Estruturas de dados II

Árvore 2-3

André Tavares da Silva

andre.silva@udesc.br

UDESC

Árvores 2-3

- Estrutura de árvore autobalanceada para dados classificados
  - Acesso sequencial
  - Acesso aleatório (pesquisa)

- Possui complexidade de tempo logarítmo
  - Acesso aleatório: O(log n)
  - Inserção: O(log n)
  - Remoção: O(log n)

- Idealizada por J. E. Hopcroft em 1970
  - São árvores com ótima isometria – cada direita, centro ou esquerda contém aproximadamente a mesma quantidade de dados.

UDESC

Árvores 2-3 – Características

- Cada nó pode ter uma ou duas chaves;
- Todas as folhas estão no mesmo nível;
- Caso um nó não folha tiver uma única chave ela terá duas sub-árvores, como uma árvore ABB:
  - Filho à esquerda são menores que o pai
  - Filho à direita são maiores que o pai
- Caso um nó tenha duas chaves, ela possuirá três sub-árvores:
  - Filho à esquerda é menor que a menor chave
  - Filho à direita é maior que a maior chave
  - Filho central tem valor entre a maior e a menor chave
- Pode ser chamada de árvore multivias por não ser binária

UDESC

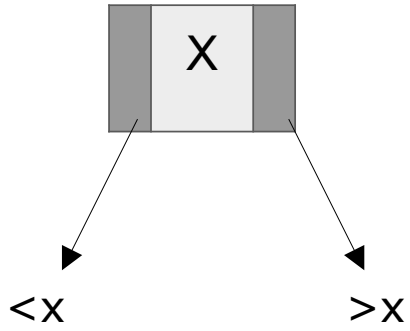
Conceitos da árvores 2-3

- Propriedades da árvore
  - Todos os nós folhas estão no mesmo nível
  - A árvore cresce para a raiz, diferente das árvores binárias anteriores (ABB, AVL e Rubro-Negra) que crescem sempre nas folhas
  - As chaves de um nó com chave dupla são ordenados de forma crescente

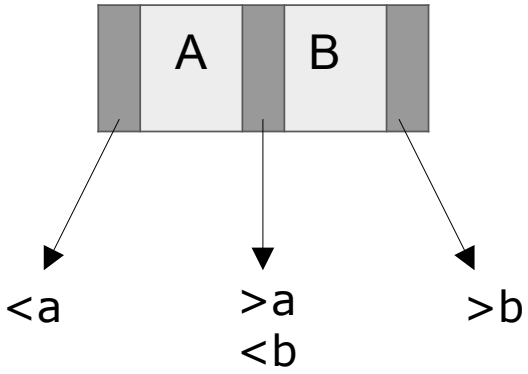
- Operações em uma árvore 2-3
  - Adicionar uma chave
  - Remover uma chave
  - Localizar uma chave
  - Percorrer a árvore

Estrutura da árvores 2-3

Nó com uma chave

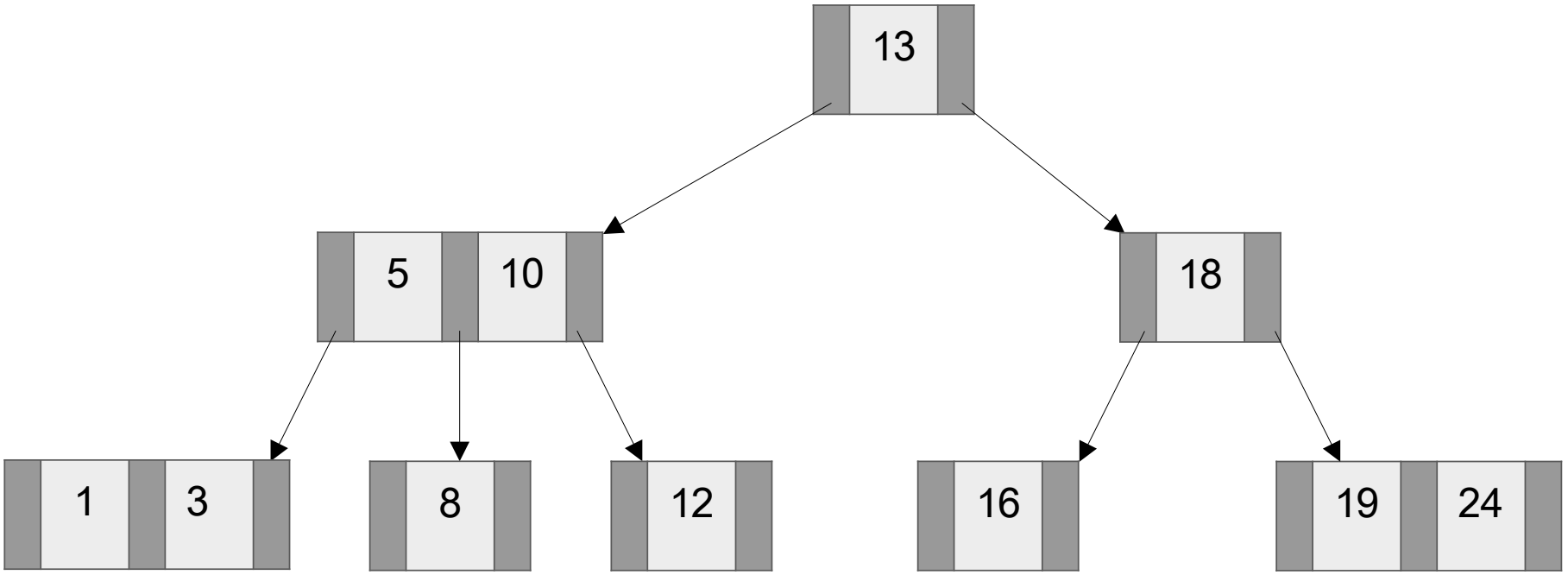


Nó com duas chaves



UDESC

Estrutura da árvores 2-3



UDESC

Inserção em árvores 2-3

- Regra 1: Se a árvore estiver vazia:
  - Cria um nó e adiciona o valor ao nó.
- Se a árvore não estiver vazia, localiza o nó em que o valor pertence e utiliza as demais regras;
- Regra 2: Se o nó tiver apenas um valor:
  - Transforma o nó em duplo e insere o valor;
  - O balanceamento da árvore não sofre alteração.
- Regra 3: Se o nó folha tiver dois valores:
  - Chave com valor mediano é promovido para o pai;
  - Caso seja o nó raiz, cria uma nova raiz e divide o nó mantendo o balanceamento;
  - Caso o nó pai (não raiz) também tenha dois valores, a operação é repetida até encontrar um nó com apenas um valor ou até chegar na raiz.

UDESC

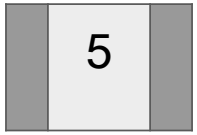
Exemplo – árvore vazia:

Insere 5

UDESC

Exemplo – árvore vazia:

Insere 5

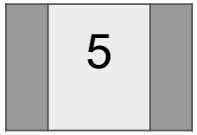


Regra 1: árvore vazia

UDESC

Exemplo – Inserção em nó com chave única:

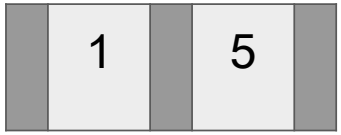
Insere 1



UDESC

Exemplo – Inserção em nó com chave única:

Insere 1

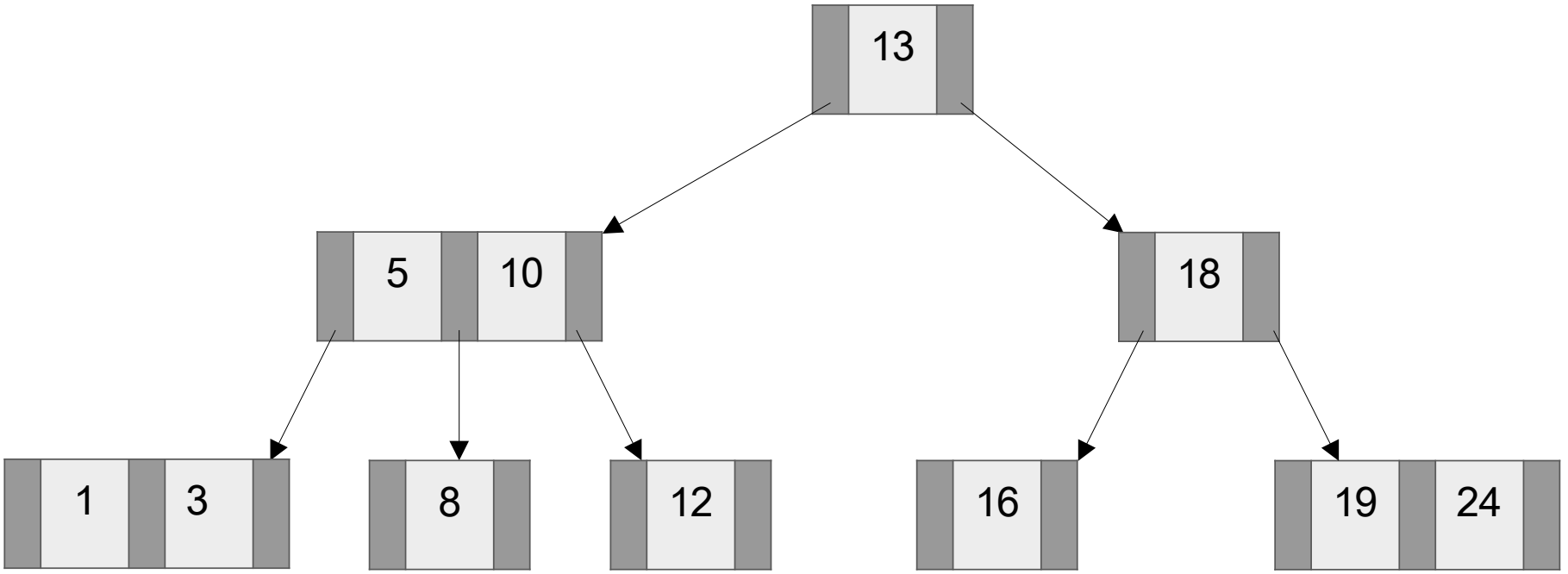


Regra 2: inserção em nó de valor único

UDESC

Exemplo:

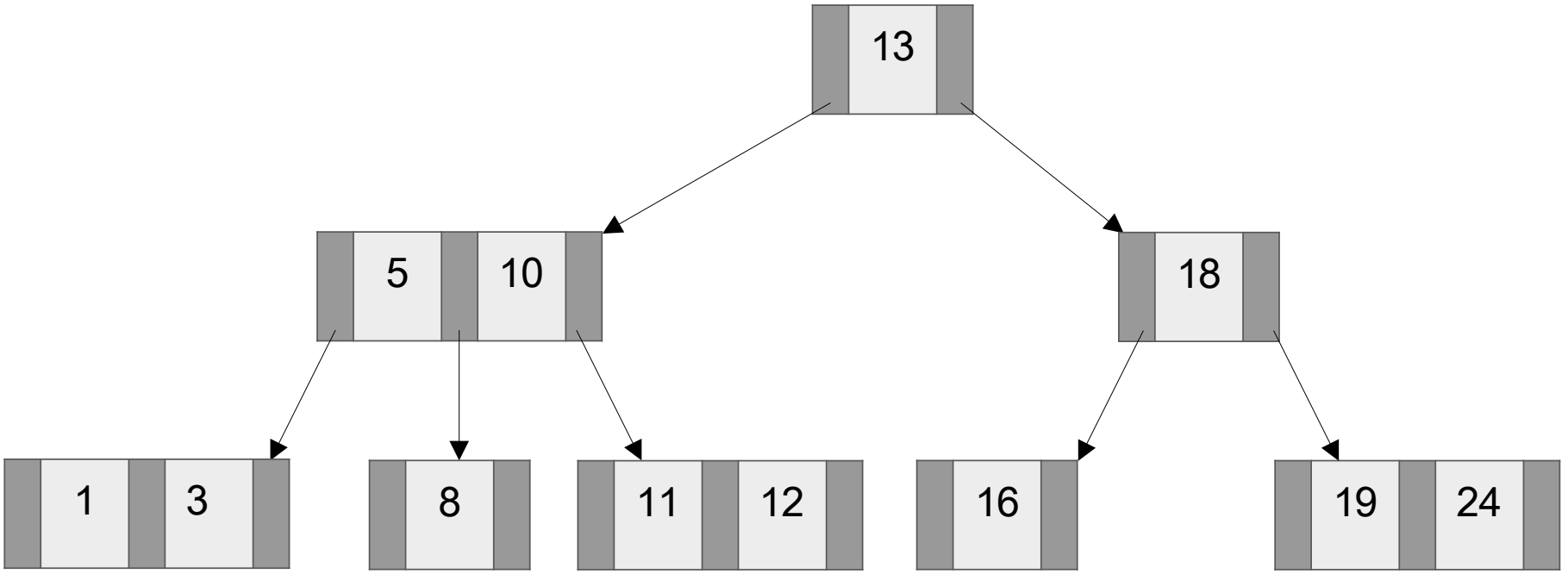
Insere 11



UDESC

Exemplo:

Insere 11

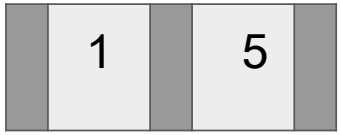


Regra 2: inserção em
nó de valor único

UDESC

Exemplo – Inserção em nó duplo raiz:

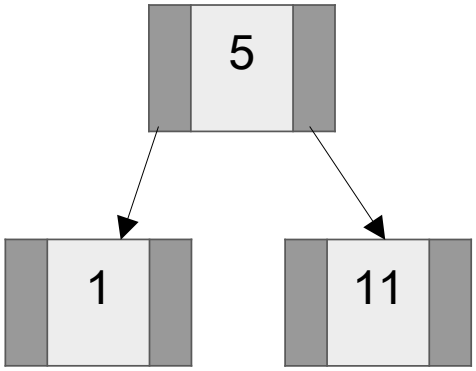
Insere 11



UDESC

Exemplo – Inserção em nó duplo raiz:

Insere 11

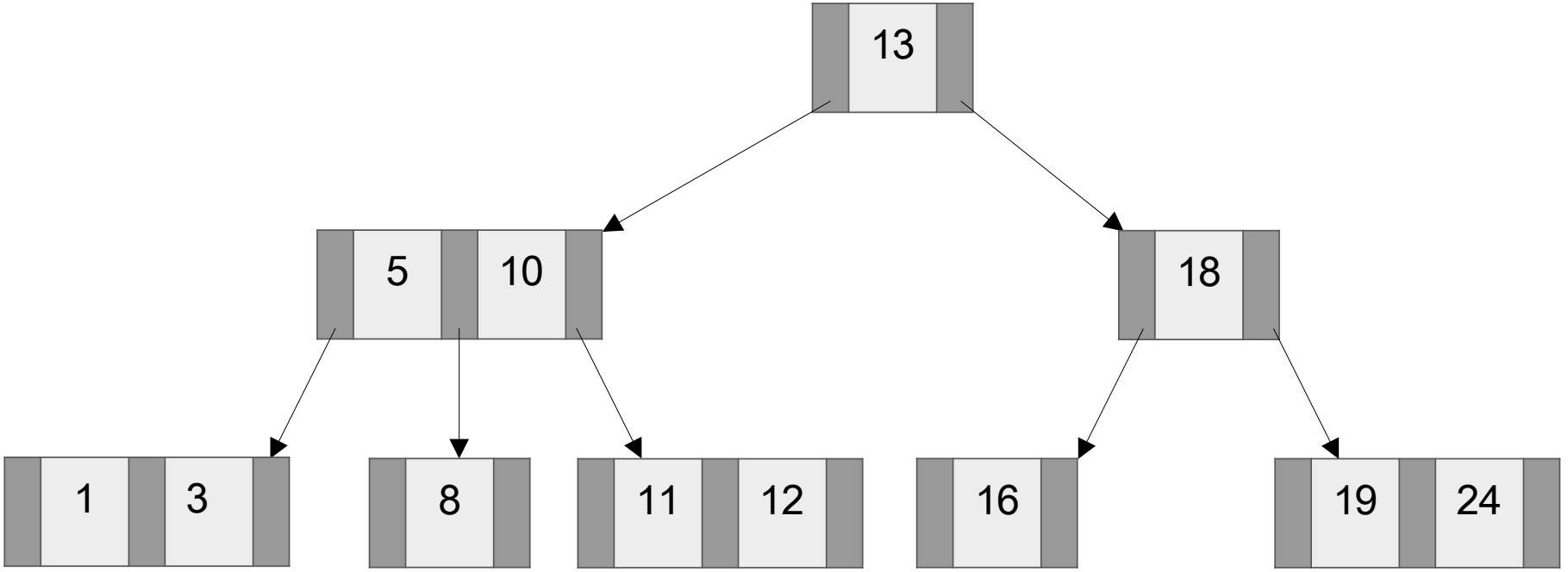


Regra 3: Se o nó folha tiver dois valores:
- Chave com valor mediano é promovido para o pai;
- Caso seja o nó raiz, cria uma nova raiz e divide o nó mantendo o balanceamento;

UDESC

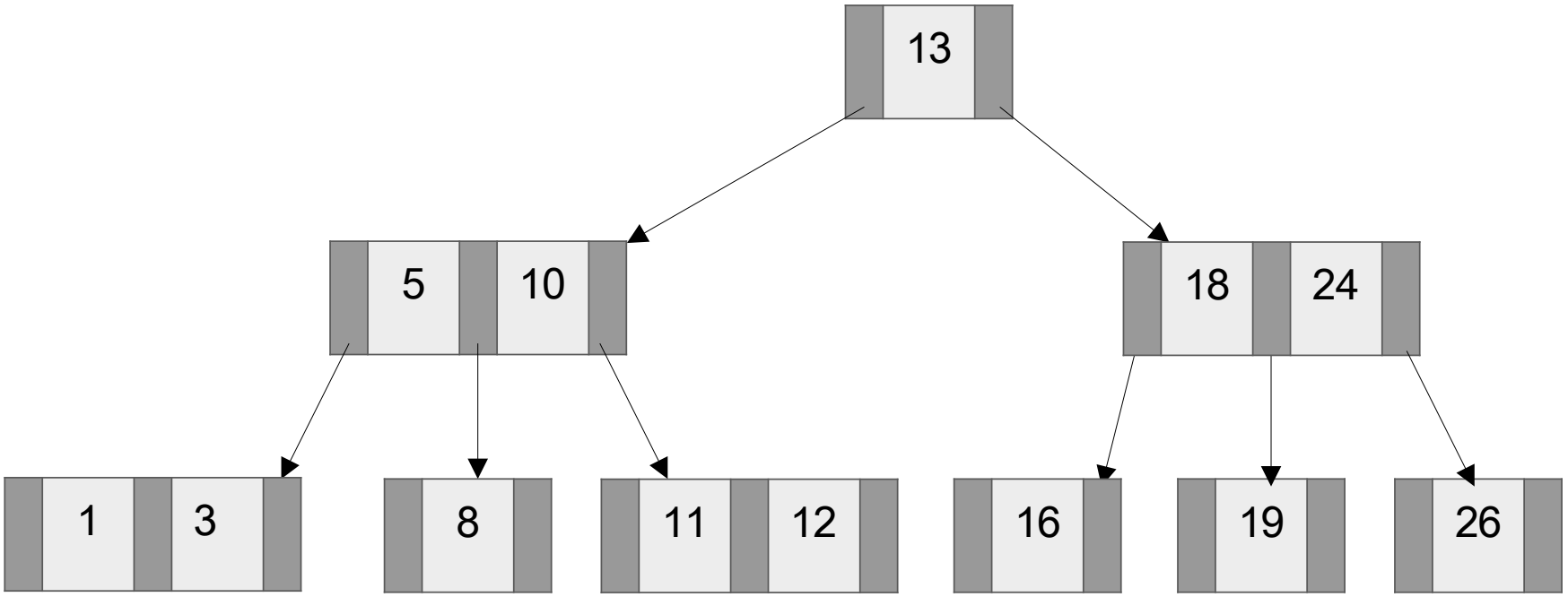
Exemplo – Inserção em nó duplo raiz:

Insere 26



Exemplo – Inserção em nó duplo raiz:

Insere 26

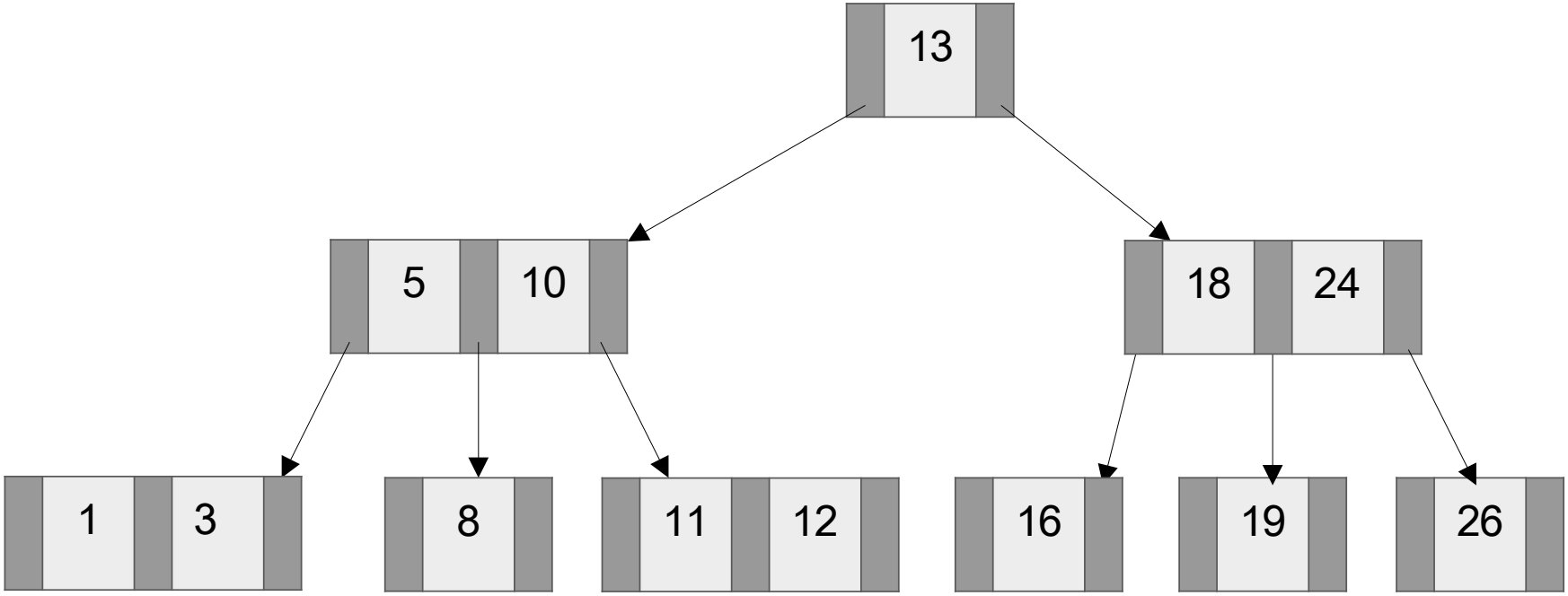


Regra 3: inserção em nó folha duplo
- Chave com valor mediano é promovido para o pai

UDESC

Exemplo – Inserção em nó duplo com pai duplo:

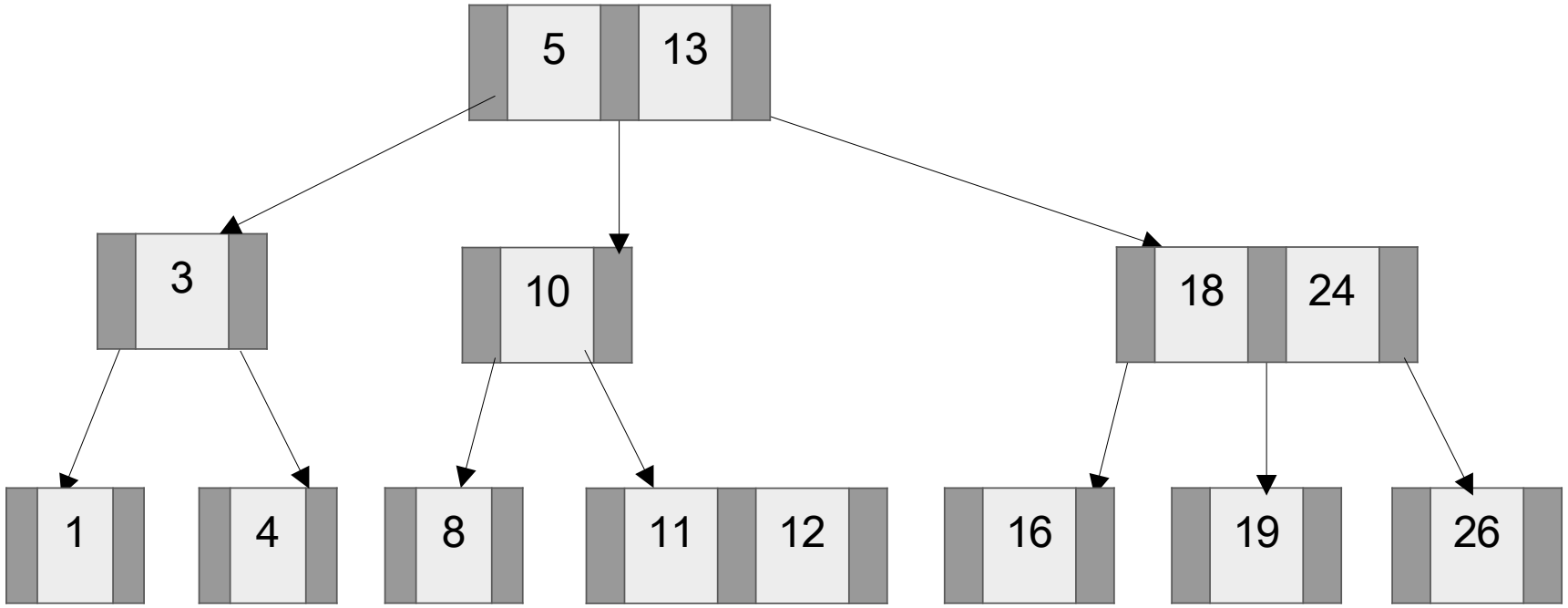
Insere 4



UDESC

Exemplo – Inserção em nó duplo com pai duplo:

Insere 4

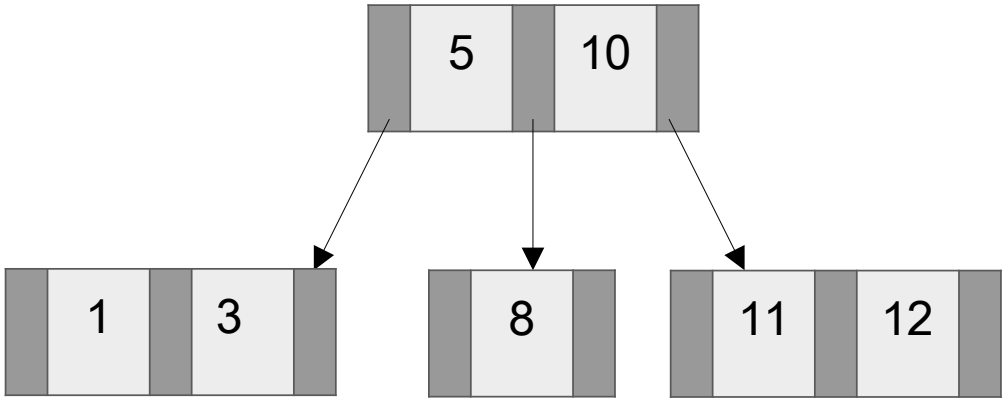


Regra 3: inserção em nó folha duplo com pai duplo
- quebra nó folha e repete operação até encontrar um nó simples
- neste caso a altura da árvore não muda

UDESC

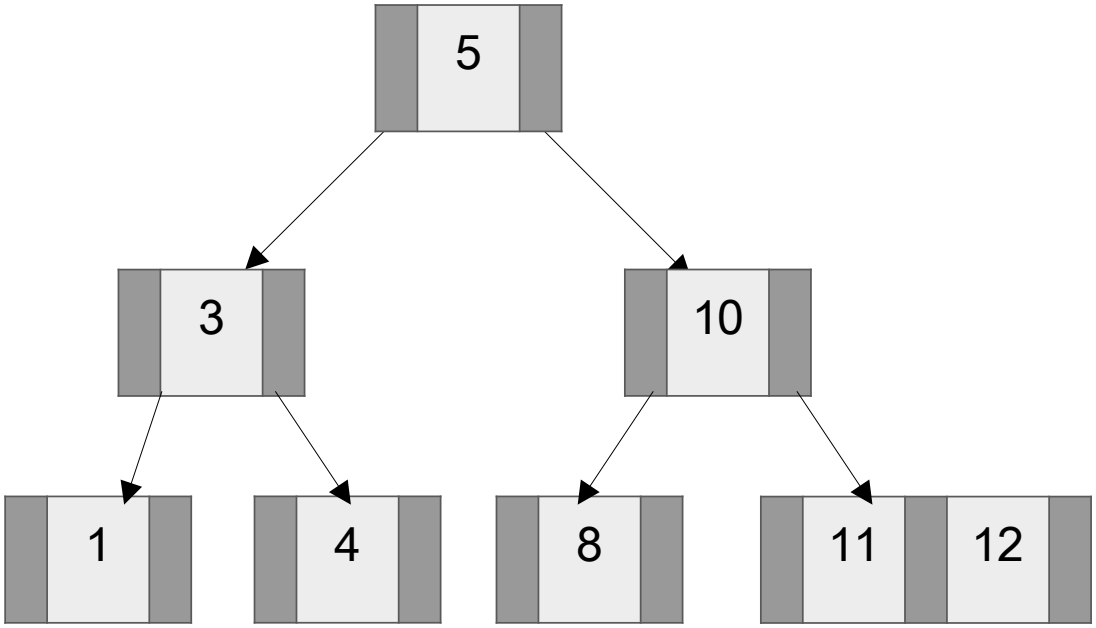
Exemplo – Inserção em nó duplo com pai duplo:

Insere 4



Exemplo – Inserção em nó duplo com pai duplo:

Insere 4



Regra 3: inserção em nó folha duplo com pai duplo
- quebra nó folha e repete operação até encontrar a raiz
  (nenhum nó simples até a raiz)
- neste caso a altura da árvore aumenta um nível

UDESC

Estrutura da árvores 2-3

- Estrutura da árvore

```
typedef struct _no {
  int num_chaves;
  int chave_e, chave_d;
  struct _no *esq, *dir, *centro;
} No;
```

```
// cria um nó com chaves e ponteiros
No *criaNo(int ch1, int ch2, int num_chaves, No *pe, No *pc, No *pd);

// verifica se o nó em questão é folha
int ehFolha(No *no);

// coloca uma nova chave e sub-arvore em nó com apenas uma chave
void adicionaChave(No *no, int ch, No *p);
```

UDESC

Operações em árvores 2-3

- Localizar um nó a partir de uma chave

```
No*  localizaNo(No* no, int chave) {
  if (no==NULL)
    return NULL;       // não encontrou
  if (chave == no->chave_e)
    return no;         // é a chave esquerda
  if ((no->num_chaves == 2) && (chave == no->chave_d))
    return no;         // é a chave direita
  if (chave < no->chave_e)
    return localizaNo(no->esq, chave);
  else if ((no->num_chaves == 1) || (chave < no->chave_d))
    return localizaNo(no->centro, chave);
  else
    return localizaNo(no->dir, chave);
}
```

Operações em árvores 2-3

- Quebra nó (split) ao inserir em chave dupla

```
No *quebraNo(No *no, int valor, int *rval, No *subarvore){
    No *aux;
    if (valor > no->chave_r) {  // valor esta mais a direita
       *rval = no->chave_d;     // promove a antiga maior
       aux = no->dir;
       no->dir = NULL;          // elimina o terceiro filho
       no->num_chaves = 1;      // atualiza o número de chaves
       return criaNo(valor, 0, 1, aux, subarvore, NULL);
    } else if (valor >= no->chave_l){ // valor esta no meio
       *rval = valor;           // continua sendo promovido
       aux = no->dir;
       no->dir = NULL;
       no->num_chaves = 1;
       return criaNo(no->rkey, 0, 1, subarvore, aux, NULL);
    } else {                    // val esta a mais a esquerda
       *rval = no->esq;
       // primeiro cria o nó a direita...
       aux = criaNo(no->chave_r, 0, 1, no->centro, no->dir, NULL);
       no->esq = valor; // ...em seguida arruma o nó a esquerda
       no->num_chaves = 1;
       no->dir = NULL;
       no->centro = subarvore;
       return aux;
    }
}
```

Operações em árvores 2-3

- Insere novo valor na árvore (se necessário retorna novo nó e valor rval)

```
No *insere(No **no, int valor, int *rval){
  No *aux;
  int vaux, promov;

  if (*no == NULL) {     // arvore vazia
    *no = criaNo(valor, 0, 1, NULL, NULL, NULL);  // cria no folha com o novo valor
    return NULL;         // nada a fazer depois
  }
  if (ehFolha(*no)){     // chegou a folha
    if ((*no)->num_chaves == 1){ // caso fácil
      adicionaChave(*no, valor, NULL);
      return NULL;
    } else {
      paux = quebraNo(*no, valor, &vaux, NULL);
      *rval = vaux;
      return aux;
    }
  } else {               // continua a procura
    ...
}
```

Operações em árvores 2-3

- Insere novo valor na árvore (cont.)

```
No *insere(No **no, int valor, int *rval){
  No *aux;
  int vaux, promov;
  if (*no == NULL) { ... // arvore vazia
  if (ehFolha(*no)){ ... // chegou a folha
  } else {               // continua a procura
    if (valor < (*no)->chave_e)
       aux = insere( &((*no)->esq), valor, &vaux);
    else if (((*no)->num_chaves == 1) || (valor < (*no)->chave_e))
       aux = insere( &((*no)->centro), valor, &vaux);
    else
       aux = insere( &((*no)->dir), valor, &vaux);
    if (aux == NULL) // nao promoveu
       return NULL;
    else if ((*no)->num_chaves == 1){
      adicionaChave(*no, vaux, aux);
      return NULL;
    } else {
      No *aux2 = quebraNo(*no, vaux, &promov, aux);
      *rval = promov;
      return aux2;
    }
  }
}
```

UDESC

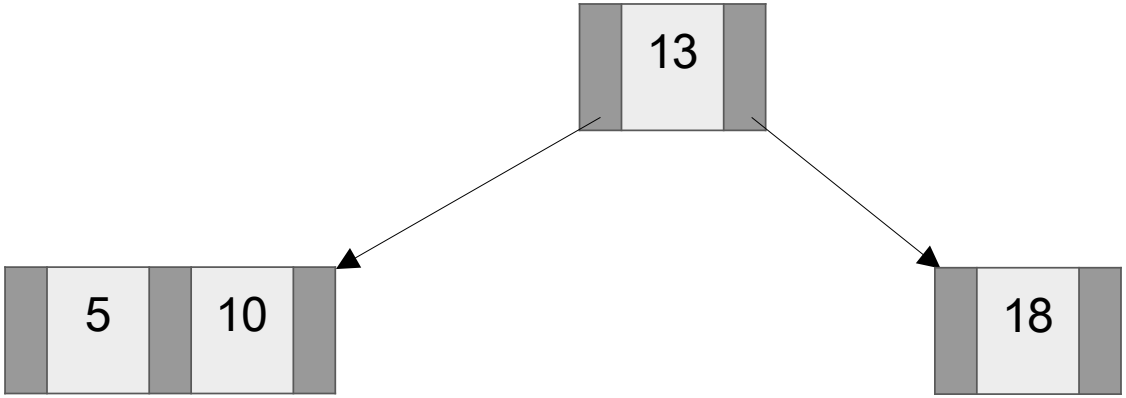
Remoção em árvores 2-3

- Passo 1: Localizar a chave a ser removida;
- Passo 2: Verificar se o nó é folha. Caso não for folha, trocar por seu sucessor. A remoção somente ocorre nas folhas:
  - Se o nó folha tiver duas chaves, basta remover a chave desejada;
  - Caso o nó tiver apenas a chave a ser removida, é preciso reequilibrar os nós.
- Passo 3: Reequilíbrio. Após a remoção, podem ser necessárias operações para reequilibrar a árvore:
  - Fusão: Se um nó com uma chave for removido e o seu irmão for também um nó com uma chave, os nós podem ser fundidos em um nó com duas chaves (nó 3).
  - Rotação: Se um nó com uma chave for removido e o seu irmão for um nó com duas chaves, pode ser realizada uma rotação para redistribuir as chaves e manter o equilíbrio.

UDESC

Exemplo – Remoção em nó duplo folha:

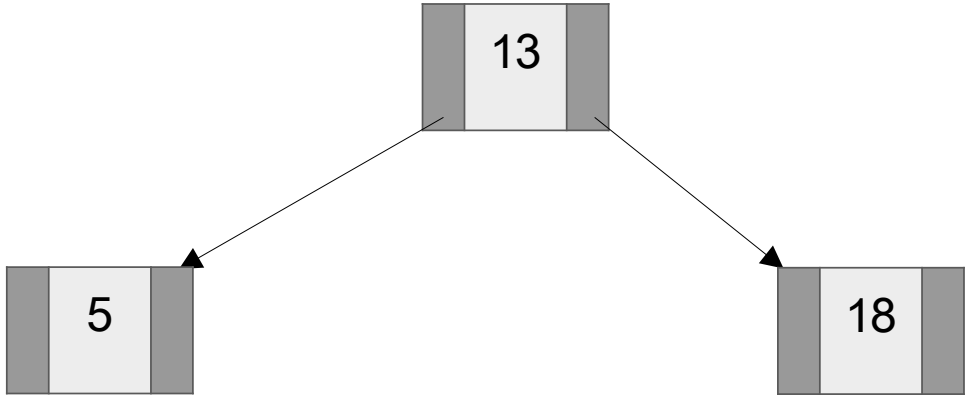
Remove 10



UDESC

Exemplo – Remoção em nó duplo folha:

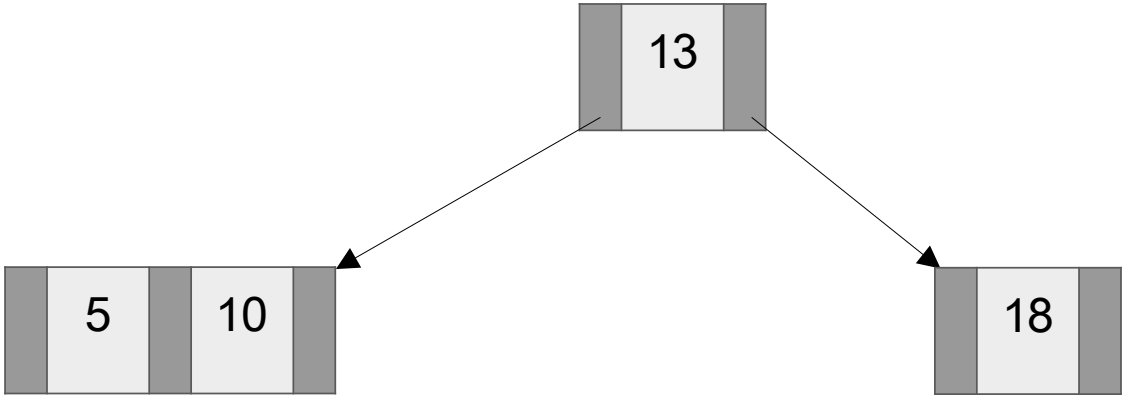
Remove 10



UDESC

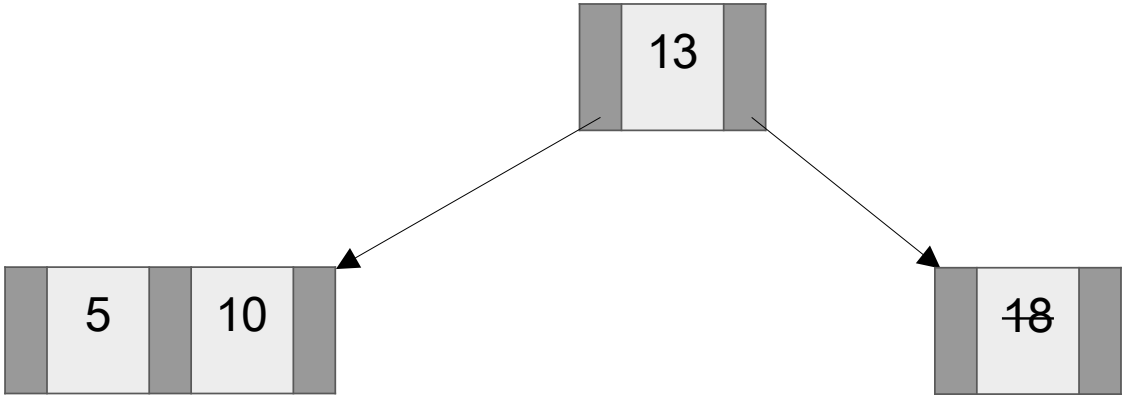
Exemplo – Remoção em nó simples folha:

Remove 18



Exemplo – Remoção em nó simples folha:

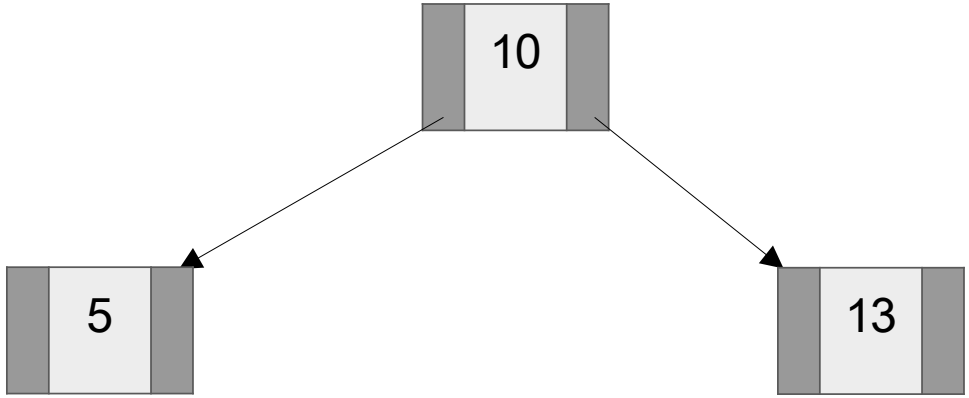
Remove 18



Rotação: Se um nó com uma chave for removido e o seu irmão for um nó com duas chaves, pode ser realizada uma rotação para redistribuir as chaves e manter o equilíbrio.

Exemplo – Remoção em nó simples folha:

Remove 18

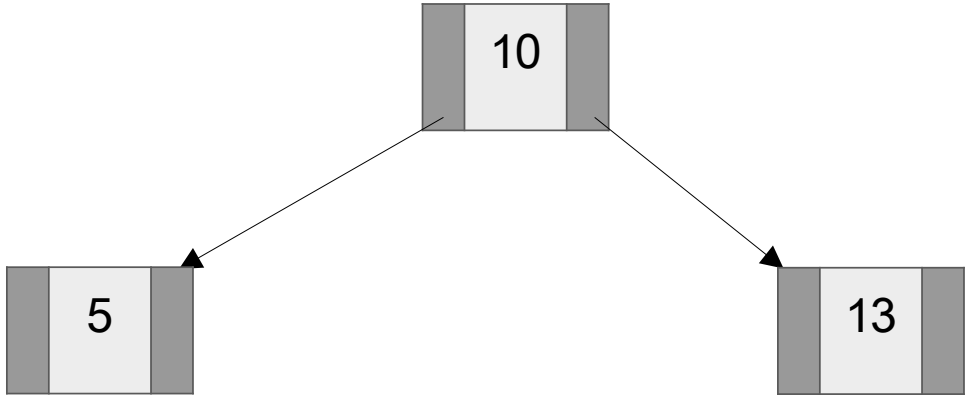


Rotação: Se um nó com uma chave for removido e o seu irmão for um nó com duas chaves, pode ser realizada uma rotação para redistribuir as chaves e manter o equilíbrio.

UDESC

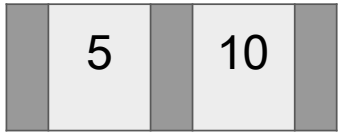
Exemplo – Remoção em nó simples folha:

Remove 13



Exemplo – Remoção em nó simples folha:

Remove 13

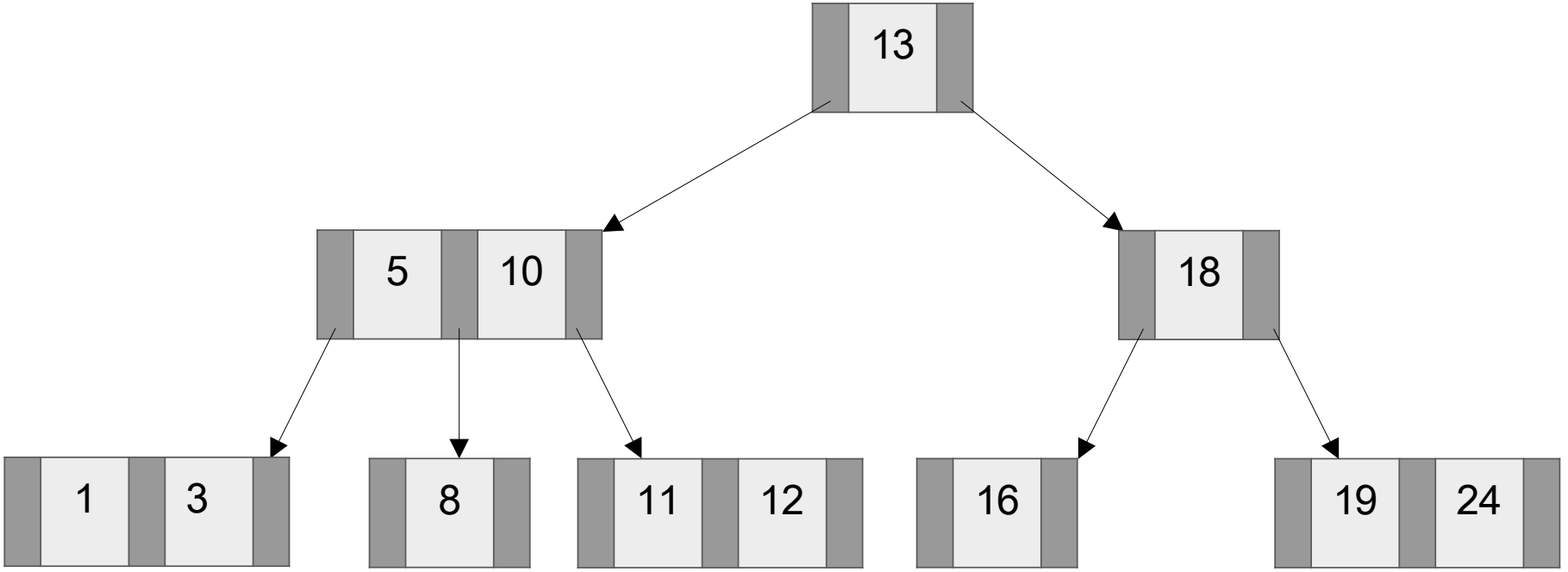


Fusão: Se um nó com uma chave for removido e o seu irmão for também um nó com uma chave, os nós podem ser fundidos em um nó com duas chaves (nó 3).

UDESC

Exemplo – Remoção em nó duplo:

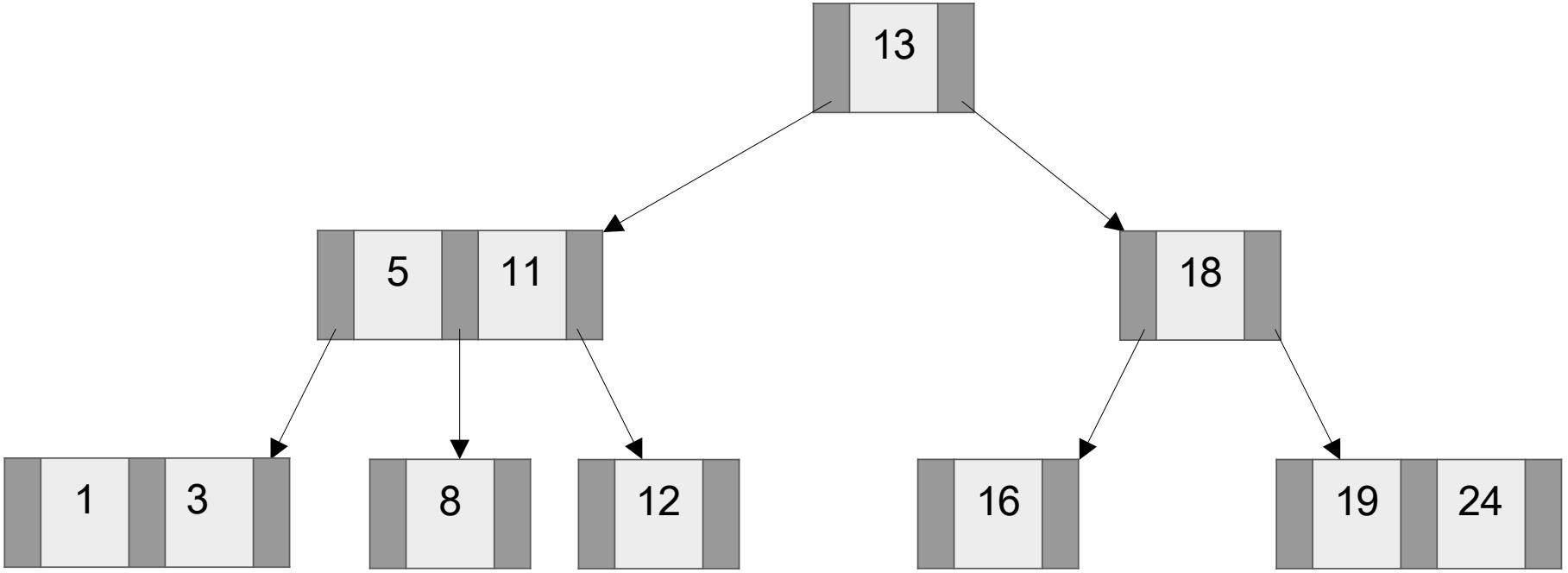
Remove 10



UDESC

Exemplo – Remoção em nó duplo:

Remove 10

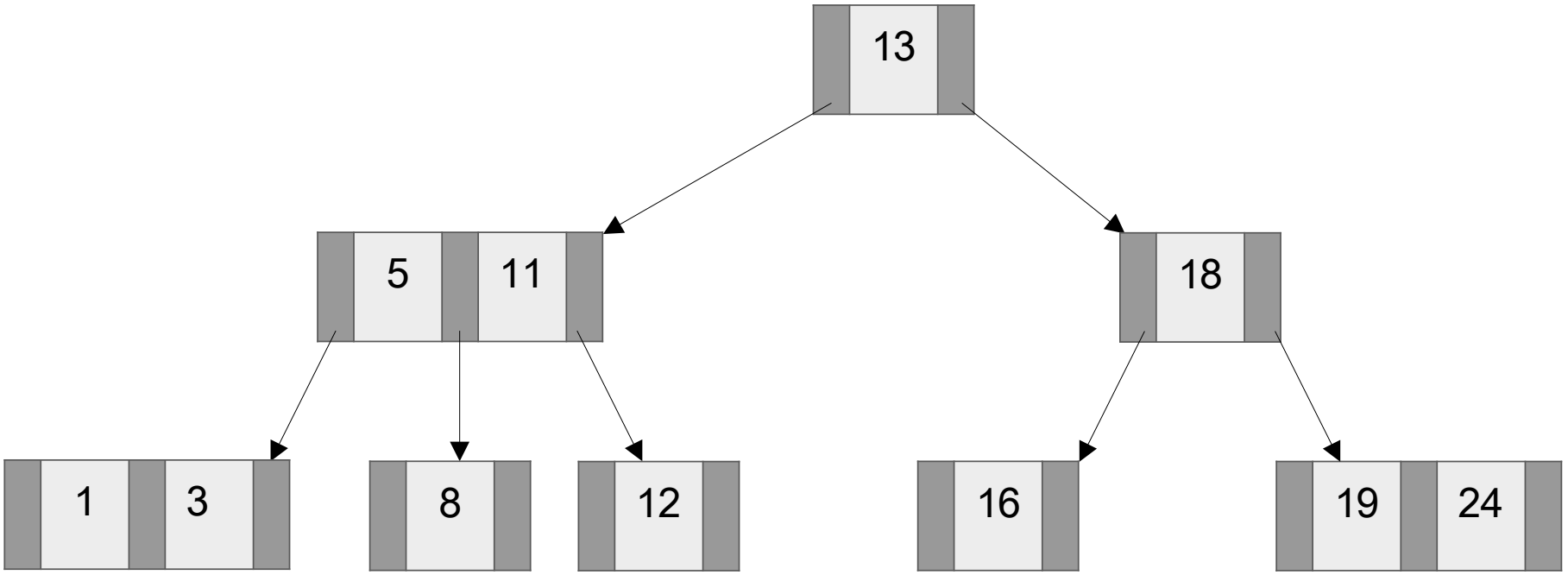


Caso não for folha, trocar por seu sucessor. A remoção somente ocorre nas folhas:
- Se o nó folha tiver duas chaves, basta remover a chave desejada

UDESC

Exemplo – Remoção em nó simples:

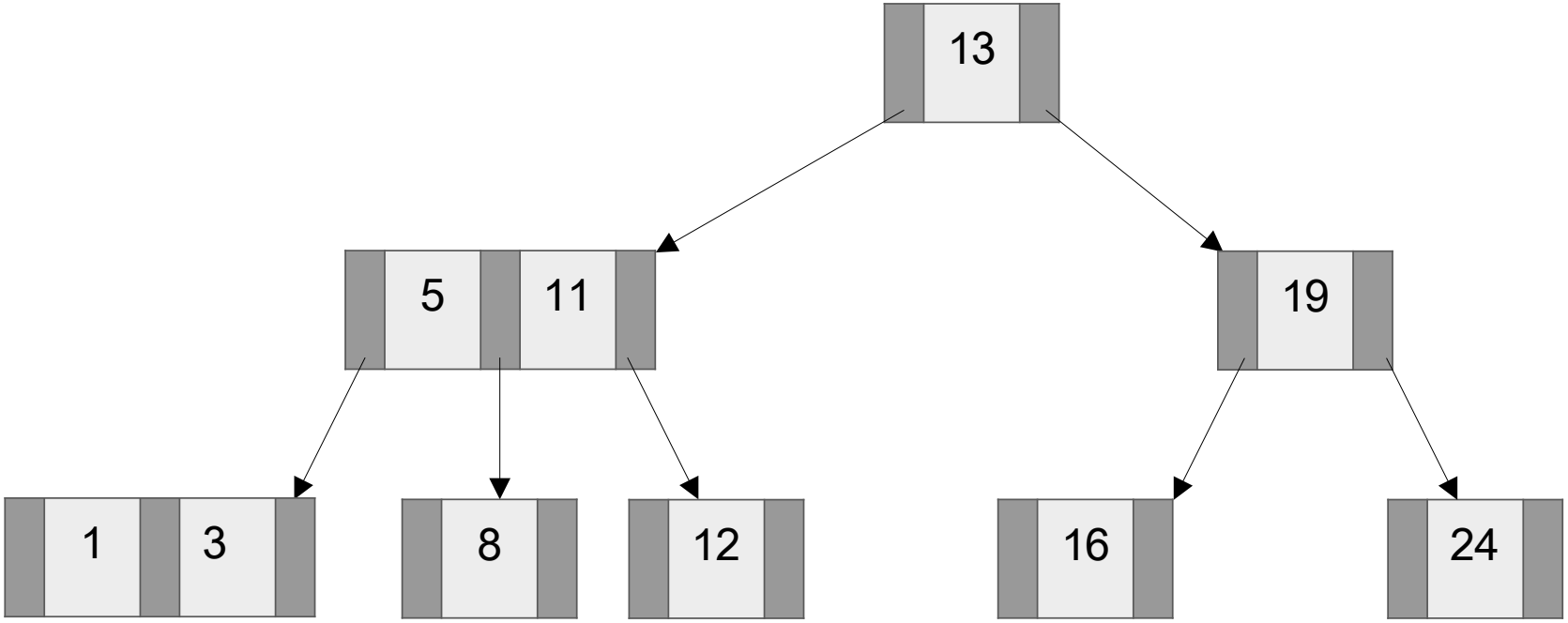
Remove 18



UDESC

Exemplo – Remoção em nó simples:

Remove 18

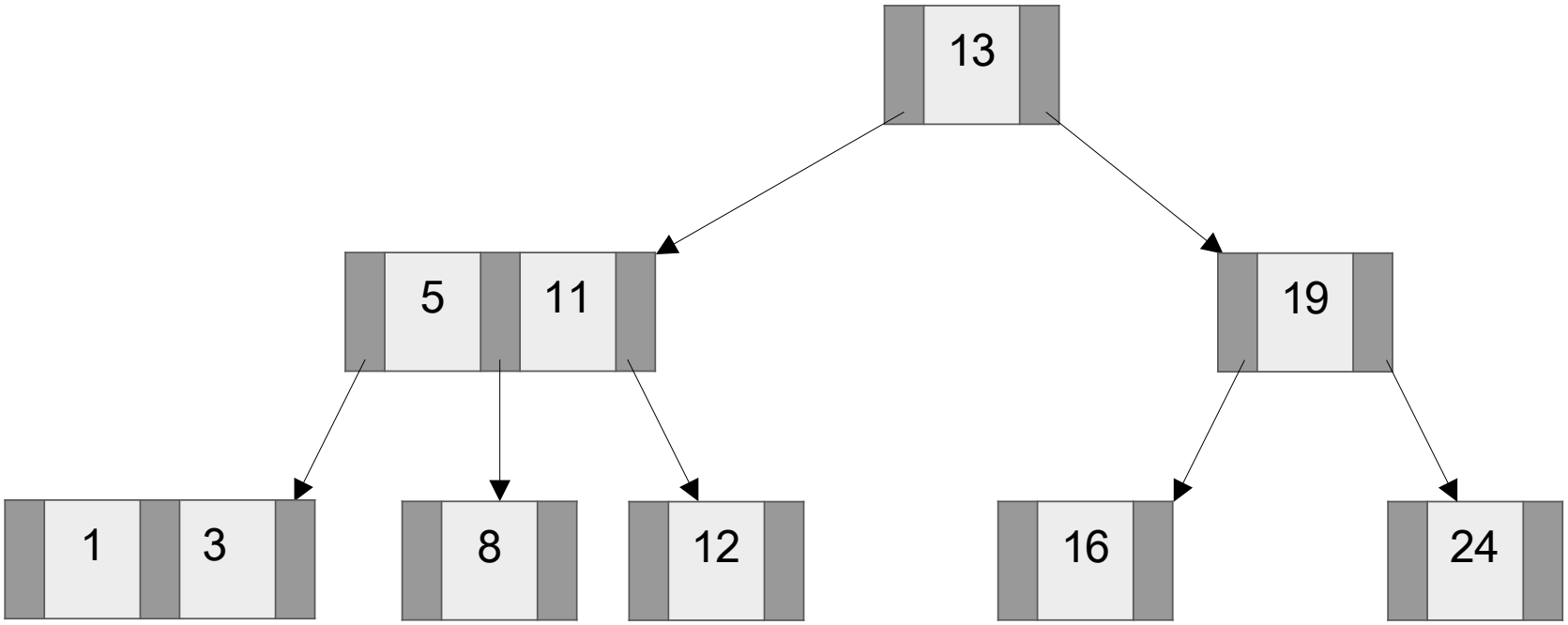


Caso não for folha, trocar por seu sucessor. A remoção somente ocorre nas folhas:
- Se o nó folha tiver duas chaves, basta remover a chave desejada

UDESC

Exemplo – Remoção em nó simples:

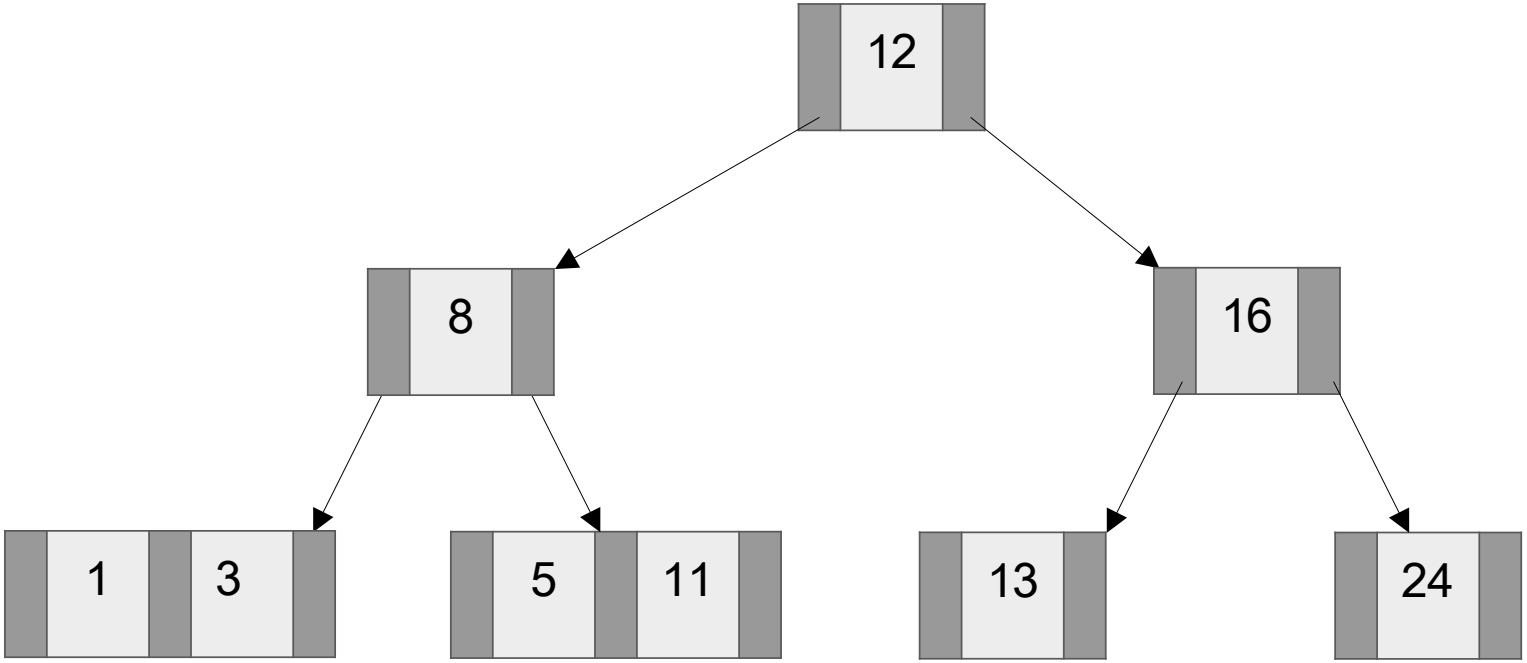
Remove 19



UDESC

Exemplo – Remoção em nó simples:

Remove 19



Após sucessivas rotações e fusões

UDESC

Exercícios

1. Quantas chaves, no máximo, pode conter uma árvore 2-3 de altura 2? Qual o número mínimo de chaves em uma árvore 2-3 de altura 2? Desenhe exemplos dessas árvores.

2. Desenhe a árvore 2-3 que resulta da inserção das chaves [10, 1, 20, 30, 18, 25, 24, 11, 3], nesta ordem, em uma árvore inicialmente vazia.

3. Os dados numa árvore 2-3 são mantidos ordenados? Justifique.

4. Existe alguma relação entre a árvore Rubro-Negra e a árvore 2-3? Seria possível representar qualquer árvore Rubro-Negra em uma árvore 2-3?

5. Pesquise sobre árvores 2-3-4 (também chamada de 2-4). Verifique qual a relação entre ela e a árvore Rubro-Negra. Qual vantagem e desvantagem em relação à arvore 2-3?

Estruturas de dados II

Árvore 2-3

André Tavares da Silva

andre.silva@udesc.br