40

Material Complementar

- http://www3.ifrn.edu.br/~jurandy/fdp/doc/aprenda-python/capitulo_17.html
- https://www.youtube.com/watch?v=hWKvkh_hCVc
- <https://www.youtube.com/watch?v=83Eot8f0Br4>