EDA - Aula 1

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Busca

Inserção Remoção

Aula 11 Árvore Binária de Busca (ABB)

Estruturas de Dados Avançadas

Professor Eurinardo Rodrigues Costa Universidade Federal do Ceará Campus Russas

2021.1

Sumário

FDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

> Algoritmos Busca Inserção Remocão

- Aulas Passadas
- 2 Árvore Binária de Busca (ABB)
 - Algoritmos
 - Busca
 - Inserção
 - Remoção

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Inserção Remoção

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

> Busca Inserção

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

> Igoritmos Busca

Inserção Remoção

Árvores

Toda árvore possui folha

FDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

> lgoritmos Busca

Inserção Remoção

- Toda árvore possui folha
- Toda árvore G = (V, E), temos que |E| = |V| 1

EDA - Aula 11
Prof.

Aulas Passadas

Árvore Binária de Busca (ABB)

> Algoritmos Busca Inserção

- Toda árvore possui folha
- Toda árvore G = (V, E), temos que |E| = |V| 1
- Toda AB G = (V, E), temos que |V| + 1 ponteiros nulos

EDA - Aula 1^o Prof. Eurinardo

Aulas Passadas

de Busca (ABB)

Busca Inserção

- Toda árvore possui folha
- Toda árvore G = (V, E), temos que |E| = |V| 1
- Toda AB G = (V, E), temos que |V| + 1 ponteiros nulos
- Conversão de uma floresta em uma AB

EDA - Aula 1^o Prof. Eurinardo

Aulas Passadas

de Busca (ABB)

Busca Inserção

- Toda árvore possui folha
- Toda árvore G = (V, E), temos que |E| = |V| 1
- Toda AB G = (V, E), temos que |V| + 1 ponteiros nulos
- Conversão de uma floresta em uma AB
- Árvore Binária de Busca

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Busca Insercão

EDA - Aula 11

Prof. Eurinardo

Aulas Passada:

Árvore Binária de Busca (ABB)

> Algoritmos Busca

Definição (Árvore Binária de Busca)

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

> Algoritmos Busca

Busca Inserção Remoção

Definição (Árvore Binária de Busca)

AB em que cada nó v temos que

EDA - Aula 11

Prof. Eurinardo

Aulas Passada:

Árvore Binária de Busca (ABB)

> lgoritmos Busca

Busca Inserção Remoção

Definição (Árvore Binária de Busca)

AB em que cada nó v temos que

v.esq.chave < v.chave < v.dir.chave

EDA - Aula 11

Prof. Eurinardo

Aulas Passada

Árvore Binária de Busca (ABB)

> Busca Inserção

Definição (Árvore Binária de Busca)

AB em que cada nó v temos que

v.esq.chave < v.chave < v.dir.chave

OBS.: considere os elementos distintos

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Busca

Inserção Remocão

Algoritmo 2: buscar(x, pt, f)

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Busca

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Busca

Inserção Romoção Algoritmo 3: buscar(x, pt, f)

Entrada: $x \rightarrow$ chave procurada

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Igoritmo

Busca

Inserçao Domoção Algoritmo 4: buscar(x, pt, f)

Entrada: $x \rightarrow$ chave procurada

pt o ponteiro para raíz da ABB

Busca

Algoritmo 5: buscar(x, pt, f)

Entrada: $x \rightarrow$ chave procurada

pt o ponteiro para raíz da ABB

$$\int_{0}^{\infty} dt dt = \lambda$$

Saída:
$$f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}$$

$$3$$
 , se $pt.dir = \lambda$

Prof.

Busca

Algoritmo 6: buscar(x, pt, f)

Entrada: $x \rightarrow$ chave procurada

 $pt \rightarrow$ ponteiro para raíz da ABB

Saída:
$$f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}$$

2 , se
$$pt.esq = \lambda$$

, se
$$\mathit{pt.dir} = \lambda$$

1 se
$$pt = \lambda$$
 então

Prof.

Busca

Algoritmo 7: buscar(x, pt, f)

Entrada: $x \rightarrow$ chave procurada

 $pt \rightarrow$ ponteiro para raíz da ABB

$$\int_{0}^{\infty} 0 \quad \text{, se } pt = \lambda$$

Saída: $f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}$

2 , se
$$pt.esq = \lambda$$

, se
$$pt.dir = \lambda$$

1 se
$$pt = \lambda$$
 então $f \leftarrow 0$

Prof.

Busca

Algoritmo 8: buscar(x, pt, f)

Entrada: $x \rightarrow$ chave procurada

 $pt \rightarrow$ ponteiro para raíz da ABB

$$\int_{0}^{\infty} se pt = \lambda$$

Saída: $f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}$

2 , se
$$pt.esq = \lambda$$

3 , se $pt.dir = \lambda$

1 se $pt = \lambda$ então $f \leftarrow 0$;

2 senão

Prof.

Algoritmo 9: buscar(x, pt, f)

Entrada: $x \rightarrow$ chave procurada

 $pt \rightarrow$ ponteiro para raíz da ABB

$$\int 0$$
 , se $pt = \lambda$

Saída: $f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}$

, se
$$\textit{pt.dir} = \lambda$$

1 se
$$pt = \lambda$$
 então $f \leftarrow 0$;

2 senão

se
$$pt.chave = x$$
 então

Prof.

```
Algoritmo 10: buscar(x, pt, f)
```

Entrada: $x \rightarrow$ chave procurada $pt \rightarrow$ ponteiro para raíz da ABB Saída: $f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}$

aída:
$$f \leftarrow \begin{cases} 1 & \text{, se } x \text{ esta ha Ab} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}$$

1 se $pt = \lambda$ então $f \leftarrow 0$;

2 senão

se
$$pt.chave = x$$
 então $f \leftarrow 1$;

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Igoritmos

busca ...

Inserção

```
Algoritmo 11: buscar(x, pt, f)Entrada: x \rightarrow chave procuradapt \rightarrow ponteiro para raíz da ABBSaída: f \leftarrow
\begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}1 se pt = \lambda então f \leftarrow 0;2 senão3 se pt.chave = x então f \leftarrow 1;4 senão
```

```
EDA - Aula 1<sup>-</sup>
```

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Busca

Inserção

```
Algoritmo 12: buscar(x, pt, f)
    Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
   Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
1 se pt = \lambda então f \leftarrow 0;
2 senão
            se pt.chave = x então f \leftarrow 1;
            senão
5
                   se x < pt.chave então
```

```
EDA - Aula 11
```

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

```
.lgoritmo:
```

Busca

```
Algoritmo 13: buscar(x, pt, f)
    Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
   Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
1 se pt = \lambda então f \leftarrow 0;
2 senão
           se pt.chave = x então f \leftarrow 1;
           senão
                   se x < pt.chave então
                           se pt.esq = \lambda então
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

```
(Igoritmo:
```

Busca

```
Algoritmo 14: buscar(x, pt, f)
    Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
   Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
1 se pt = \lambda então f \leftarrow 0;
2 senão
           se pt.chave = x então f \leftarrow 1;
           senão
                   se x < pt.chave então
                           se pt.esq = \lambda então f \leftarrow 2;
```

```
EDA - Aula 11
```

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

lgoritmo

Busca

```
Algoritmo 15: buscar(x, pt, f)
    Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
   Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
1 se pt = \lambda então f \leftarrow 0;
2 senão
           se pt.chave = x então f \leftarrow 1;
           senão
                   se x < pt.chave então
                           se pt.esq = \lambda então f \leftarrow 2;
                           senão
```

```
EDA - Aula 11
```

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

.lgoritmo:

Busca

```
Algoritmo 16: buscar(x, pt, f)
   Entrada: x \rightarrow chave procurada
                   pt → ponteiro para raíz da ABB
   Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
1 se pt = \lambda então f \leftarrow 0;
2 senão
           se pt.chave = x então f \leftarrow 1;
           senão
                   se x < pt.chave então
                           se pt.esq = \lambda então f \leftarrow 2;
6
                           senão
                                pt \leftarrow pt.esq
```

```
EDA - Aula 11
```

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

dgoritmo

Innovaña

```
Algoritmo 17: buscar(x, pt, f)
    Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
   Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
1 se pt = \lambda então f \leftarrow 0;
2 senão
           se pt.chave = x então f \leftarrow 1;
           senão
5
                   se x < pt.chave então
                           se pt.esq = \lambda então f \leftarrow 2;
6
                           senão
                                 pt \leftarrow pt.esq
buscar(x, pt, f)
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

ugoritmo

busca

```
Algoritmo 18: buscar(x, pt, f)
    Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
    Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
 1 se pt = \lambda então f \leftarrow 0;
 2 senão
            se pt.chave = x então f \leftarrow 1;
            senão
 5
                   se x < pt.chave então
                           se pt.esq = \lambda então f \leftarrow 2;
 6
                           senão
                                pt \leftarrow pt.esq
                                buscar(x, pt, f)
                   senão
10
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

ugoritmo

Dusca

```
Algoritmo 19: buscar(x, pt, f)
     Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
    Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
 1 se pt = \lambda então f \leftarrow 0:
 2 senão
            se pt.chave = x então f \leftarrow 1;
            senão
 5
                    se x < pt.chave então
                            se pt.esq = \lambda então f \leftarrow 2;
 6
                            senão
 7
                                pt \leftarrow pt.esq
buscar(x, pt, f)
                    senão
10
                            se pt.dir = \lambda então
11
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

ugoritmo

Dusca

```
Algoritmo 20: buscar(x, pt, f)
     Entrada: x \rightarrow chave procurada
                     pt → ponteiro para raíz da ABB
    Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
 1 se pt = \lambda então f \leftarrow 0:
 2 senão
            se pt.chave = x então f \leftarrow 1:
            senão
 5
                    se x < pt.chave então
                            se pt.esq = \lambda então f \leftarrow 2;
 6
                            senão
 7
                                pt \leftarrow pt.esq
buscar(x, pt, f)
                    senão
10
                            se pt.dir = \lambda então f \leftarrow 3;
11
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

lgoritmo

Dusca

```
Algoritmo 21: buscar(x, pt, f)
     Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
    Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
 1 se pt = \lambda então f \leftarrow 0:
 2 senão
            se pt.chave = x então f \leftarrow 1:
            senão
 5
                   se x < pt.chave então
                           se pt.esq = \lambda então f \leftarrow 2;
 6
                           senão
 7
                                pt \leftarrow pt.esq
buscar(x, pt, f)
                   senão
10
                           se pt.dir = \lambda então f \leftarrow 3;
11
12
                           senão
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

ugoritmo

busca

```
Algoritmo 22: buscar(x, pt, f)
     Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
    Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
 1 se pt = \lambda então f \leftarrow 0:
 2 senão
            se pt.chave = x então f \leftarrow 1:
            senão
 5
                   se x < pt.chave então
                           se pt.esa = \lambda então f \leftarrow 2:
 6
                           senão
 7
                               pt \leftarrow pt.esq
buscar(x, pt, f)
                   senão
10
                           se pt.dir = \lambda então f \leftarrow 3;
11
12
                           senão
                                 pt \leftarrow pt.dir
13
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

ugoritmo

busca

Remoção

```
Algoritmo 23: buscar(x, pt, f)
    Entrada: x \rightarrow chave procurada
                    pt → ponteiro para raíz da ABB
    Saída: f \leftarrow \begin{cases} 0 & \text{, se } pt = \lambda \\ 1 & \text{, se } x \text{ está na ABB} \\ 2 & \text{, se } pt.esq = \lambda \\ 3 & \text{, se } pt.dir = \lambda \end{cases}
 1 se pt = \lambda então f \leftarrow 0:
 2 senão
            se pt.chave = x então f \leftarrow 1:
            senão
 5
                   se x < pt.chave então
                           se pt.esa = \lambda então f \leftarrow 2:
 6
                           senão
 7
                                 pt \leftarrow pt.esq
                                buscar(x, pt, f)
                   senão
10
                           se pt.dir = \lambda então f \leftarrow 3;
11
12
                           senão
                               pt \leftarrow pt.dir
buscar(x, pt, f)
13
14
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Busca

Inserção

Prof.

Inserção

Algoritmo 25: inserir(x, ptraiz)

Prof.

Inserção

Algoritmo 26: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Inserção

Algoritmo 27: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB $ptraiz \rightarrow$ ponteiro para a raiz da ABB

EDA - Aula 11

Prof. Eurinardo

Aulas Passada:

Arvore Binária de Busca (ABB)

agonimo:

Inserção

Algoritmo 28: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB

 $\textit{ptraiz} \rightarrow \text{ponteiro para a raiz da ABB}$

Saída: $ptraiz \leftarrow ponteiro para raiz da ABB com x$

inserido

EDA - Aula 11

Prof. Eurinardo

Aulas Passada:

Árvore Binária de Busca (ABB)

Algoritmos

Inserção

Algoritmo 29: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB ptraiz \rightarrow ponteiro para a raiz da ABB

Saída: $ptraiz \leftarrow ponteiro para raiz da ABB com x$

inserido

1 pt ← ptraiz

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Arvore Binária de Busca (ABB)

Algoritmos

Inserção

Algoritmo 30: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB ptraiz \rightarrow ponteiro para a raiz da ABB

Saída: $ptraiz \leftarrow ponteiro para raiz da ABB com <math>x$

inserido

1 pt ← ptraiz

2 buscar(x, pt, f)

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Arvore Binária de Busca (ABB)

Algoritmos

Inserção

Algoritmo 31: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB

ptraiz → ponteiro para a raiz da ABB

Saída: ptraiz ← ponteiro para raiz da ABB com x inserido

1 pt ← ptraiz

2 buscar(x, pt, f)

set f = 1 então

EDA - Aula 1

Prof. Eurinardo

Aulas Passadas

Arvore Binária de Busca (ABB)

Algoritmos

Inserção

Algoritmo 32: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB ptraiz \rightarrow ponteiro para a raiz da ABB

Saída: ptraiz ← ponteiro para raiz da ABB com x inserido

1 pt ← ptraiz

2 buscar(x, pt, f)

3 se f = 1 então "elemento já existe";

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Arvore Binária de Busca (ABB)

Algoritmos

Inserção

Algoritmo 33: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB

 $\textit{ptraiz} \rightarrow \text{ponteiro para a raiz da ABB}$

Saída: ptraiz ← ponteiro para raiz da ABB com x inserido

1 pt ← ptraiz

buscar(x, pt, f)

3 se f = 1 então "elemento já existe";

4 senão

```
EDA - Aula 11
Prof.
```

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Inserção

Algoritmo 34: inserir(x, ptraiz)

Entrada: $x \rightarrow$ chave para inserir na ABB

 $\textit{ptraiz} \rightarrow \text{ponteiro para a raiz da ABB}$

Saída: ptraiz ← ponteiro para raiz da ABB com x inserido

1 pt ← ptraiz

2 buscar(x, pt, f)

з **se** f = 1 **então** "elemento já existe";

4 senão

criarNó(*novo*)

```
EDA - Aula 11
Prof.
```

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Inserção

```
Algoritmo 35: inserir(x, ptraiz)
```

Entrada: $x \to \text{chave para inserir na ABB}$ $ptraiz \to \text{ponteiro para a raiz da ABB}$

Saída: ptraiz ← ponteiro para raiz da ABB com x inserido

- 1 pt ← ptraiz
- buscar(x, pt, f)
- 3 **se** f = 1 **então** "elemento já existe";
- 4 senão
- 5 criarNó(*novo*)
- 6 novo.chave $\leftarrow x$

```
Prof. Eurinardo

Algoritmo 36: inserir(x, ptraiz)

Entrada: x \to \text{chave para inserir na ABB}

ptraiz \to \text{ponteiro para a raiz da ABB}

Saída: ptraiz \leftarrow \text{ponteiro para raiz da ABB com } x

inserido

1 pt \leftarrow ptraiz

2 buscar(x, pt, f)

1 serão

1 serão "elemento já existe";

4 senão

5 | criarNó(novo)

6 | novo, chave \leftarrow x
```

 $novo.esq \leftarrow novo.dir \leftarrow \lambda$

```
Algoritmo 37: inserir(x, ptraiz)

Entrada: x \to \text{chave para inserir na ABB}

Passadas

Arvore Binária de Busca

ABB)

Apportunos

Busca

Remoção

Fermoção

Fermo
```

 $novo.chave \leftarrow x$

se f = 0 então

 $novo.esq \leftarrow novo.dir \leftarrow \lambda$

6

8

```
Algoritmo 38: inserir(x, ptraiz)
                        Entrada: x \rightarrow chave para inserir na ABB
                                   ptraiz → ponteiro para a raiz da ABB
                        Saída: ptraiz \leftarrow ponteiro para raiz da ABB com x
                                 inserido
                     1 pt ← ptraiz
                     2 buscar(x, pt, f)
Inserção
                      se f = 1 então "elemento já existe";
                     4 senão
                             criarNó(novo)
                      5
                              novo.chave \leftarrow x
                     6
                              novo.esq \leftarrow novo.dir \leftarrow \lambda
                              se f = 0 então ptraiz \leftarrow novo;
                     8
```

```
Algoritmo 39: inserir(x, ptraiz)
  Entrada: x \rightarrow chave para inserir na ABB
             ptraiz → ponteiro para a raiz da ABB
  Saída: ptraiz \leftarrow ponteiro para raiz da ABB com x
           inserido
1 pt ← ptraiz
2 buscar(x, pt, f)
se f = 1 então "elemento já existe";
4 senão
       criarNó(novo)
5
        novo.chave \leftarrow x
6
        novo.esq \leftarrow novo.dir \leftarrow \lambda
        se f = 0 então ptraiz \leftarrow novo;
8
```

se f = 2 então

```
Algoritmo 40: inserir(x, ptraiz)
  Entrada: x \rightarrow chave para inserir na ABB
             ptraiz → ponteiro para a raiz da ABB
  Saída: ptraiz \leftarrow ponteiro para raiz da ABB com x
           inserido
1 pt ← ptraiz
2 buscar(x, pt, f)
se f = 1 então "elemento já existe";
4 senão
       criarNó(novo)
5
        novo.chave \leftarrow x
6
        novo.esq \leftarrow novo.dir \leftarrow \lambda
        se f = 0 então ptraiz \leftarrow novo;
8
        se f = 2 então pt.esq \leftarrow novo;
9
```

```
EDA - Aula 11
Prof.
Eurinardo
```

Aulas Passadas

Arvore Binária de Busca (ABB)

Busca

Inserção

```
Algoritmo 41: inserir(x, ptraiz)
   Entrada: x \rightarrow chave para inserir na ABB
              ptraiz → ponteiro para a raiz da ABB
   Saída: ptraiz \leftarrow ponteiro para raiz da ABB com x
            inserido
1 pt ← ptraiz
2 buscar(x, pt, f)
 se f = 1 então "elemento já existe";
4 senão
        criarNó(novo)
 5
        novo.chave \leftarrow x
6
        novo.esq \leftarrow novo.dir \leftarrow \lambda
        se f = 0 então ptraiz \leftarrow novo;
8
        se f = 2 então pt.esq \leftarrow novo;
        se f = 3 então
10
```

```
Algoritmo 42: inserir(x, ptraiz)
  Entrada: x \rightarrow chave para inserir na ABB
             ptraiz → ponteiro para a raiz da ABB
  Saída: ptraiz \leftarrow ponteiro para raiz da ABB com x
           inserido
1 pt ← ptraiz
2 buscar(x, pt, f)
 se f = 1 então "elemento já existe";
4 senão
       criarNó(novo)
5
        novo.chave \leftarrow x
6
        novo.esg \leftarrow novo.dir \leftarrow \lambda
        se f = 0 então ptraiz \leftarrow novo;
8
        se f = 2 então pt.esq \leftarrow novo;
```

se f = 3 então $pt.dir \leftarrow novo$;

10

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Lanca e e e

Remoção

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Inserção Remoção Algoritmo 44: remover(x, ptraiz)

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Lancas in

Remoção

Algoritmo 45: remover(x, ptraiz)

 $\textbf{1} \quad \textit{pt} \leftarrow \textit{ptraiz}$

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Insercão

Remoção

Algoritmo 46: remover(x, ptraiz)

- $\textbf{1} \quad \textit{pt} \leftarrow \textit{ptraiz}$
- 2 buscar(x, pt, f)

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Incorção

Remoção

Algoritmo 47: remover(x, ptraiz)

- $\textbf{1} \quad \textit{pt} \leftarrow \textit{ptraiz}$
- 2 buscar(x, pt, f)
- 3 se $f \neq 1$ então

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Insercão

Remoção

Algoritmo 48: remover(x, ptraiz)

- 1 $pt \leftarrow ptraiz$
- buscar(x, pt, f)
- 3 se $f \neq 1$ então "elemento não existe";

EDA - Aula 1

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Incorção

Remoção

Algoritmo 49: remover(x, ptraiz)

- 1 pt ← ptraiz
- buscar(x, pt, f)
- 3 se $f \neq 1$ então "elemento não existe";
- 4 senão

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Inserção Remoção $\mathbf{1} \quad \textit{pt} \leftarrow \textit{ptraiz}$

buscar(x, pt, f)

з se $f \neq 1$ então "elemento não existe";

Algoritmo 50: remover(x, ptraiz)

4 senão

 $aux \leftarrow pt$

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Insercão

Remoção

```
Algoritmo 51: remover(x, ptraiz)
```

```
1 pt \leftarrow ptraiz

2 buscar(x, pt, f)

3 \mathbf{se} \ f \neq 1 então "elemento não existe";

4 \mathbf{senão}

5 aux \leftarrow pt

6 \mathbf{se} \ pt.esq = \lambda \ \mathbf{então}
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Insercão

Remoção

Algoritmo 52: remover(x, ptraiz)

```
1 pt \leftarrow ptraiz

2 buscar(x, pt, f)

3 \mathbf{se} \ f \neq 1 então "elemento não existe";

4 \mathbf{senão}

5 | aux \leftarrow pt

6 \mathbf{se} \ pt. esq = \lambda \ \mathbf{então}

7 | pt \leftarrow pt. dir
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos Busca

Inserçao Remocão

```
Algoritmo 53: remover(x, ptraiz)
```

```
1 pt \leftarrow ptraiz

2 buscar(x, pt, f)

3 se f \neq 1 então "elemento não existe";

4 senão

5 | aux \leftarrow pt

6 | se \ pt.esq = \lambda \ então

7 | pt \leftarrow pt.dir

8 senão
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmo

Inserção Remoção

```
Algoritmo 54: remover(x, ptraiz)
```

```
1 pt \leftarrow ptraiz
2 buscar(x, pt, f)
3 se f \neq 1 então "elemento não existe";
4 senão
5 | aux \leftarrow pt
6 | se pt.esq = \lambda então
7 | pt \leftarrow pt.dir
8 | se pt.dir = \lambda então
9 | se pt.dir = \lambda então
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Incorção

Remocão

```
Algoritmo 55: remover(x, ptraiz)
```

```
\begin{array}{lll} & pt \leftarrow ptraiz \\ & \text{buscar}(x,pt,f) \\ & \text{se } f \neq 1 \text{ então "elemento não existe"}; \\ & \text{senão} \\ & \text{5} & aux \leftarrow pt \\ & \text{se } pt.esq = \lambda \text{ então} \\ & & pt \leftarrow pt.dir \\ & \text{senão} \\ & & \text{se } pt.dir = \lambda \text{ então} \\ & & \text{genasor} \\ \\ & & \text{genasor} \\ & & \text{genasor}
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Inserção

Remoção

```
Algoritmo 56: remover(x, ptraiz)
```

```
1 pt ← ptraiz
2 buscar(x, pt, f)
3 se f \neq 1 então "elemento não existe";
4 senão
         aux \leftarrow pt
         se pt.esq = \lambda então
              pt \leftarrow pt.dir
        senão
8
              se pt.dir = \lambda então
                   pt \leftarrow pt.esq
10
              senão
11
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos Busca

Inserção Remocão

```
Algoritmo 57: remover(x, ptraiz)
1 pt ← ptraiz
2 buscar(x, pt, f)
з se f \neq 1 então "elemento não existe";
4 senão
          aux \leftarrow pt
          se pt.esq = \lambda então
               pt \leftarrow pt.dir
 7
 8
          senão
               se pt.dir = \lambda então
                     pt \leftarrow pt.esa
10
               senão
11
                     s \leftarrow pt.dir
12
13
                     se s.esq \neq \lambda então
                           enquanto s.esq \neq \lambda faça
14
                                 PaiS \leftarrow s
15
16
                                 s \leftarrow s.esa
                           PaiS.esq \leftarrow s.dir
17
                           s.dir \leftarrow pt.dir
18
```

EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Busca

Inserção Remoção

```
Algoritmo 58: remover(x, ptraiz)
1 pt ← ptraiz
2 buscar(x, pt, f)
з se f \neq 1 então "elemento não existe";
4 senão
         aux \leftarrow pt
         se pt.esq = \lambda então
 7
               pt \leftarrow pt.dir
 8
         senão
               se pt.dir = \lambda então
 9
                     pt \leftarrow pt.esa
10
               senão
11
12
                     s \leftarrow pt.dir
13
                     se s.esq \neq \lambda então
                           enguanto s.esq \neq \lambda faca
14
                                 PaiS \leftarrow s
15
16
                                s \leftarrow s.esa
                           PaiS.esg ← s.dir
17
                          s.dir \leftarrow pt.dir
18
                     s.esq \leftarrow pt.esq
19
```

7

8

9

10

11

12

13

14

15 16

17

18

19

20

Algoritmo 59: remover(x, ptraiz)

Prof.

Remoção

```
1 pt ← ptraiz
2 buscar(x, pt, f)
з se f \neq 1 então "elemento não existe";
4 senão
        aux \leftarrow pt
        se pt.esq = \lambda então
              pt \leftarrow pt.dir
        senão
              se pt.dir = \lambda então
                    pt \leftarrow pt.esa
              senão
                    s \leftarrow pt.dir
                    se s.esq \neq \lambda então
                         enguanto s.esq \neq \lambda faca
```

 $PaiS \leftarrow s$

 $s \leftarrow s.esa$ PaiS.esg ← s.dir

 $s.dir \leftarrow pt.dir$

 $s.esq \leftarrow pt.esq$

 $pt \leftarrow s$

EDA - Aula 11

Prof.

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos Busca

Inserção Remocão

```
Algoritmo 60: remover(x, ptraiz)
 1 pt ← ptraiz
 2 buscar(x, pt, f)
   se f \neq 1 então "elemento não existe";
 4 senão
          aux \leftarrow pt
          se pt.esq = \lambda então
               pt \leftarrow pt.dir
 8
          senão
               se pt.dir = \lambda então
 9
                     pt \leftarrow pt.esa
10
               senão
11
12
                     s \leftarrow pt.dir
13
                     se s.esq \neq \lambda então
                           enguanto s.esq \neq \lambda faca
14
                                 PaiS ← s
15
16
                                s \leftarrow s.esa
                           PaiS.esq ← s.dir
17
                           s.dir \leftarrow pt.dir
18
                     s.esq \leftarrow pt.esq
19
                     pt \leftarrow s
20
          liberar(aux)
21
```

Bibliografia

EDA - Aula 11 Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Busca Inserção Remoção SZWARCFITER, Jayme; MARKENZON, Lilian. Estruturas de Dados e Seus Algoritmos.3a edição. LTC, ano 2010. (ISBN 9788521617501). EDA - Aula 11

Prof. Eurinardo

Aulas Passadas

Árvore Binária de Busca (ABB)

Algoritmos

Insercão

Remoção

Obrigado!