Tópicos Avançados em Estrutura de Dados

Atividade 12

Bruna Galastri Guedes	18.00189-0
Daniel Ughini Xavier	18.00022-3
Rodolfo Cochi Bezerra	18.00202-0
Vítor Martin Simoni	18.00050-9
Leonardo Cury Haddad	18.00442-3
Leonardo de Barros Rodrigues	18.02401-7

01/08/2020

Questão 1

Main:

```
package br.maua;

import br.maua.Arvore.NoDaArvore;

public class Main {

    public static void main(String[] args) {
        //Criacao da Arvore
        NoDaArvore no6 = new NoDaArvore(6,null,null);
        NoDaArvore no5 = new NoDaArvore(5,null,null);
        NoDaArvore no3 = new NoDaArvore(3,no5,no6);
        NoDaArvore no4 = new NoDaArvore(4,null, null);
        NoDaArvore no2 = new NoDaArvore(2,no3, no4);
        NoDaArvore no1 = new NoDaArvore(1,null, null);
        NoDaArvore raiz = new NoDaArvore(0,no1, no2);

        //Prints
        System.out.println("Pre-ordem");
        raiz.imprimirPreOrdem(raiz);
        System.out.println("Em Ordem");
        raiz.imprimirEmOrdem(raiz);
        System.out.println("Pos Ordem");
        raiz.imprimirPosOrdem(raiz);
        System.out.println("Verificacao");
        raiz.Verificacao(raiz, 5);
        System.out.println("Maior Numero");
        raiz.MaiorNum(raiz);
        System.out.println(raiz.getMaiornum());
        System.out.println("Soma");
        raiz.Soma(raiz);
        System.out.println(raiz.getSoma());
        System.out.println("Menor num");
        raiz.MenorNum(raiz);
        System.out.println(raiz.getMenornum());
        System.out.println("contar nos");
        raiz.ContarNos(raiz);
        System.out.println(raiz.getNos());
        System.out.println("Media");
        raiz.Media(raiz);
        System.out.println("folhas");
        raiz.ContaFolhas(raiz);
        System.out.println(raiz.getFolhas());
        System.out.println("Contar Nulls");
        raiz.ContarNull(raiz);
        System.out.println(raiz.getNulos());
        System.out.println("Altura");
        System.out.println(raiz.altura(raiz));


    }
}
```

Classe Nó das Árvores:

```
package br.maua.Arvore;

public class NoDaArvore {

    //Estrutura da arvore
    public int valor;
    public NoDaArvore esquerda;
    public NoDaArvore direita;

    //Construtor do no da Arvore
    public NoDaArvore(int valor, NoDaArvore esquerda, NoDaArvore direita) {
        this.valor = valor;
        this.esquerda = esquerda;
        this.direita = direita;
    }


    //Variaveis auxiliares
    int maiornum = 0;
    int menornum = 0;
    int soma = 0;
    int nos = 0;
    int folhas = 0;
    int nulos = 0;


    //Funcoes


    //Funcao para imprimir a arvore
    public void imprimirPreOrdem(NoDaArvore n) {
        System.out.println(n.valor);
        //Printar caso tenha alguem a esquerda
        if (n.esquerda != null) {
            imprimirPreOrdem(n.esquerda);
        }
        //Printar caso tenha alguem a direita
        if (n.direita != null) {
            imprimirPreOrdem(n.direita);
        }
    }

    public void imprimirEmOrdem(NoDaArvore n) {
        //Printar caso tenha alguem a esquerda
        if (n.esquerda != null) {
            imprimirEmOrdem(n.esquerda);
        }
        System.out.println(n.valor);

        //Printar caso tenha alguem a direita
        if (n.direita != null) {
```

```
        imprimirEmOrdem(n.direita);
    }
}

public void imprimirPosOrdem(NoDaArvore n) {
    //Printar caso tenha alguem a esquerda
    if (n.esquerda != null) {
        imprimirPosOrdem(n.esquerda);
    }
    //Printar caso tenha alguem a direita
    if (n.direita != null) {
        imprimirPosOrdem(n.direita);
    }
    System.out.println(n.valor);
}

public void Verificacao(NoDaArvore n, int numero) {
    if (numero == n.valor) {
        System.out.println("Achei!");
    }
    if ((n.esquerda != null) && (numero != n.valor)) {
        Verificacao(n.esquerda, numero);
    }
    if ((n.direita != null) && (numero != n.valor)) {
        Verificacao(n.direita, numero);
    }

}


public void MaiorNum(NoDaArvore n) {

    if (n.esquerda != null) {
        MaiorNum(n.esquerda);
    }
    if (n.direita != null) {
        MaiorNum(n.direita);
    }
    if (n.valor > maiornum) {
        maiornum = n.valor;
    }
}


public void MenorNum(NoDaArvore n) {
    if (n.esquerda != null) {
        MenorNum(n.esquerda);
    }
    if (n.direita != null) {
        MenorNum(n.direita);
    }
    if (n.valor < menornum) {
        menornum = n.valor;
    }
}
```

```
public void Soma(NoDaArvore n) {

    if (n.esquerda != null) {
        Soma(n.esquerda);
    }
    if (n.direita != null) {
        Soma(n.direita);
    }
    soma += n.valor;
}


public void ContarNos(NoDaArvore n) {

    if ((n.esquerda != null)) {
        ContarNos(n.esquerda);
    }
    if ((n.direita != null)) {
        ContarNos(n.direita);
    }
    nos += 1;

}

public void Media(NoDaArvore n) {
    n.Soma(n);
    n.ContarNos(n);
    int med = n.getSoma() / n.getNos();
    System.out.println(med);
}


public void ContaFolhas(NoDaArvore n) {
    if (n.esquerda != null) {
        ContaFolhas(n.esquerda);
    }
    if ((n.direita != null)) {
        ContaFolhas(n.direita);
    }
    if ((n.esquerda == null) && (n.direita == null)) {
        folhas += 1;
    }
}


public void ContarNull(NoDaArvore n) {
    if (n.esquerda != null) {
        ContarNull(n.esquerda);
    }
    if ((n.direita != null)) {
        ContarNull(n.direita);
    }
    if (n.esquerda == null) {
        nulos += 1;
    }
```

```
        if (n.direita == null) {
            nulos += 1;
        }
    }


    public int altura(NoDaArvore n) {
        if (n != null) {
            int he, hd;

            he = altura(n.esquerda);
            hd = altura(n.direita);

            if (he > hd) {
                return he + 1;
            } else {
                return hd + 1;
            }
        }
        return 0;
    }


    //GETTERS

    public int getMaiornum() {
        return maiornum;
    }

    public int getMenornum() {
        return menornum;
    }

    public int getSoma() {
        return soma;
    }

    public int getNos() {
        return nos;
    }

    public int getFolhas() {
        return folhas;
    }

    public int getNulos() {
        return nulos;
    }
}
```

Questão 2

```
Pre-order
0
1
2
3
5
6
4
Em Ordem
1
0
5
3
6
2
4
Pos Ordem
1
5
6
3
4
2
0
```

Questão 3

```
  public void Verificacao(NoDaArvore n, int numero) {
        if (numero == n.valor) {
            System.out.println("Achei!");
        }
        if ((n.esquerda != null) && (numero != n.valor)) {
            Verificacao(n.esquerda, numero);
        }
        if ((n.direita != null) && (numero != n.valor)) {
            Verificacao(n.direita, numero);
        }

    }
```

Questão 4

```
public void MaiorNum(NoDaArvore n) {

        if (n.esquerda != null) {
            MaiorNum(n.esquerda);
        }
        if (n.direita != null) {
            MaiorNum(n.direita);
```

```
        }
        if (n.valor > maiornum) {
            maiornum = n.valor;
        }
    }
```

Questão 5

```
public void MenorNum(NoDaArvore n) {
        if (n.esquerda != null) {
            MenorNum(n.esquerda);
        }
        if (n.direita != null) {
            MenorNum(n.direita);
        }
        if (n.valor < menornum) {
            menornum = n.valor;
        }
    }
```

Questão 6

```
    public void media(NoDaArvore n) {
        n.Soma(n);
        n.ContarNos(n);
        int med = n.getSoma() / n.getNos();
        System.out.println(med);
    }
```

Questão 7

```
 public void ContarNull(NoDaArvore n) {
        if (n.esquerda != null) {
            ContarNull(n.esquerda);
        }
        if ((n.direita != null)) {
            ContarNull(n.direita);
        }
        if (n.esquerda == null) {
            nulos += 1;
        }
        if (n.direita == null) {
            nulos += 1;
```

```
        }
    }
```

Questão 8

```
public void ContarNos(NoDaArvore n) {

        if ((n.esquerda != null)) {
            ContarNos(n.esquerda);
        }
        if ((n.direita != null)) {
            ContarNos(n.direita);
        }
        nos += 1;

    }
```

Questão 9

```
 public void ContaFolhas(NoDaArvore n) {
        if (n.esquerda != null) {
            ContaFolhas(n.esquerda);
        }
        if ((n.direita != null)) {
            ContaFolhas(n.direita);
        }
        if ((n.esquerda == null) && (n.direita == null)) {
            folhas += 1;
        }
    }
```

Questão 10

```
    public int altura(NoDaArvore n) {
        if (n != null) {
            int he, hd;

            he = altura(n.esquerda);
            hd = altura(n.direita);

            if (he > hd) {
                return he + 1;
            } else {
                return hd + 1;
```

```
        }
    }
    return 0;
}
```