**Listagem 1:** Exemplo de inserção em árvores binárias

|  |
| --- |
| public void inserir(No node, int valor) {    //verifica se a árvore já foi criada     if (node != null) {      //Verifica se o valor a ser inserido é menor que o nodo corrente da árovre, se sim vai para subarvore esquerda      if (valor < node.valor) {          //Se tiver elemento no nodo esquerdo continua a busca          if (node.esquerda != null) {              inserir(node.esquerda, valor);          } else {              //Se nodo esquerdo vazio insere o novo nodo aqui              System.out.println("  Inserindo " + valor + " a esquerda de " + node.valor);              node.esquerda = new No(valor);          }      //Verifica se o valor a ser inserido é maior que o nodo corrente da árvore, se sim vai para subarvore direita      } else if (valor > node.valor) {          //Se tiver elemento no nodo direito continua a busca          if (node.direita != null) {              inserir(node.direita, valor);          } else {              //Se nodo direito vazio insere o novo nodo aqui              System.out.println("  Inserindo " + valor + " a direita de " + node.valor);              node.direita = new No(valor);          }      }    }  } |
|  |

**Listagem 2:** Exemplo de exclusão em árvores binárias

|  |
| --- |
| public No removeValorMinimoDaArvore(No node) {      if (node == null) {          System.out.println("  ERRO ");      } else if (node.esquerda != null) {          node.esquerda = removeValorMinimoDaArvore(node.esquerda);          return node;      } else {          //Se não tiver elemento esquerdo só nós resta o da direita          return node.direita;      }      return null;  } |

**Listagem 3:** Exemplo de caminhamento prefixado em árvores binárias

|  |
| --- |
| public void prefixado(No no) {      if(no != null){          System.out.print(no.valor + " ");          prefixado(no.esquerda);          prefixado(no.direita);      }  } |

**Listagem 4**: Exemplo de caminhamento posfixado em árvores binárias

|  |
| --- |
| public void posfixado(No no) {      if(no != null){          posfixado(no.esquerda);          posfixado(no.direita);          System.out.print(no.valor + " ");      }  } |

**Listagem 5:** Exemplo de caminhamento em ordem em árvores binárias

|  |
| --- |
| public void emordem(No no) {      if(no != null){          emordem(no.esquerda);      System.out.print(no.valor + " ");          emordem(no.direita);      }  } |

**public** **class** **TesteArvoreBinaria** {

**public** **static** **void** **main**(String[] args) {

**new** TesteArvoreBinaria().run();

}

**static** **class** **No** {

No esquerda;

No direita;

**int** valor;

**public** **No**(**int** valor) {

**this**.valor = valor;

}

}

**public** **void** **run**() {

No raiz = **new** No(20);

System.**out**.println("Exemplo de Arvore Binaria");

System.**out**.println("Criando arvore com a raiz " + raiz.valor);

inserir(raiz, 22);

inserir(raiz, 6);

inserir(raiz, 15);

inserir(raiz, 8);

inserir(raiz, 17);

inserir(raiz, 7);

inserir(raiz, 3);

inserir(raiz, 11);

inserir(raiz, 9);

remover(raiz, 15);

}

**public** **void** **inserir**(No node, **int** valor) {

**if** (valor < node.valor) {

**if** (node.esquerda != null) {

inserir(node.esquerda, valor);

} **else** {

System.**out**.println(" Inserindo " + valor + " a esqueda de " + node.valor);

node.esquerda = **new** No(valor);

}

} **else** **if** (valor > node.valor) {

**if** (node.direita != null) {

inserir(node.direita, valor);

} **else** {

System.**out**.println(" Inserindo " + valor + " a direita de " + node.valor);

node.direita = **new** No(valor);

}

}

}

**public** No **remover**(No node, **int** valor) {

System.**out**.println(" Corendo No " + node.valor);

**if** (node == null) {

System.**out**.println(" Arvore vazia ");

}

**if** (valor < node.valor) {

node.esquerda = remover(node.esquerda, valor);

} **else** **if** (valor > node.valor) {

node.direita = remover(node.direita, valor);

} **else** **if** (node.esquerda != null && node.direita != null) *// 2 filhos*

{

System.**out**.println(" Removeu No " + node.valor);

node.valor = encontraMinimo(node.direita).valor;

node.direita = removeMinimo(node.direita);

} **else** {

System.**out**.println(" Removeu No " + node.valor);

node = (node.esquerda == null) ? node.esquerda : node.direita;

}

**return** node;

}

**public** No **removeMinimo**(No node) {

**if** (node == null) {

System.**out**.println(" ERRO ");

} **else** **if** (node.esquerda != null) {

node.esquerda = removeMinimo(node.esquerda);

**return** node;

} **else** {

**return** node.direita;

}

**return** null;

}

**public** No **encontraMinimo**(No node) {

**if** (node != null) {

**while** (node.esquerda != null) {

node = node.esquerda;

}

}

**return** node;

}

}

public class ArvoreBinaria {

    private No raiz;

    private ArvoreBinaria arvoreEsquerda;

    private ArvoreBinaria arvoreDireita;

    public ArvoreBinaria() { }

    public ArvoreBinaria getArvoreDireita() {

        return arvoreDireita;

    }

    public void setArvoreDireita(ArvoreBinaria arvoreDireita) {

        this.arvoreDireita = arvoreDireita;

    }

    public ArvoreBinaria getArvoreEsquerda() {

        return arvoreEsquerda;

    }

    public void setArvoreEsquerda(ArvoreBinaria arvoreEsquerda) {

        this.arvoreEsquerda = arvoreEsquerda;

    }

    public No getRaiz() {

        return raiz;

    }

    public void setRaiz(No raiz) {

        this.raiz = raiz;

    }

    public void insereAluno(int matricula, String nome) {

        Aluno aluno = new Aluno(matricula, nome);

        No no = new No(aluno);

        inserir(no);

    }

    public void inserir(No no) {

        if (this.raiz == null) {

            this.raiz = no;

        } else {

            if (no.getAluno().getMatricula() > this.raiz.getAluno().getMatricula()) {

                if (this.arvoreDireita == null) {

                    this.arvoreDireita = new ArvoreBinaria();

                }

                this.arvoreDireita.inserir(no);

            } else if (no.getAluno().getMatricula() < this.raiz.getAluno().getMatricula()) {

                if (this.arvoreEsquerda == null) {

                    this.arvoreEsquerda = new ArvoreBinaria();

                }

                this.arvoreEsquerda.inserir(no);

            }

        }

    }

    public void percorrerInOrder() {

        if (this.raiz == null) {

           return;

        }

        if (this.arvoreEsquerda != null) {

            this.arvoreEsquerda.percorrerInOrder();

        }

        System.out.println("Matrícula: " + this.raiz.getAluno().getMatricula());

        System.out.println("Nome: " + this.raiz.getAluno().getNome());

        if (this.arvoreDireita != null) {

            this.arvoreDireita.percorrerInOrder();

        }

    }

    public void percorrerPreOrder() {

        if (this.raiz == null) {

           return;

        }

        System.out.println("Matrícula: " + this.raiz.getAluno().getMatricula());

        System.out.println("Nome: " + this.raiz.getAluno().getNome());

        if (this.arvoreEsquerda != null) {

            this.arvoreEsquerda.percorrerPreOrder();

        }

        if (this.arvoreDireita != null) {

            this.arvoreDireita.percorrerPreOrder();

        }

    }

    public void percorrerPostOrder() {

        if (this.raiz == null) {

           return;

        }

        if (this.arvoreEsquerda != null) {

            this.arvoreEsquerda.percorrerPostOrder();

        }

        if (this.arvoreDireita != null) {

            this.arvoreDireita.percorrerPostOrder();

        }

        System.out.println("Matrícula: " + this.raiz.getAluno().getMatricula());

        System.out.println("Nome: " + this.raiz.getAluno().getNome());

    }

    public Aluno busca(int matricula) {

        if (this.raiz == null) {

            return null;

        } else {

            if (matricula == this.raiz.getAluno().getMatricula()) {

                return this.raiz.getAluno();

            } else {

                if (matricula > this.raiz.getAluno().getMatricula()) {

                    if (this.arvoreDireita == null) {

                        return null;

                    }

                    return this.arvoreDireita.busca(matricula);

                } else {

                    if (this.arvoreEsquerda == null) {

                        return null;

                    }

                    return this.arvoreEsquerda.busca(matricula);

                }

            }

        }

    }

    public class No {

        private Aluno aluno;

        public No(Aluno aluno) {

            this.aluno = aluno;

        }

        public Aluno getAluno() {

            return aluno;

        }

        public void setAluno(Aluno aluno) {

            this.aluno = aluno;

        }

    }

    public class Aluno {

        private int matricula;

        private String nome;

        public Aluno(int mat, String nome) {

            this.matricula = mat;

            this.nome = nome;

        }

        public int getMatricula() {

            return matricula;

        }

        public void setMatricula(int mat) {

            this.matricula = mat;

        }

        public String getNome() {

            return nome;

        }

        public void setNome(String nome) {

            this.nome = nome;

        }

    }

}