Implementando ABB

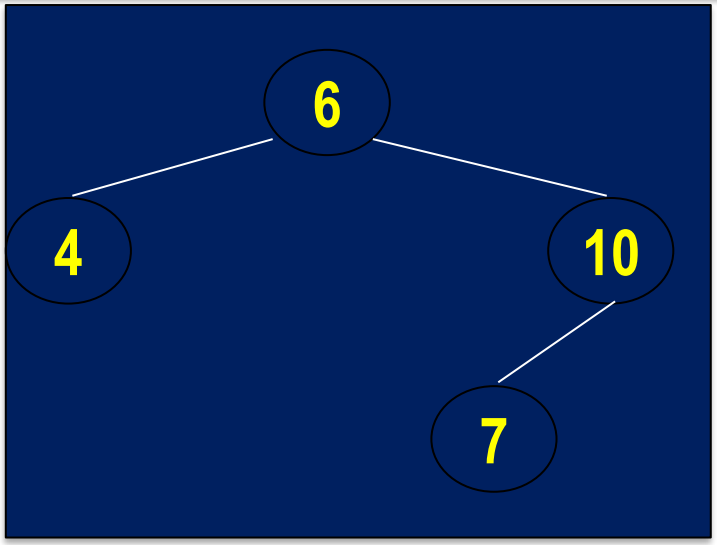
Códigos de Alta Performance

PROFa. PATRÍCIA MAGNA - profpatricia.magna@fiap.com.br

Operações Necessárias Para Implementar ABB

- Apresentação de todos os elementos (já implementado)
- Inserção de um elemento
- Consulta (pesquisa) por conteúdo
- Remoção de um elemento
- Contagem de elementos presentes
- Entre outras...

Implementação da Inserção de um Elemento na ABB

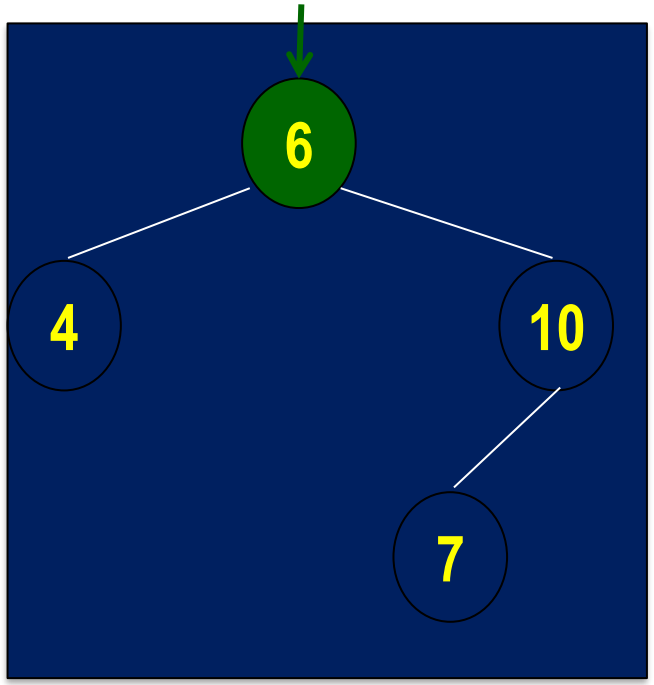



```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```

Inserindo Nó com valor 5 na ABB em JAVA

```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```

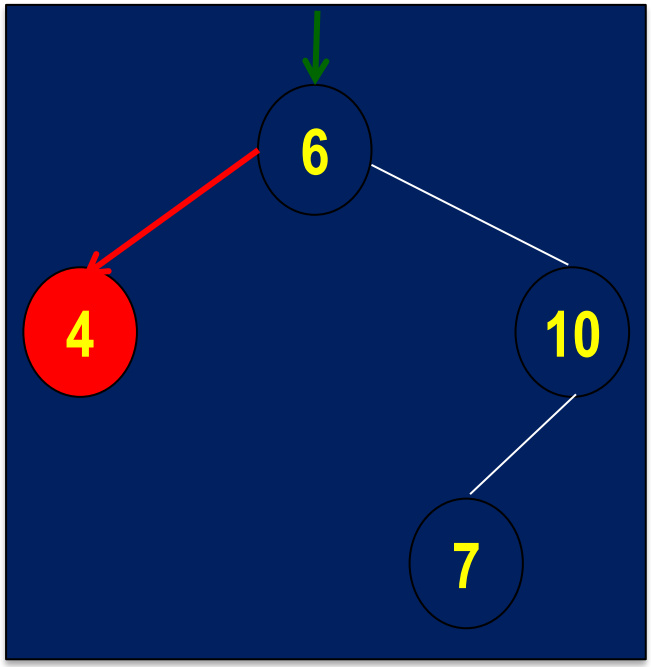
5



Implementação de ABB em JAVA

```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```

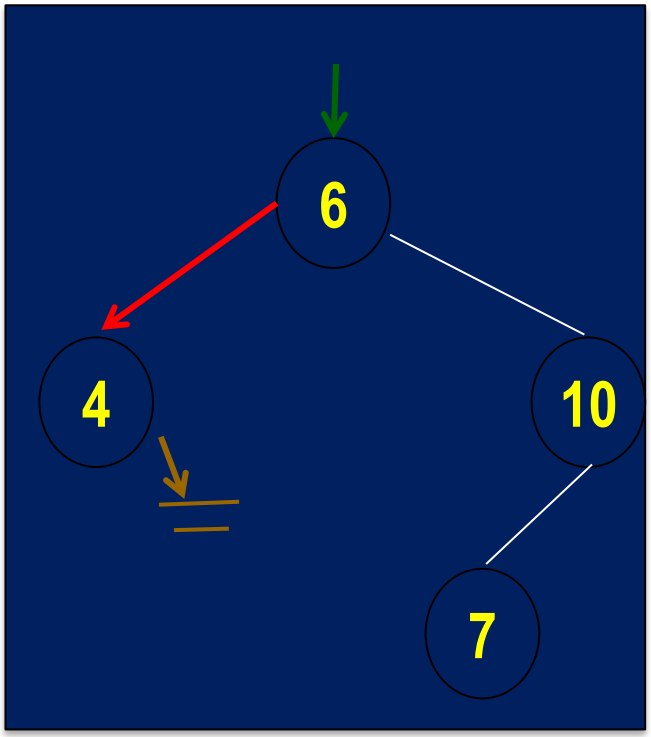
5



Inserindo Nó com valor 5 na ABB em JAVA

```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```

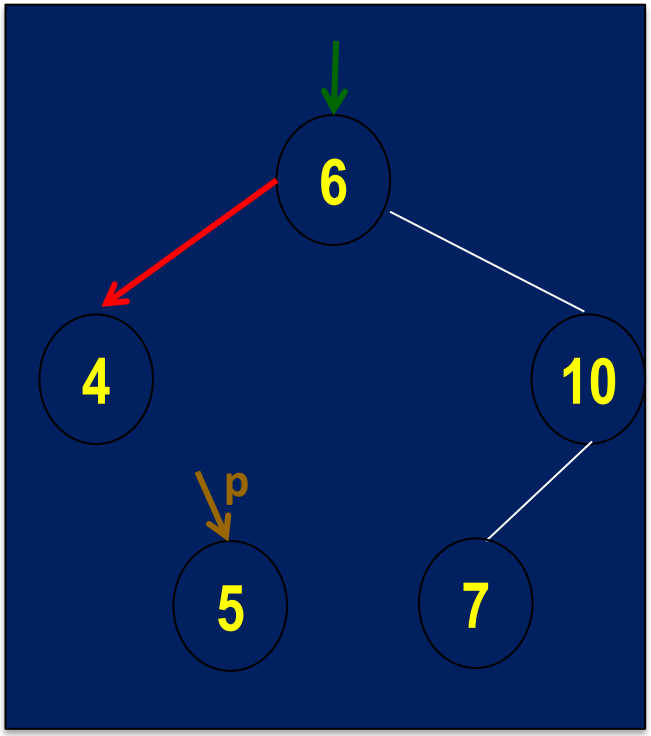
5



Inserindo Nó com valor 5 na ABB em JAVA

```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```

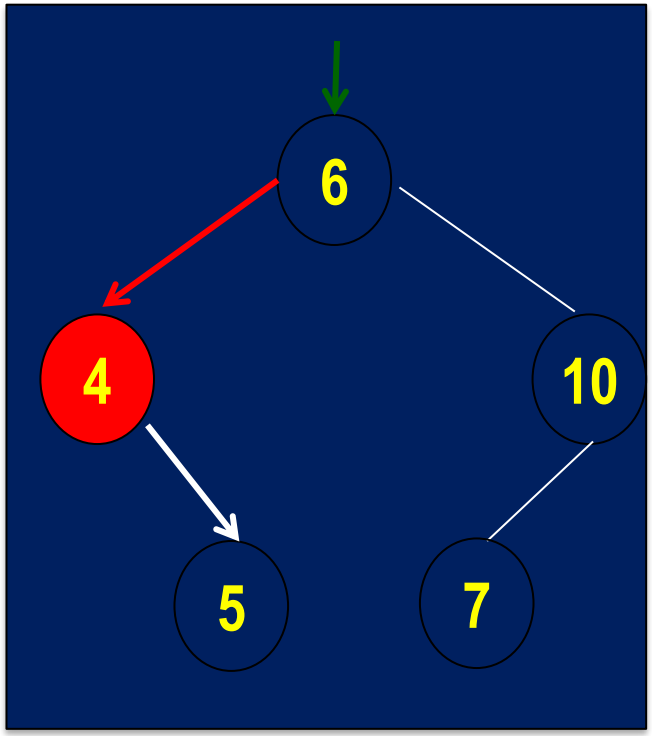
5



Inserindo Nó com valor 5 na ABB em JAVA

```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```

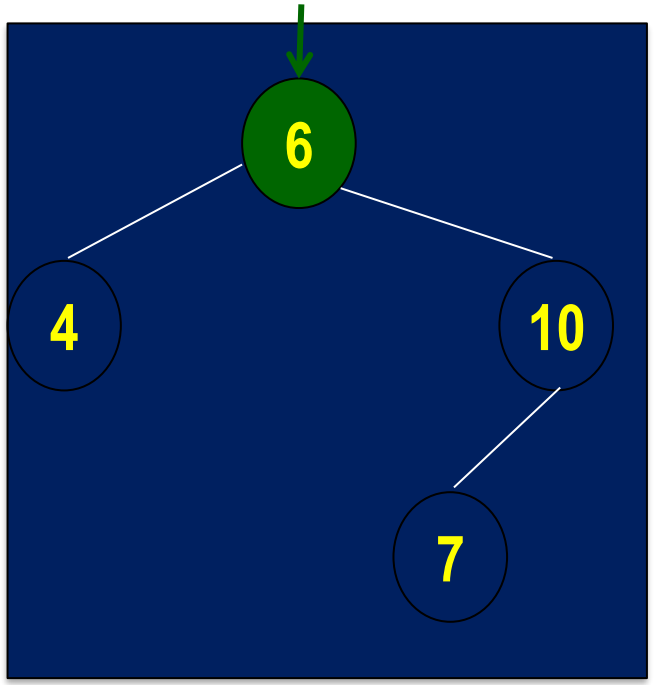
5



Inserindo Nó com valor 5 na ABB em JAVA

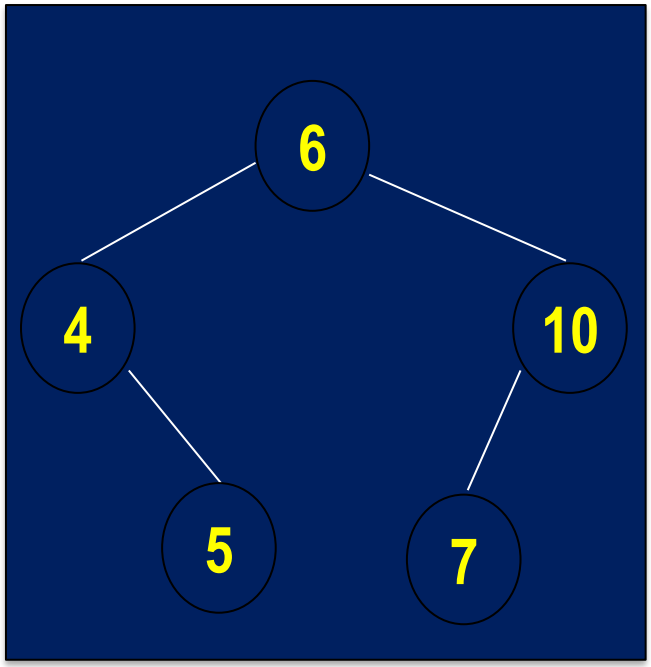
```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```

5



Inserindo Nó com valor 5 na ABB em JAVA

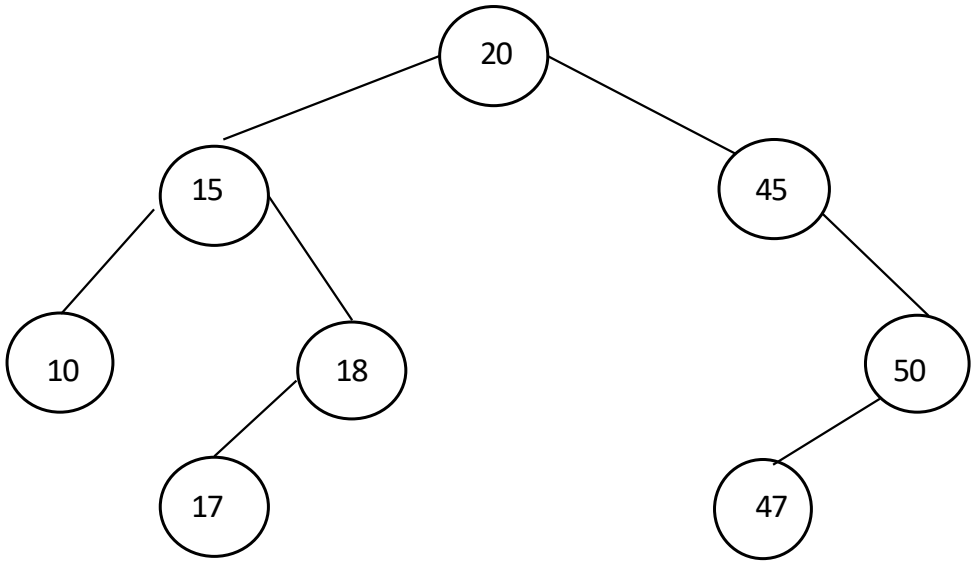
```
public ARVORE inserir(ARVORE p, int info) {
        // insere elemento em uma ABB
        if (p == null) {
                p = new ARVORE();
                p.dado = info;
                p.esq = null;
                p.dir = null;
        }
        else if (info < p.dado)
                p.esq= inserir (p.esq, info);
             else
                p.dir=inserir(p.dir, info);
        return p;
        }
```



Exercícios

1. Elabore o método **contaNos()** que retorna o número de nós presentes em uma ABB.
2. Implemente o método **consulta()** que pesquisa na ABB se um valor está presente, retornando *true*. Caso elemento não está presente na ABB o método retorna *false*.
3. Baseando na implementação do método consulta, elabore o método **contaConsulta()** que ao invés de retorna valor booleano retorne a quantidade de comparações realizadas até que o valor fosse encontrado ou não na ABB.

Remoção de Nós em uma ABB: seleção por valor

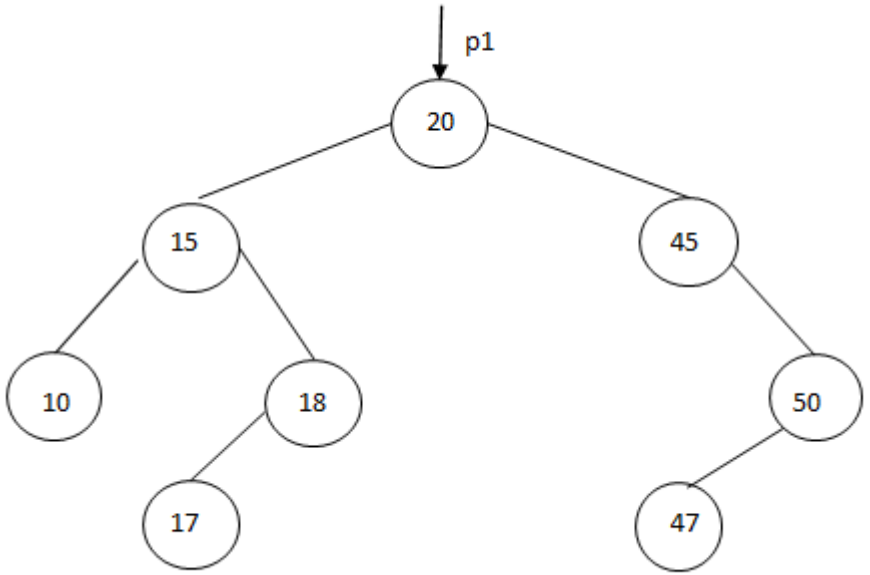


Há 3 situações que devem ser consideradas para remover um nó de uma árvore binária:

- O nó a ser removido é folha;
- O nó possui apenas 1 filho;
- O nó possui 2 filhos

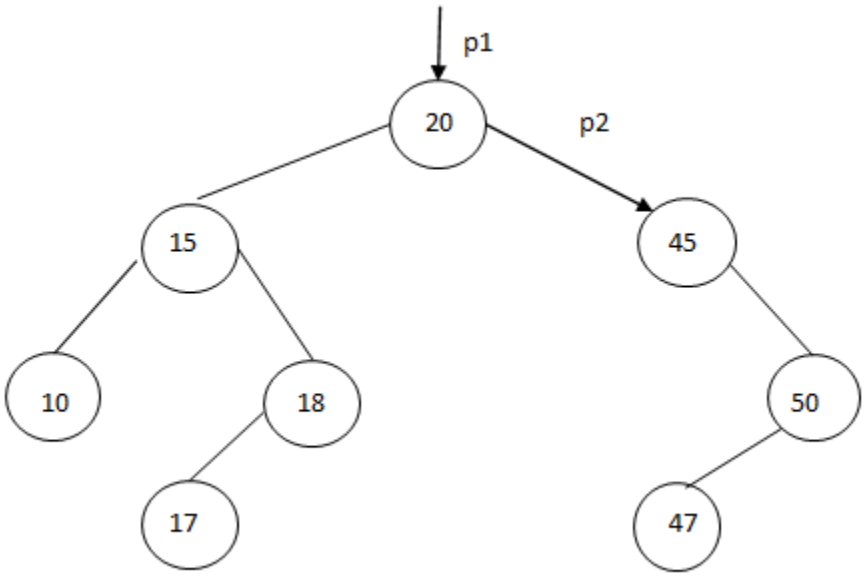
Remoção do Nó com valor 47

```
public  ARVORE removeValor (ARVORE p, int info)
if (p!=null){
  if(info == p.dado){
     if (p.esq == null && p.dir==null)
         //nó a ser removido é nó folha
         return null;
...
          }
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



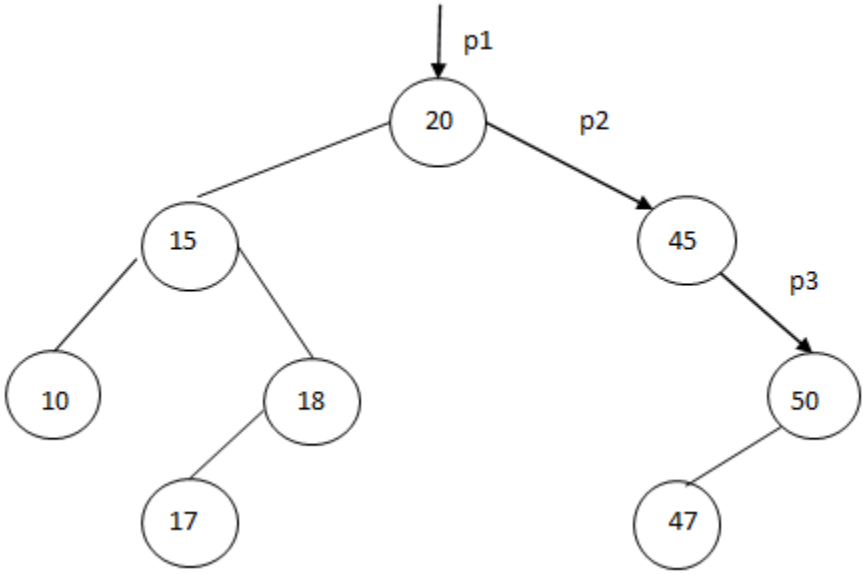
Remoção do Nó com valor 47

```
public  ARVORE removeValor (ARVORE p, int info)
if (p!=null){
  if(info == p.dado){
     if (p.esq == null && p.dir==null)
         //nó a ser removido é nó folha
         return null;
...
          }
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



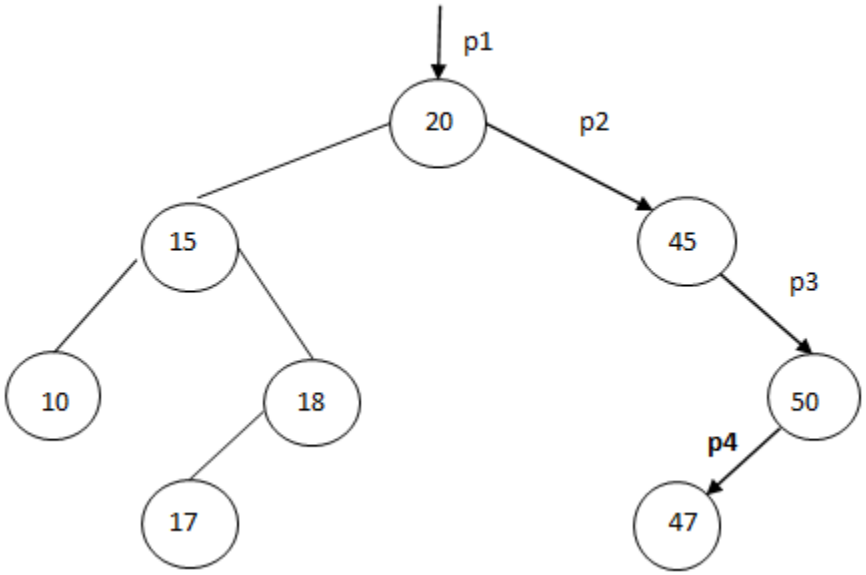
Remoção do Nó com valor 47

```
public  ARVORE removeValor (ARVORE p, int info)
if (p!=null){
  if(info == p.dado){
     if (p.esq == null && p.dir==null)
         //nó a ser removido é nó folha
         return null;
...
          }
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



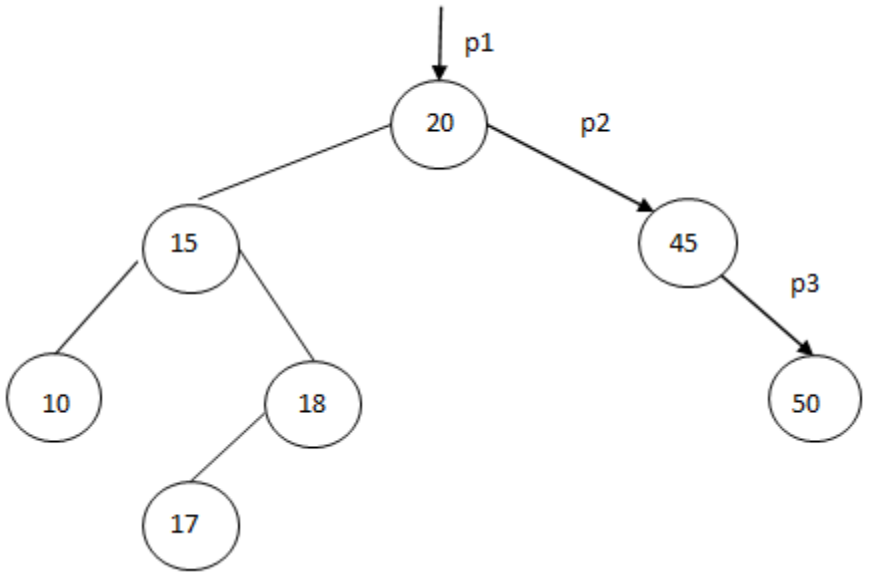
Remoção do Nó com valor 47

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
  if(info == p.dado){
     if (p.esq == null && p.dir==null)
         //nó a ser removido é nó folha
         return null;
...
          }
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



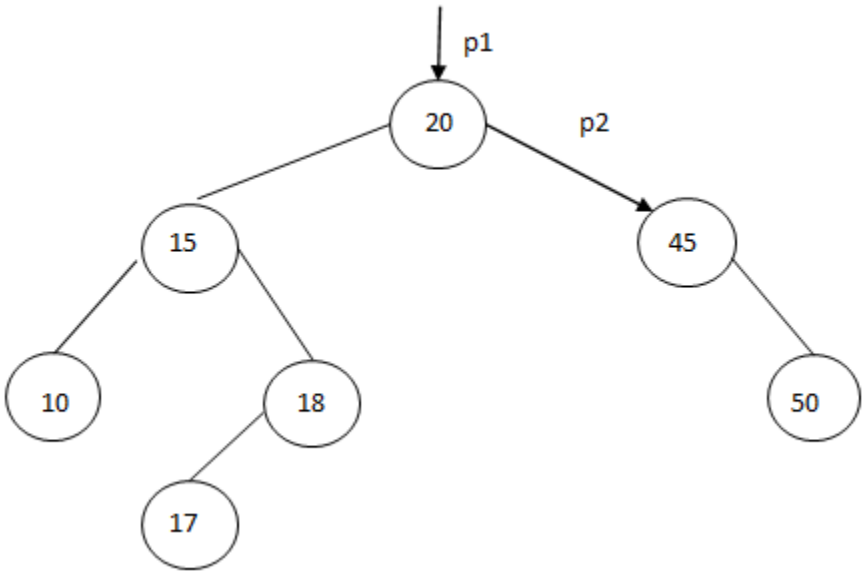
Remoção do Nó com valor 47

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
  if(info == p.dado){
     if (p.esq == null && p.dir==null)
         //nó a ser removido é nó folha
         return null;
...
         }
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



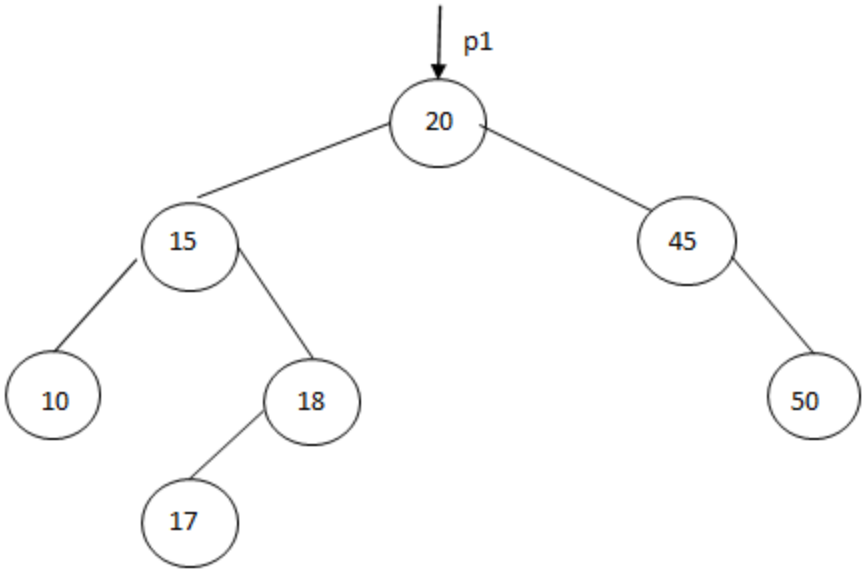
Remoção do Nó com valor 47

```
public  ARVORE removeValor (ARVORE p, int info)
if (p!=null){
  if(info == p.dado){
     if (p.esq == null && p.dir==null)
         //nó a ser removido é nó folha
         return null;
...
          }
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



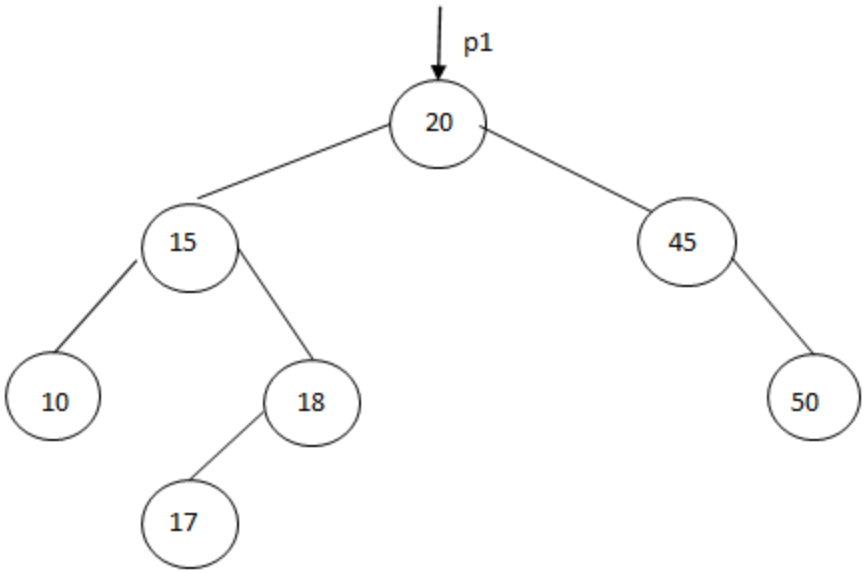
Remoção do Nó com valor 47

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
  if(info == p.dado){
     if (p.esq == null && p.dir==null)
         //nó a ser removido é nó folha
         return null;
...
          }
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



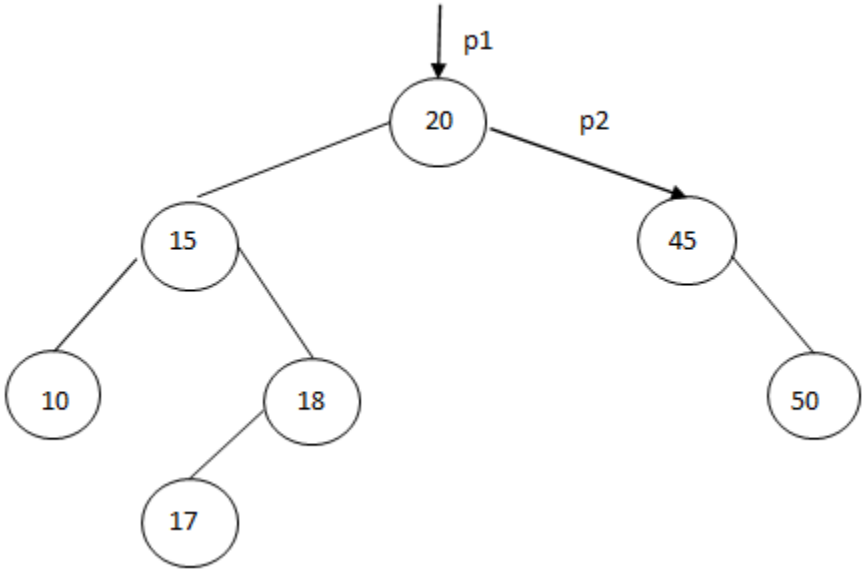
Remoção do Nó com valor 45

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
  if(info == p.dado){

   ...

   }
   else{
      //procura dado a ser removido na ABB
      if(info<p.dado)
          p.esq = removeValor(p.esq,info);
      else
          p.dir = removeValor(p.dir,info);
      }
  }
return p;
}
```



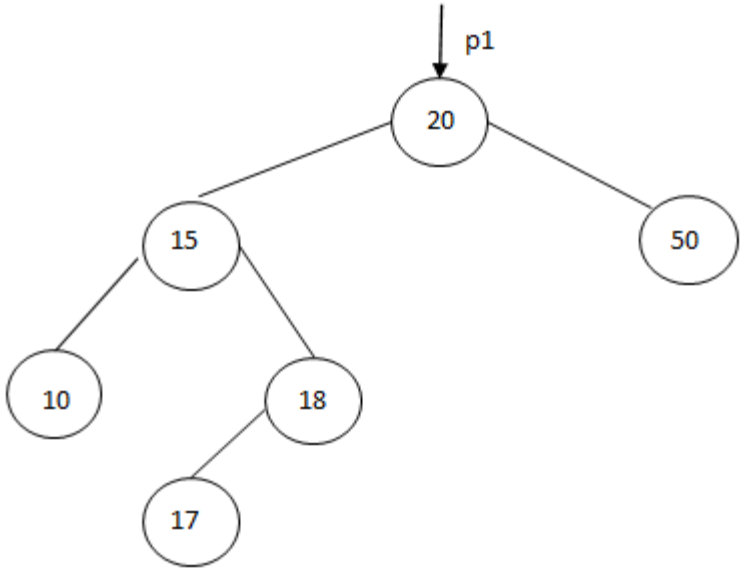
Remoção do Nó com valor 45

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
  if(info == p.dado){
    ...
  if (p.esq==null){
      return  p.dir;


 ...
   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



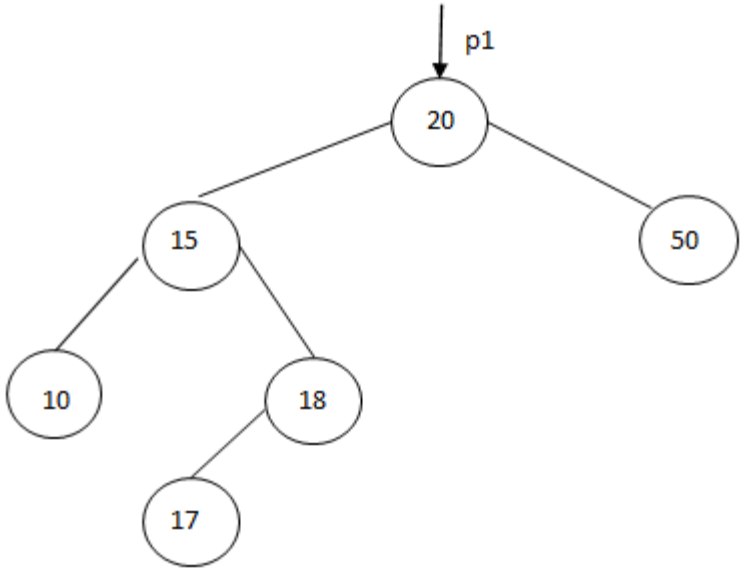
Remoção do Nó com valor 45

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
  if(info == p.dado){
   ...
  if (p.esq==null){
      return  p.dir;

 ...
   }
   else{
      //procura dado a ser removido na ABB
      if(info<p.dado)
          p.esq = removeValor(p.esq,info);
      else
          p.dir = removeValor(p.dir,info);
      }
  }
return p;
}
```



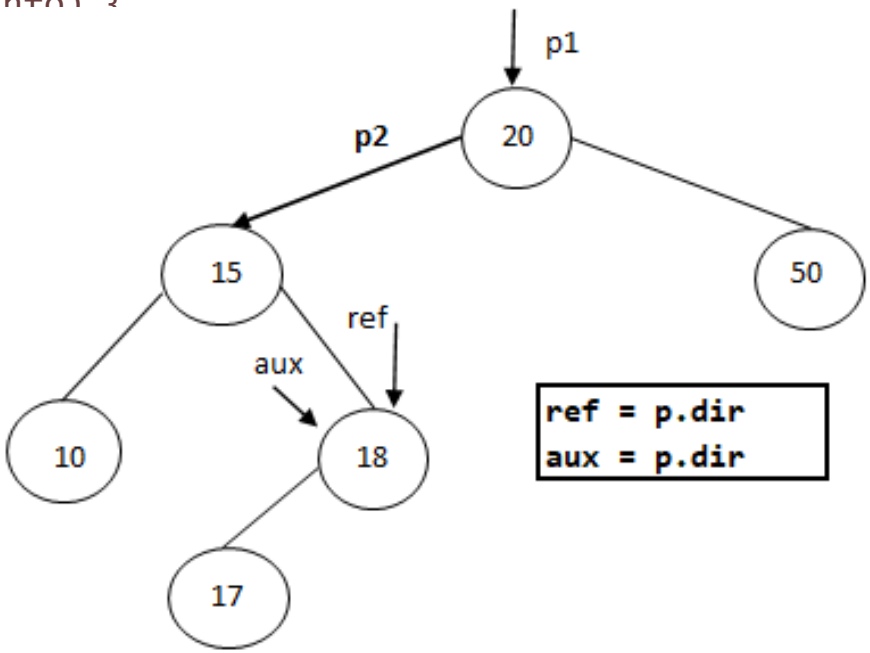
Remoção do Nó com valor 15

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
   if(info == p.dado){
   ...

   }
   else{
      //procura dado a ser removido na ABB
      if(info<p.dado)
          p.esq = removeValor(p.esq,info);
      else
          p.dir = removeValor(p.dir,info);
      }
  }
return p;
}
```



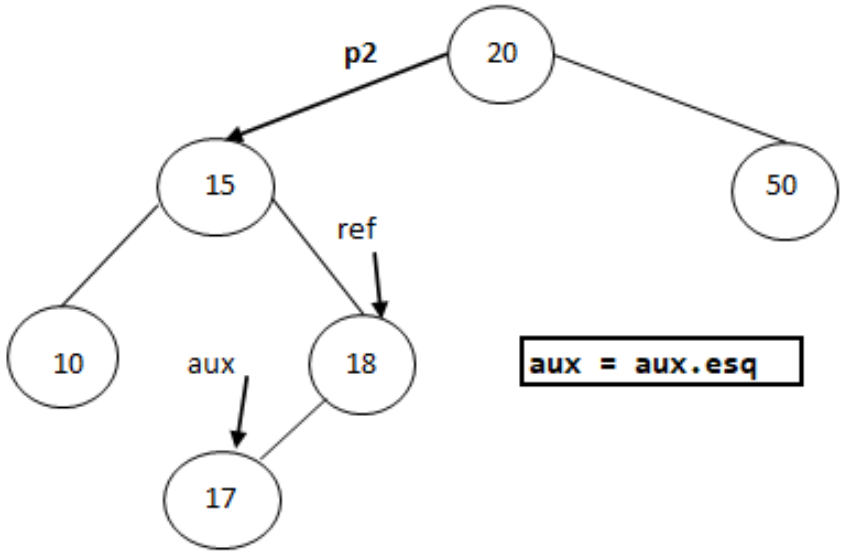
Remoção do Nó com valor 15

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
   if(info == p.dado){
   ...
   else{
        ARVORE aux, ref;
        ref = p.dir;
        aux = p.dir;
        while (aux.esq != null)
             aux = aux.esq;
        aux.esq = p.esq;
        return ref;
   }
   else{
      //procura dado a ser removido na ABB
      if(info<p.dado)
          p.esq = removeValor(p.esq,info);
      else
          p.dir = removeValor(p.dir,info);
      }
  }
return p;
}
```



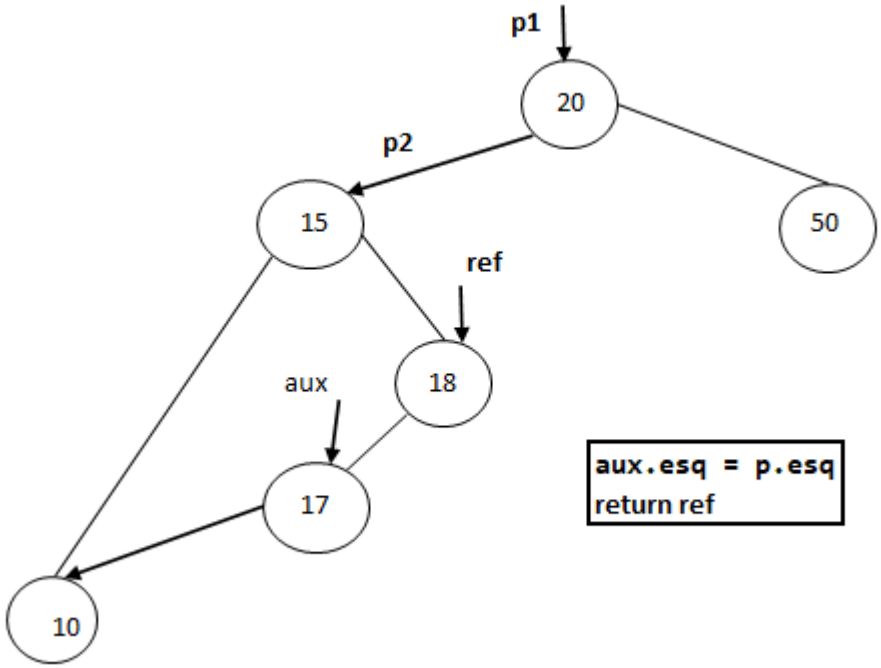
Remoção do Nó com valor 15

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
   if(info == p.dado){
   ...
   else{
       ARVORE aux, ref;
       ref = p.dir;
       aux = p.dir;
       while (aux.esq != null)
           aux = aux.esq;
       aux.esq = p.esq;
       return ref;
   }
   else{
      //procura dado a ser removido na ABB
      if(info<p.dado)
          p.esq = removeValor(p.esq,info);
      else
          p.dir = removeValor(p.dir,info);
      }
  }
return p;
}
```



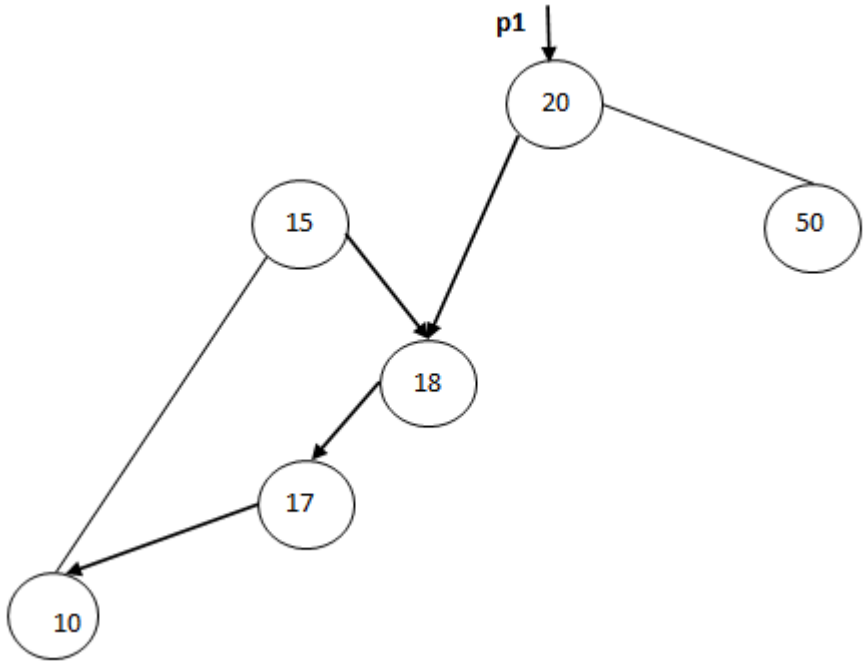
Remoção do Nó com valor 15

```
public  ARVORE removeValor (ARVORE p, int info) {
if (p!=null){
   if(info == p.dado){
   ...
   else{
       ARVORE aux, ref;
       ref = p.dir;
       aux = p.dir;
       while (aux.esq != null)
           aux = aux.esq;
       aux.esq = p.esq;
       return ref;
   }
   else{
      //procura dado a ser removido na ABB
      if(info<p.dado)
          p.esq = removeValor(p.esq,info);
      else
          p.dir = removeValor(p.dir,info);
      }
  }
return p;
}
```

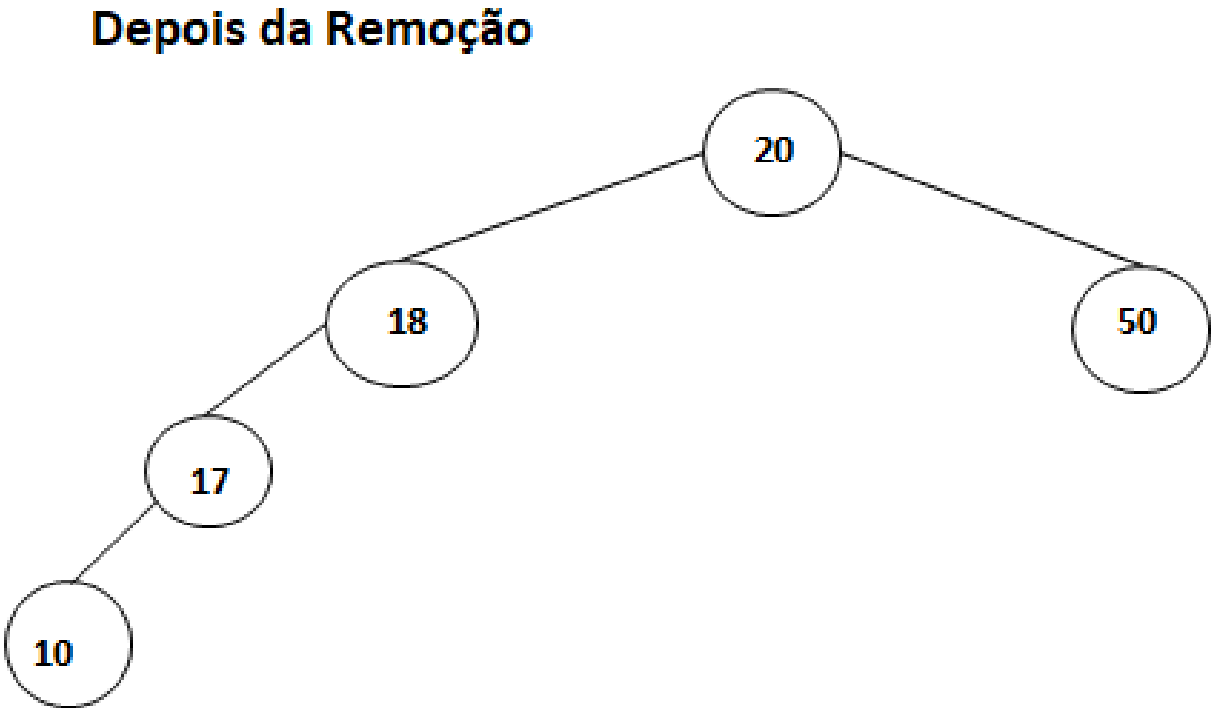


Remoção do Nó com valor 15

```
public  ARVORE removeValor (ARVORE p, int
if (p!=null){
   if(info == p.dado){
   ...
    else{
       ARVORE aux, ref;
       ref = p.dir;
       aux = p.dir;
       while (aux.esq != null)
           aux = aux.esq;
       aux.esq = p.esq;
       return ref;

   }
   else{
     //procura dado a ser removido na ABB
     if(info<p.dado)
         p.esq = removeValor(p.esq,info);
     else
         p.dir = removeValor(p.dir,info);
     }
  }
return p;
}
```



Remoção do Nó com valor 15



Método para Remoção de 1 Nó da ABB

```
public  ARVORE removeValor (ARVORE p, int info) {
  if (p!=null){
     if(info == p.dado){
       if (p.esq == null && p.dir==null)   //nó a ser removido é nó folha
           return null;
       if (p.esq==null){ //se não há sub-árvore esquerda o ponteiro passa apontar para a sub-árvore direita
           return  p.dir;
       }
       else{
           if (p.dir==null){   //se não há sub-árvore direita  o ponteiro passa apontar para a sub-árvore esquerda
             return p.esq;
           }
           else{  /*o nó a ser retirado possui sub-arvore esquerda e direita, então o nó que
                  será retirado deve-se encontrar o menor valor na sub-árvore á direita */
               ARVORE aux, ref;
               ref = p.dir;
               aux = p.dir;
               while (aux.esq != null)
                   aux = aux.esq;
               aux.esq = p.esq;
               return ref;
               }
      }
  }
  else{   //procura dado a ser removido na ABB
     if(info<p.dado)
           p.esq = removeValor(p.esq,info);
     else
           p.dir = removeValor(p.dir,info);
     }
  }
  return p;
  }
```

Exercícios

4. Elabore o método **maximo()** que retorna o maior valor armazenado em uma ABB que armazena números inteiros.
5. Elabore o método **minimo()** que retorna o menor valor armazenado em uma ABB que armazena números inteiros.
6. Construa um programa que crie uma ABB para armazenar valores inteiros e que possua um menu com as seguintes opções:
    1. Insere 1 valor na ABB;
    2. Apresenta em ordem os elementos da ABB;
    3. Consulta se um valor está presente na ABB;
    4. Apresenta o intervalo de valores (maior e menor) presentes na ABB;
    5. Remove um nó escolhido pelo seu valor.
    6. Sair do programa.

Referências Bibliográficas

•ASCÊNCIO, A.F.G; ARAUJO, G.S. **Estruturas de Dados: Algoritmos, Análise de Complexidade e Implementações em JAVA e C/C++. São Paulo, Ed.Pearson Prentice Hall, 2010.**

•TENEMBAUM, A.M et al.; **Estruturas de Dados usando C. Makron Books Ltda, 1995.**

•DEITEL, P; J.; Deitel, H.M., **Java: como programar - 8ª edição, São Paulo, Ed.Pearson Prentice Hall, 2010.**