Linguagem Python

- *Data Science*
- *Machine Learning*
- Desenvolvimento de Aplicativos
- *Big Data*

Biblioteca sbbst

Um módulo Python que implementa e executa essas atividades eficientemente é a ***sbbst*** (do inglês, *self-balancing binary search tree*).

Biblioteca sbbst

Um módulo Python que implementa e executa essas atividades eficientemente é a ***sbbst*** (do inglês, *self-balancing binary search tree*).

Usar o **!pip install sbbst** para instalar o módulo.

Biblioteca sbbst

Implementando Árvores Balanceadas com sbbst

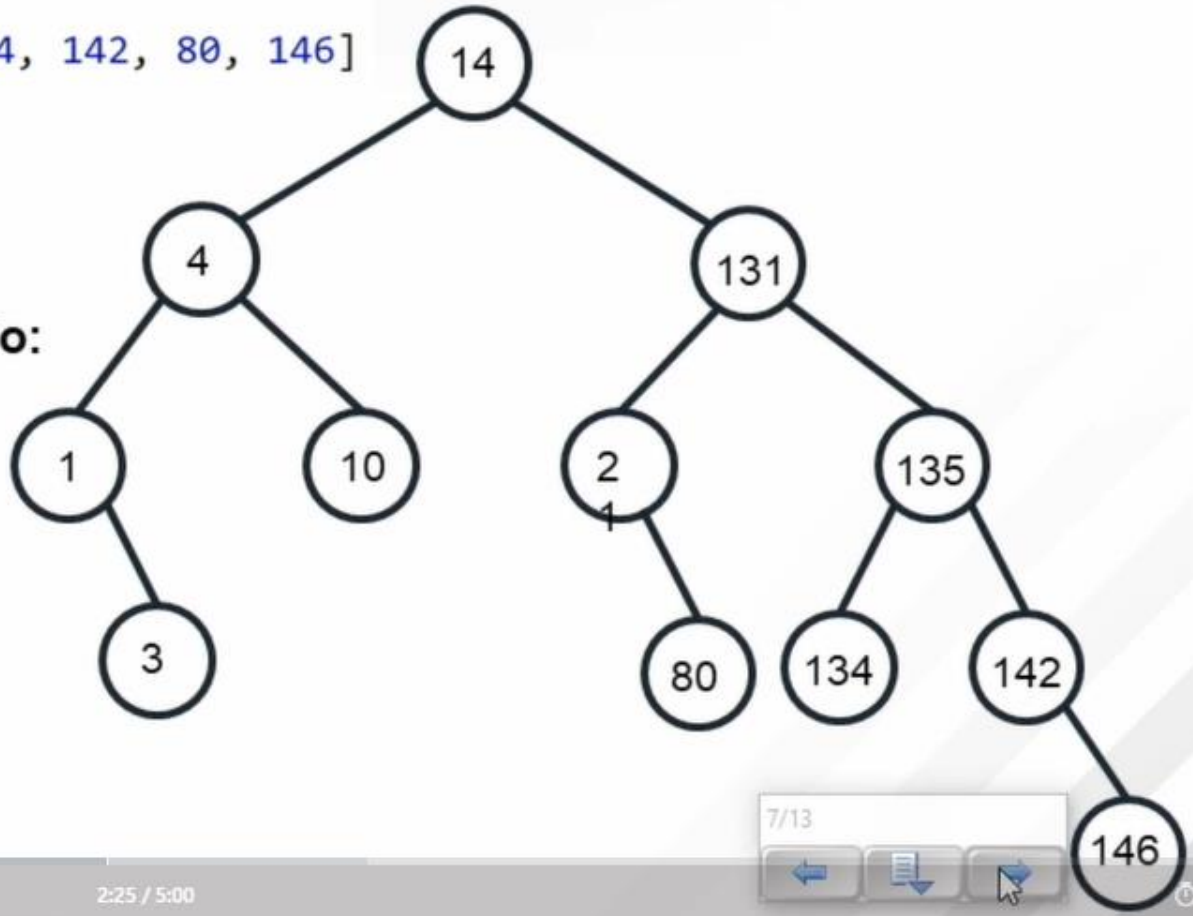
```
1  from sbbst import sbbst
2
3  tree = sbbst()
4  nums = [131, 4, 134, 135, 10, 1, 3, 21, 14, 142, 80, 146]
5  for num in nums:
6      tree.insert(num)
```

Biblioteca sbbst

Implementando Árvores Balanceadas com sbbst

```
1  from sbbst import sbbst
2
3  tree = sbbst()
4  nums = [131, 4, 134, 135, 10, 1, 3, 21, 14, 142, 80, 146]
5  for num in nums:
6      tree.insert(num)
```

Resultado:



Biblioteca sbbst

Implementando Árvores Balanceadas com sbbst

```
1  from sbbst import sbbst
2
3  tree = sbbst()
4  nums = [131, 4, 134, 135, 10, 1, 3, 21, 14, 142, 80, 146]
5  for num in nums:
6      tree.insert(num)
7
8  print("Número de elementos:",tree.getSize())
9  print("Altura:",tree.getHeightTree())
10 print("Min valor:",tree.getMinVal())
11 print("Max valor:",tree.getMaxVal())
12 print("3º menor valor:",tree.kthsmallest(3))
13 print("2º maior valor:",tree.kthlargest(2))
14 print("Pre-Ordem:",tree.inOrder())
15 print("In-Ordem:",tree.preOrder())
16 print("Pos-Ordem:",tree.postOrder())
17 tree.delete(128)
18 tree.delete(3)
19 tree.insert(55)
```

Biblioteca sbbst

Implementando Árvores Balanceadas com sbbst

```
1  from sbbst import sbbst
2
3  tree = sbbst()
4  nums = [131, 4, 134, 135, 10, 1, 3, 21, 14, 142, 80, 146]
5  for num in nums:
6      tree.insert(num)
7
8  print("Número de elementos:",tree.getSize())
9  print("Altura:",tree.getHeightTree())
10 print("Min valor:",tree.getMinVal())
11 print("Max valor:",tree.getMaxVal())
12 print("3º menor valor:",tree.kthsmallest(3))
13 print("2º maior valor:",tree.kthlargest(2))
14 print("Pre-Ordem:",tree.inOrder())
15 print("In-Ordem:",tree.preOrder())
16 print("Pos-Ordem:",tree.postOrder())
17 tree.delete(128)
18 tree.delete(3)
19 tree.insert(55)
```

```
Número de elementos: 12
Altura: 5
Min valor: 1
Max valor: 146
```

Biblioteca sbbst

Implementando Árvores Balanceadas com sbbst

```
1  from sbbst import sbbst
2
3  tree = sbbst()
4  nums = [131, 4, 134, 135, 10, 1, 3, 21, 14, 142, 80, 146]
5  for num in nums:
6      tree.insert(num)
7
8  print("Número de elementos:",tree.getSize())
9  print("Altura:",tree.getHeightTree())
10 print("Min valor:",tree.getMinVal())
11 print("Max valor:",tree.getMaxVal())
12 print("3º menor valor:",tree.kthsmallest(3))
13 print("2º maior valor:",tree.kthlargest(2))
14 print("Pre-Ordem:",tree.inOrder())
15 print("In-Ordem:",tree.preOrder())
16 print("Pos-Ordem:",tree.postOrder())
17 tree.delete(128)
18 tree.delete(3)
19 tree.insert(55)
```

Pre-Ordem: [14, 4, 1, 3, 10, 131, 21, 80, 135, 134, 142, 146]

In-Ordem: [1, 3, 4, 10, 14, 21, 80, 131, 134, 135, 142, 146]

Pos-Ordem: [3, 1, 10, 4, 80, 21, 134, 146, 142, 135, 131, 14]

Biblioteca sbbst

Implementando Árvores Balanceadas com sbbst

```
1  from sbbst import sbbst
2
3  tree = sbbst()
4  nums = [131, 4, 134, 135, 10, 1, 3, 21, 14, 142, 80, 146]
5  for num in nums:
6      tree.insert(num)
7
8  print("Número de elementos:",tree.getSize())
9  print("Altura:",tree.getHeightTree())
10 print("Min valor:",tree.getMinVal())
11 print("Max valor:",tree.getMaxVal())
12 print("3º menor valor:",tree.kthsmallest(3))
13 print("2º maior valor:",tree.kthlargest(2))
14 print("Pre-Ordem:",tree.inOrder())
15 print("In-Ordem:",tree.preOrder())
16 print("Pos-Ordem:",tree.postOrder())
17 tree.delete(128)
18 tree.delete(3)
19 tree.insert(55)
```

Operações na árvores, como **exclusão** e **inserção** de chaves!