Teórico 9

Gustavo Lopes Rodrigues

7 de Maio de 2020

1 Questão 1

```
1  public int remover(int x) throws Exception {
2    int resp;
3      if (raiz == null) {
4         throw new Exception("Erro ao remover2!");
5      } else if(x < raiz.elemento){
6         resp = remover(x, raiz.esq, raiz);
7      } else if (x > raiz.elemento){
8         resp = remover(x, raiz.dir, raiz);
9      } else if (raiz.dir == null) {
10     resp = raiz.elemento;
11        raiz = raiz.esq;
12     } else if (raiz.esq == null) {
13     resp = raiz.elemento;
14        raiz = raiz.dir;
15     } else {
16        resp = raiz.elemento;
17        raiz.esq = antecessor(raiz, raiz.esq);
18     }
19   return resp;
20  }
21
22 private int remover(int x, No i, No pai) throws Exception {
23   int resp;
24   if (i == null) {
25        throw new Exception("Erro ao remover2!");
```

```
26       } else if (x < i.elemento) {
27          resp = remover(x, i.esq, i);
28       } else if (x > i.elemento) {
29          resp = remover(x, i.dir, i);
30       } else if (i.dir == null) {
31      resp = i.elemento;
32      if ( pai.esq == i ) {
33       pai.esq = i.esq;
34      }
35      else {
36       pai.dir = i.esq;
37      }
38       } else if (i.esq == null) {
39      resp = i.elemento;
40      if ( pai.esq == i ) {
41       pai.esq = i.dir;
42      }
43      else {
44       pai.dir = i.dir;
45      }
46       } else {
47      resp = i.elemento;
48          i.esq = antecessor(i, i.esq);
49     }
50     return resp;
51   }
```

2 Questão 2 - TreeSort em C++

```
1  void treeSort(int array[]) {
2    this->raiz = null;
3    ArvoreBinaria resp = new ArvoreBinaria();
4    No menor;
5    for(int i : array) {
6      inserir(i);
7    }
8    for(int i : array) {
9      menor = mostrarMenor();
10     remover(menor.getElemento());
11     resp.inserir(menor.getElemento());
12   }
13   this->raiz = resp.getRaiz();
14 }
15
16   No mostrarMenor() {
17   No menor = new No(raiz.getElemento());
18   return mostrarMenor(raiz,menor);
19 }
20
21 No mostrarMenor(No i,No menor) {
22   if (i != null) {
23     No tmp1 = mostrarMenor(i.getEsq(),menor),tmp2 = mostrarMenor(
     i.getDir(),menor);
24     if ( i.getElemento() < menor.getElemento() ) {
25       menor = i;
26     } else if ( tmp1.getElemento() < menor.getElemento() ) {
27       menor = tmp1;
28     } else if ( tmp2.getElemento() < menor.getElemento() ) {
29       menor = tmp2;
30     }
31   }
32   return menor;
33 }
```

3 Analisando a complexidade da Questão 2

Analisando a complexidade desse algoritmo, a primeira coisa a notar-se, são os dois laços "for"dentro da função void "treeSort", como os laços vão de 0 a n, é seguro dizer que apenas os laços possuem o custo igual a: 0(n).

Porém, o algoritmo da treeSort usa o método de inserção,remoção e mostrar. Logo deixando o algoritmo mais complexo. Todos os métodos possuem um custo que varia entre O(log(n)) até O(n), logo, o pior caso na complexidade seria igual a :

$$4n^2 + 4$$

$$ou$$

$$0(n^2)$$

Já o melhor caso seria :

$$n * log(n) + n * 3log(n) + 4$$

$$ou$$

$$O(n * log(n))$$