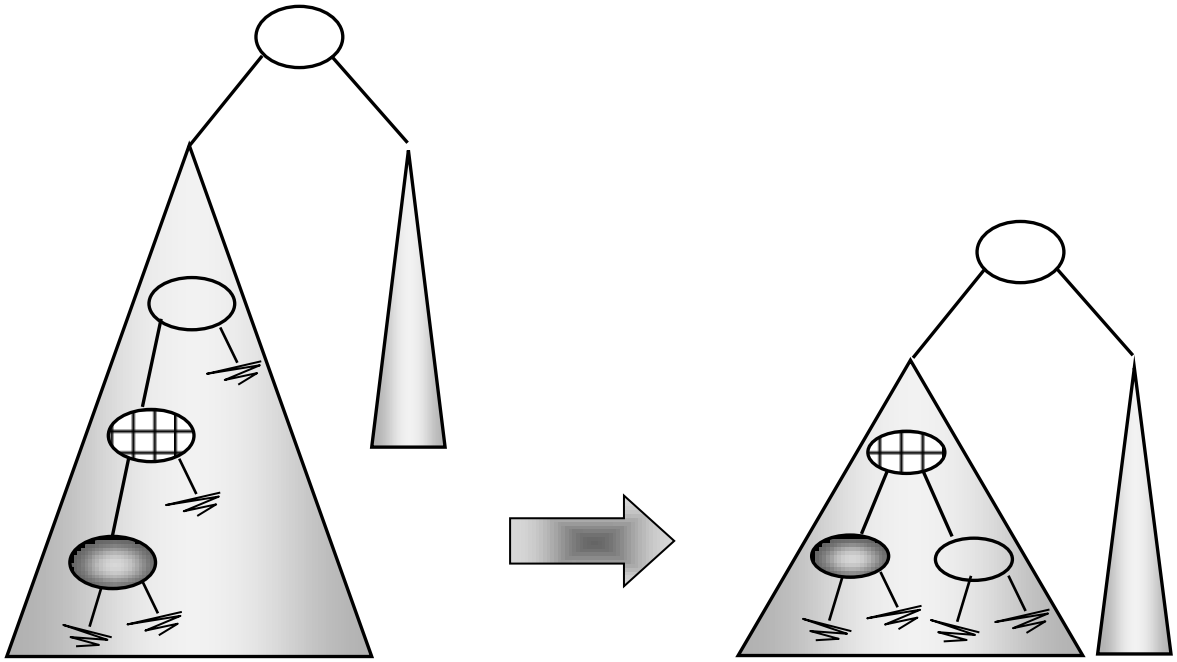
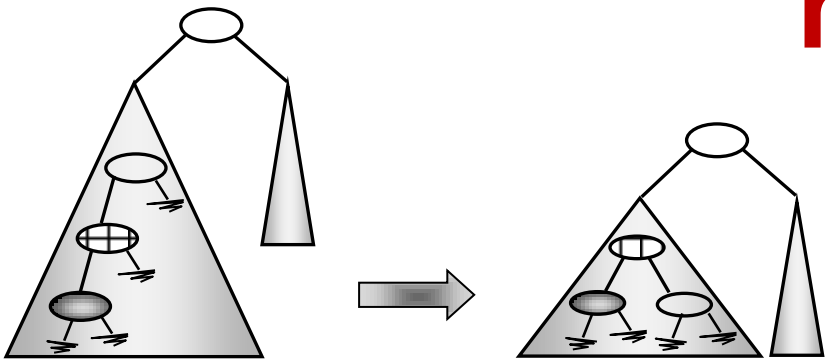
Estruturas de Dados com Jogos



Capítulo 9

Árvores Balanceadas

Seus Objetivos neste Capítulo



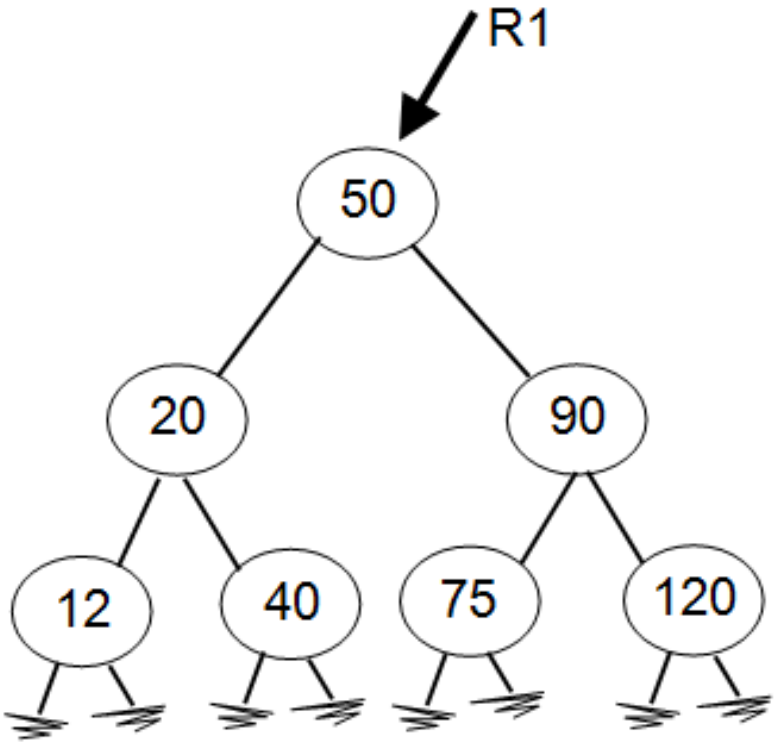
- Entender o conceito de Balanceamento, e sua importância para a eficiência das Árvores Binárias de Busca;
- Desenvolver habilidade para elaborar algoritmos sobre Árvores Binárias de Busca Balanceadas, e para adaptar a lógica do Balanceamento a novas situações, se necessário.

Exercício 9.1 Inserir Sequencia de Valores

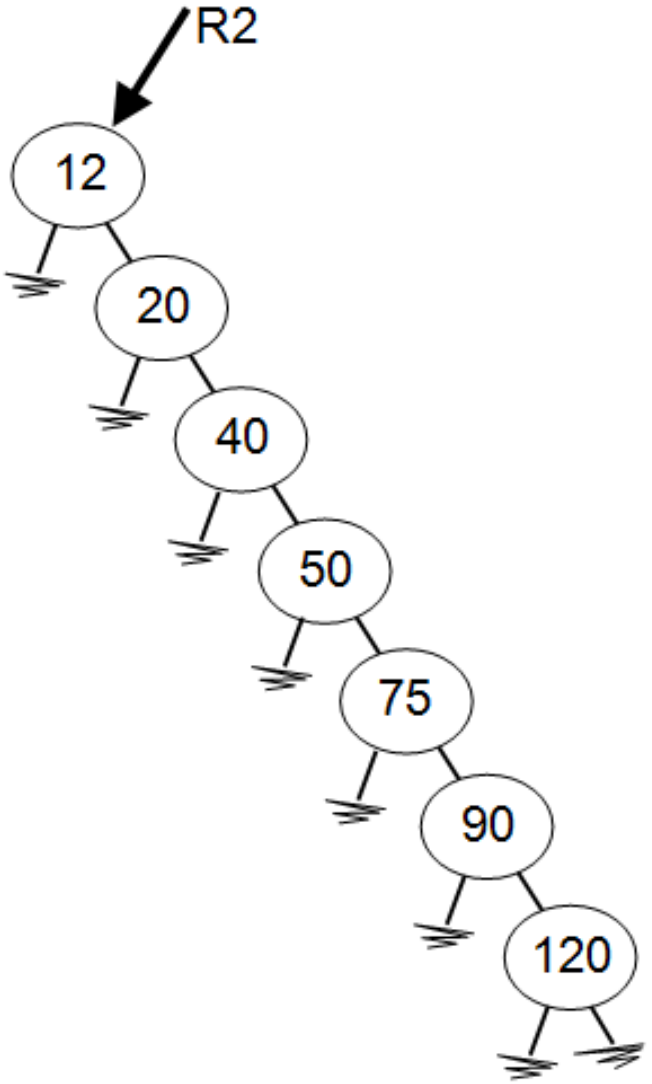
Sequencia (A): 50, 20, 40, 12, 90, 75, 120

Sequencia (B): 12, 20, 40, 50, 75, 90, 120

Exercício 9.1 Inserir Sequencia de Valores



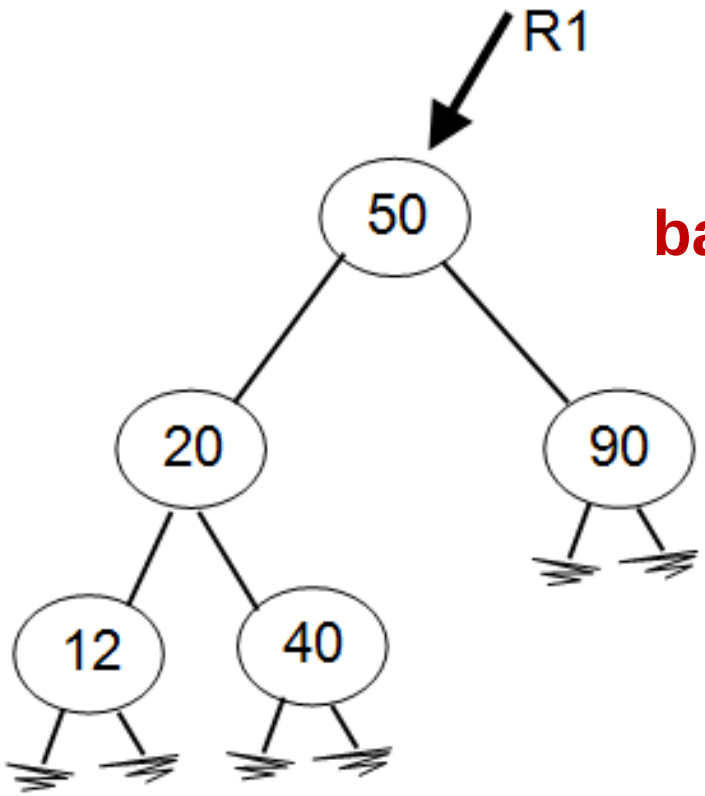
Sequencia (A): 50, 20, 40, 12, 90, 75, 120



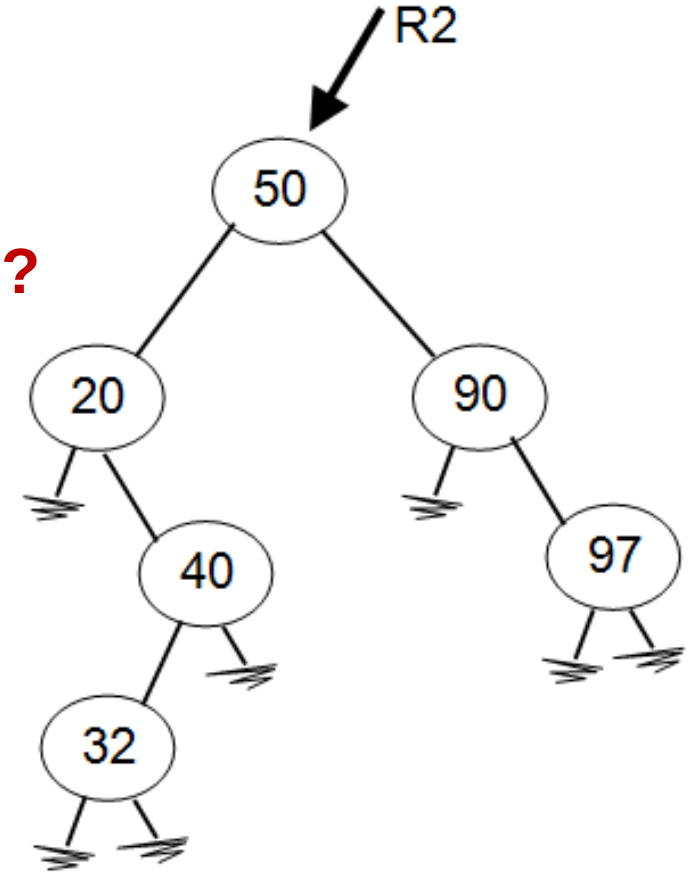
Sequencia (B): 12, 20, 40, 50, 75, 90, 120

Árvore Binária de Busca Balanceada – ABBB:

Para cada Nó da Árvore, as alturas de suas Subárvores diferem de, no máximo, 1.



Estão balanceadas?



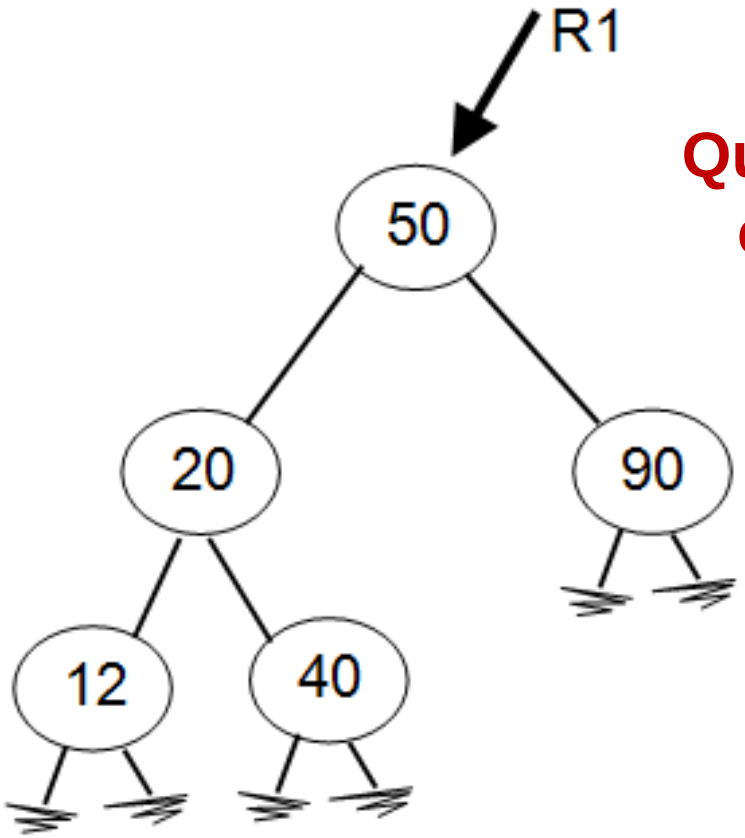
Estratégia para Manter uma ABB Balanceada:

Alterar algoritmos de inserção e eliminação para:

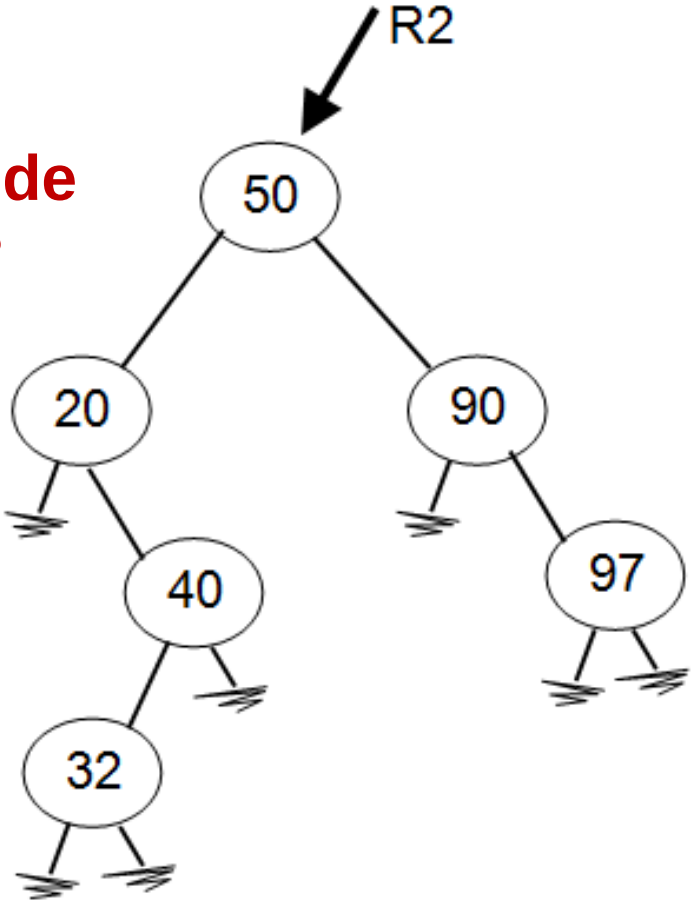
- **Monitorar** o Balanceamento da Árvore; e
- Desencadear ações de **rebalanceamento**, sempre que necessário.

Como Monitorar Objetivamente?
Fator de Balanceamento: Bal = Hd - He

Altura da Subárvore Direita menos a altura da Subárvore Esquerda.

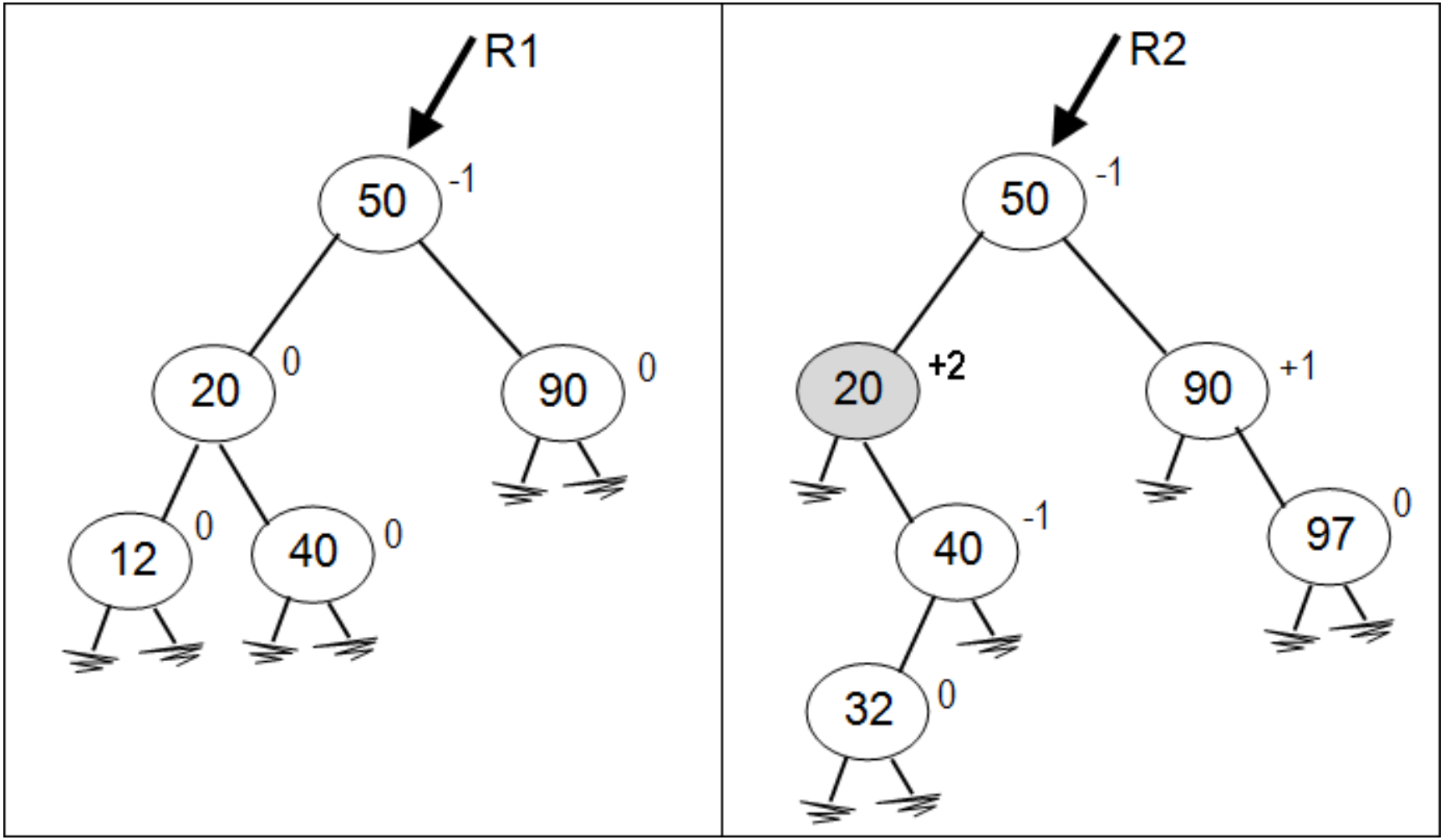


Qual o Bal de cada nó?

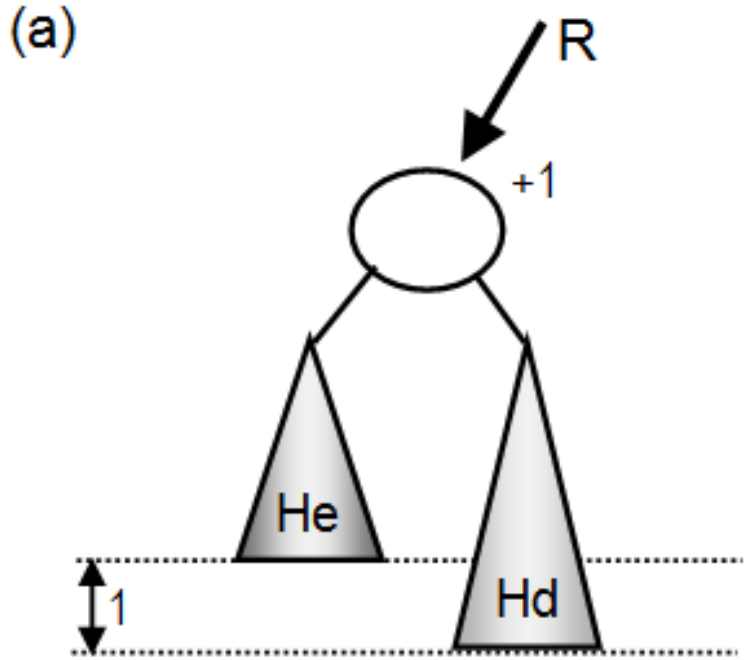
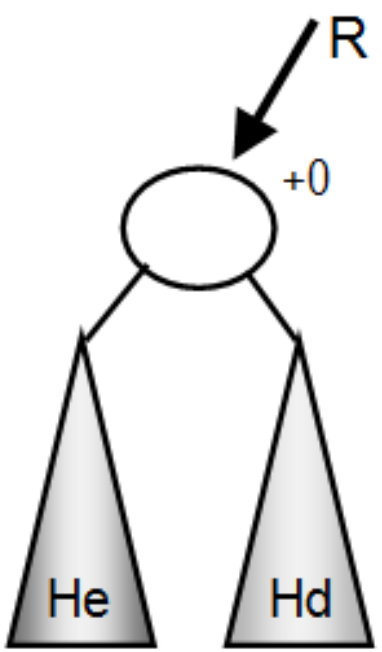


Fator de Balanceamento: Bal = Hd - He

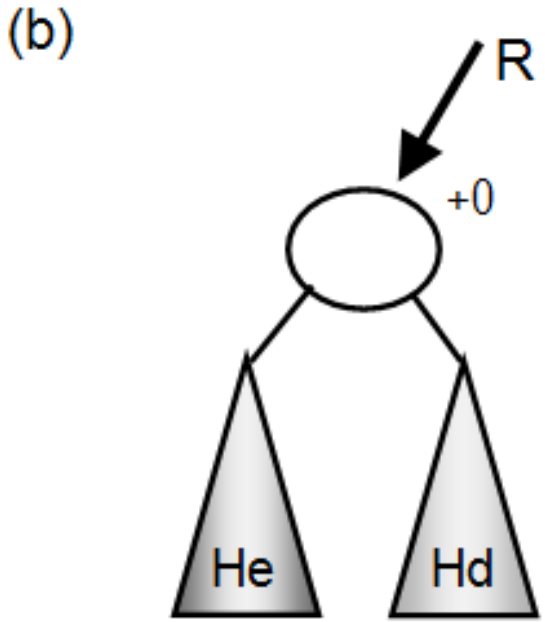
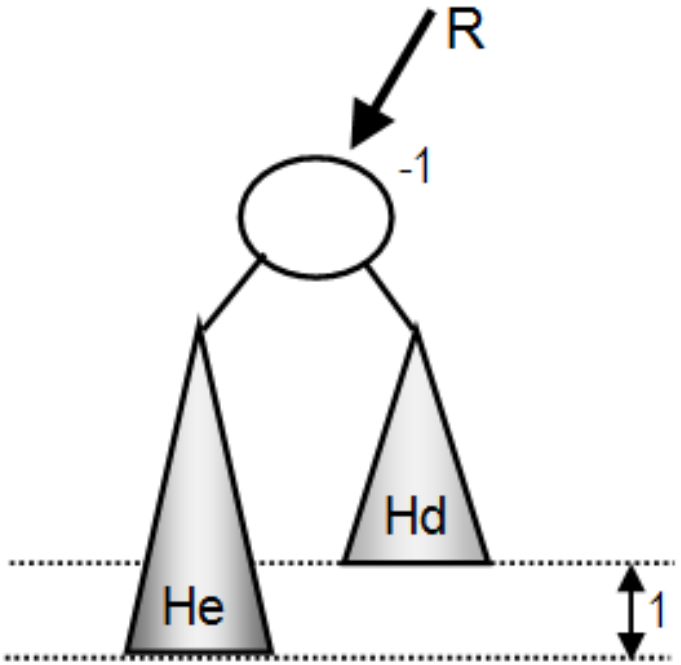
Altura da Subárvore Direita menos a altura da Subárvore Esquerda.



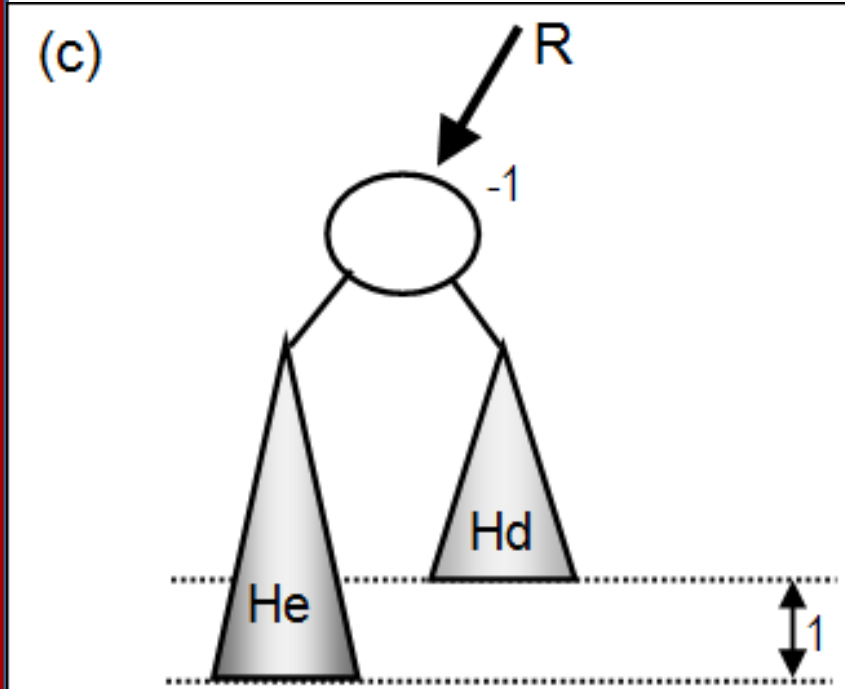
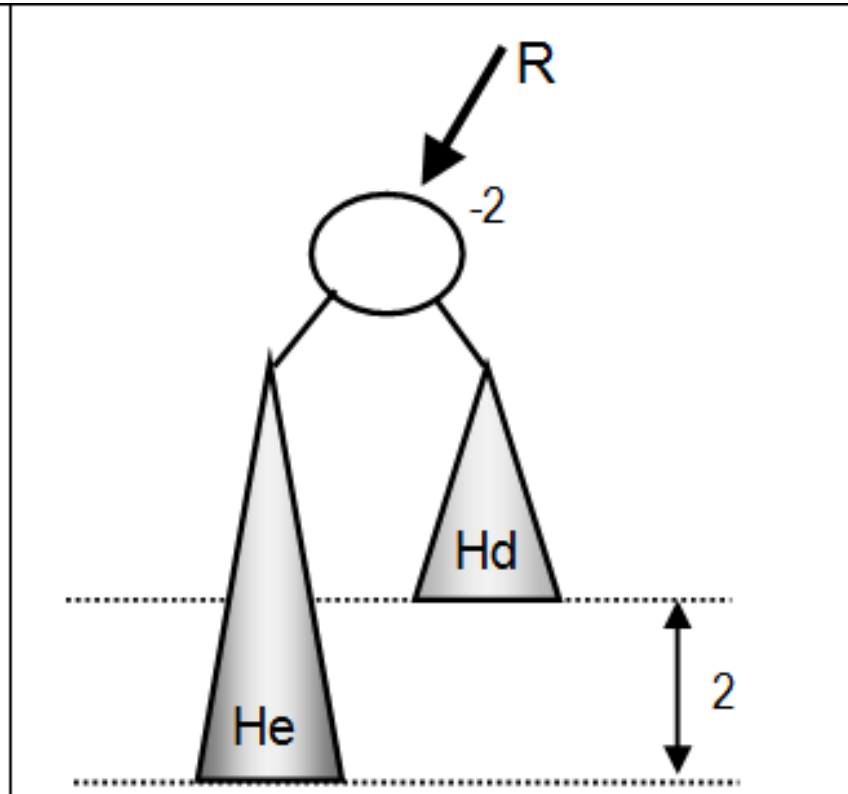
Monitorando o Balanceamento ao Inserir Elementos na Subárvore Esquerda

Antes da Inserção em SE	Após a Inserção em SE
	

Monitorando o Balanceamento ao Inserir Elementos na Subárvore Esquerda

Antes da Inserção em SE	Após a Inserção em SE
	

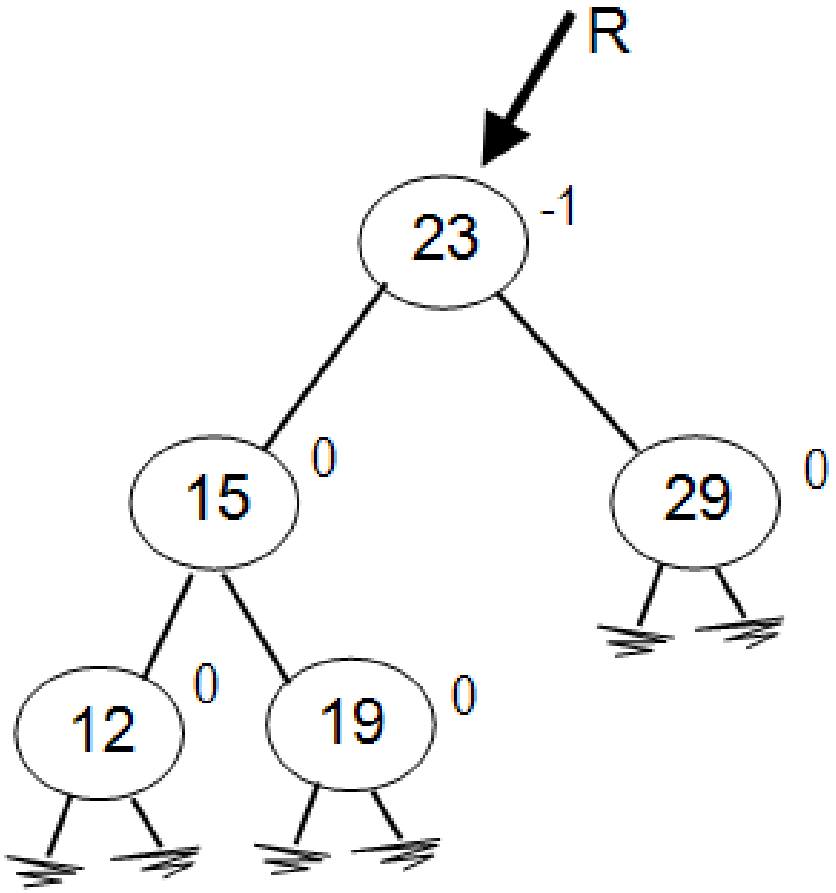
Monitorando o Balanceamento ao Inserir Elementos na Subárvore Esquerda

Antes da Inserção em SE	Após a Inserção em SE
	
	É preciso rebalancear!!!

Monitorando o Balanceamento:

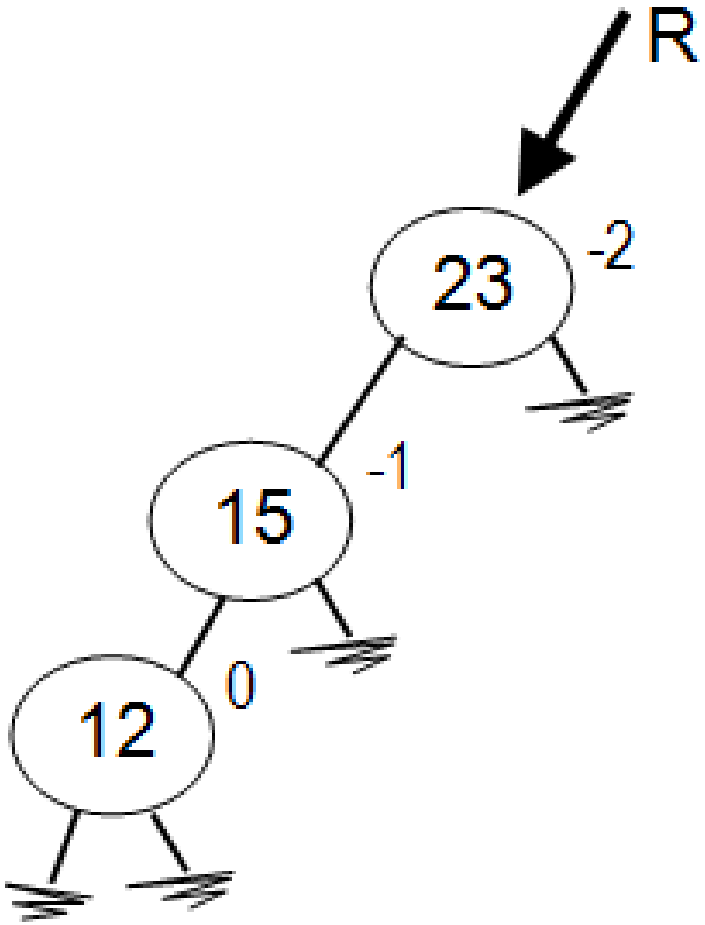
A inserção de...

25
40
9
13
17
21



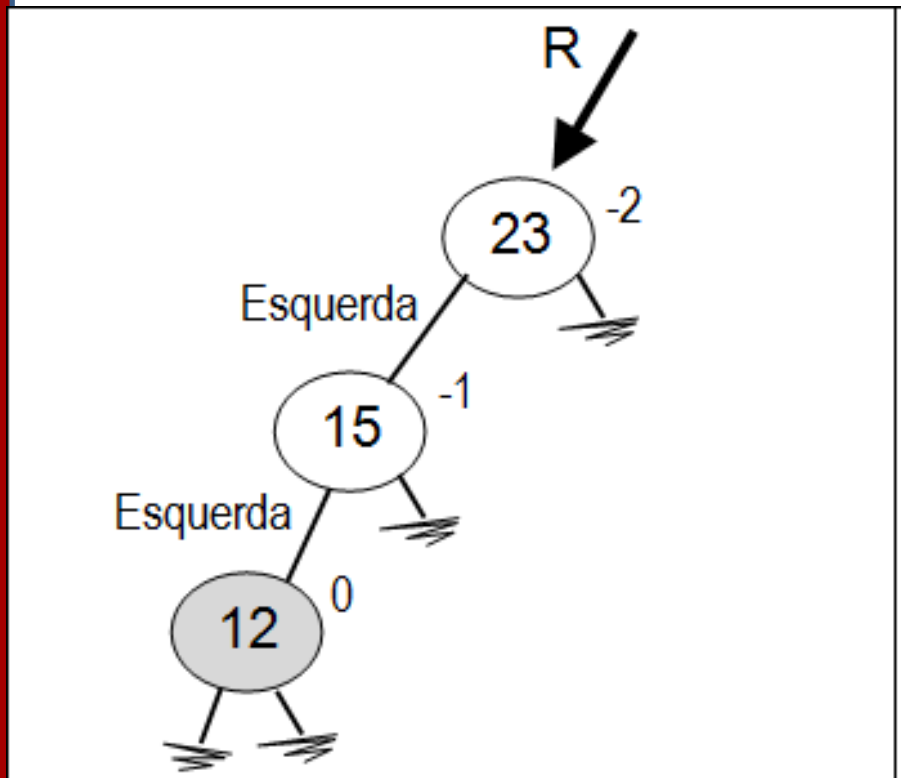
Exercício 9.4 Causaria Desbalanceamento?

Como Rebalancear?

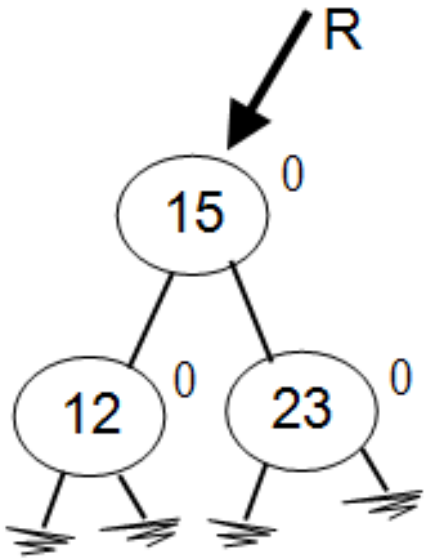


Casos de Rebalanceamento: Caso 1 - Rotação Simples EE

(insere)

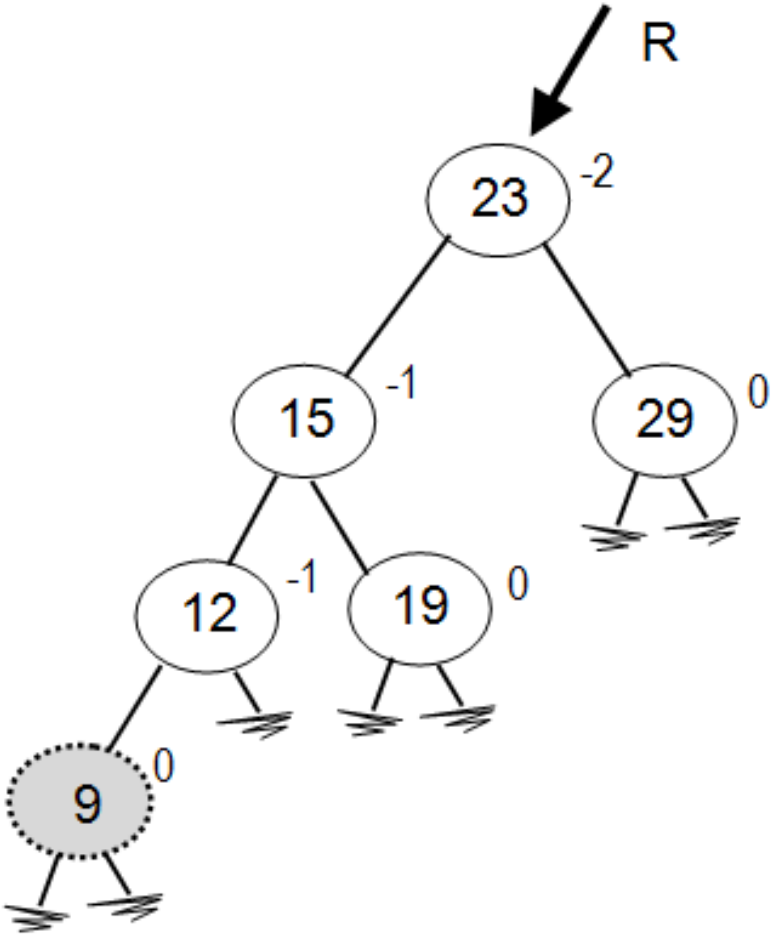


(a) Antes de Balancear



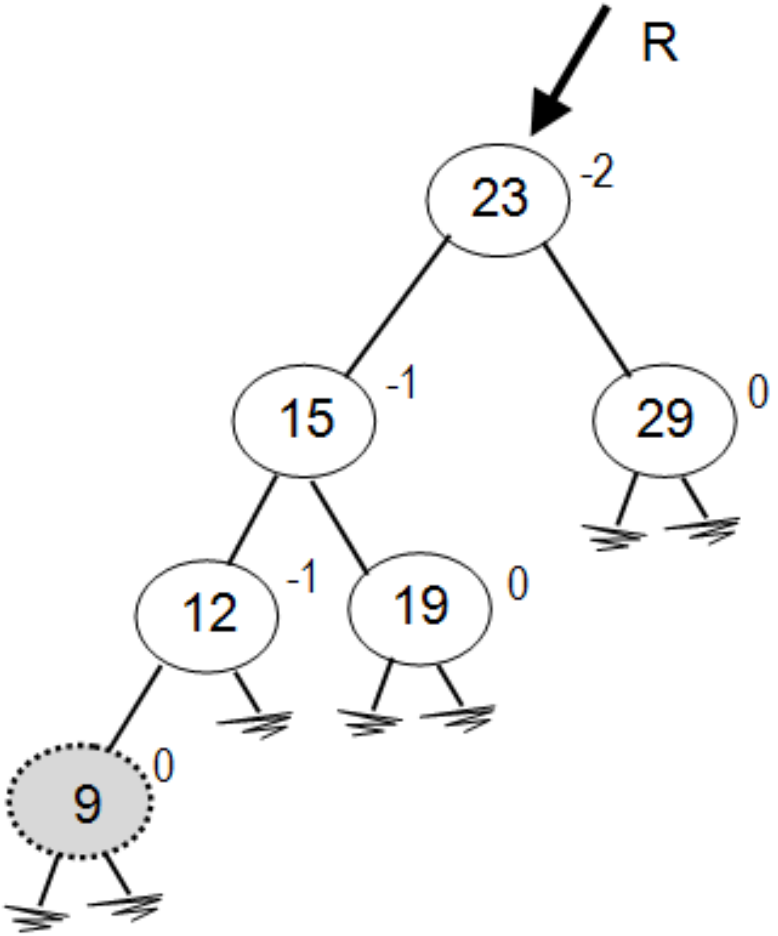
(b) Depois de Balancear

Caso 1 - Rotação Simples EE - Insere

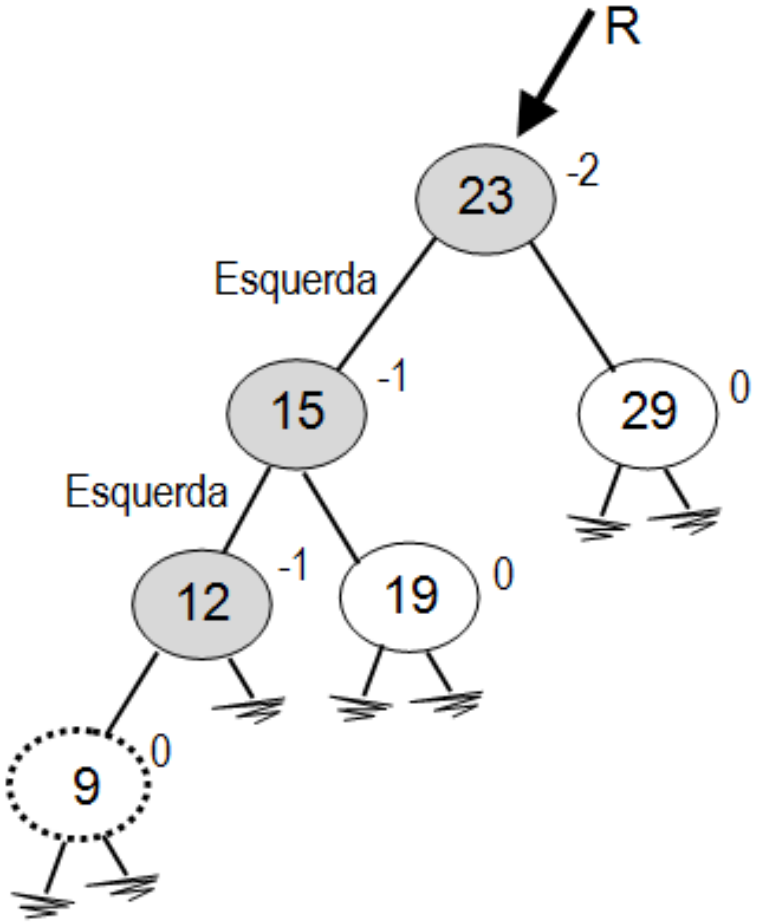


(a) Inseriu e Desbalanceou

Caso 1 - Rotação Simples EE - Insere

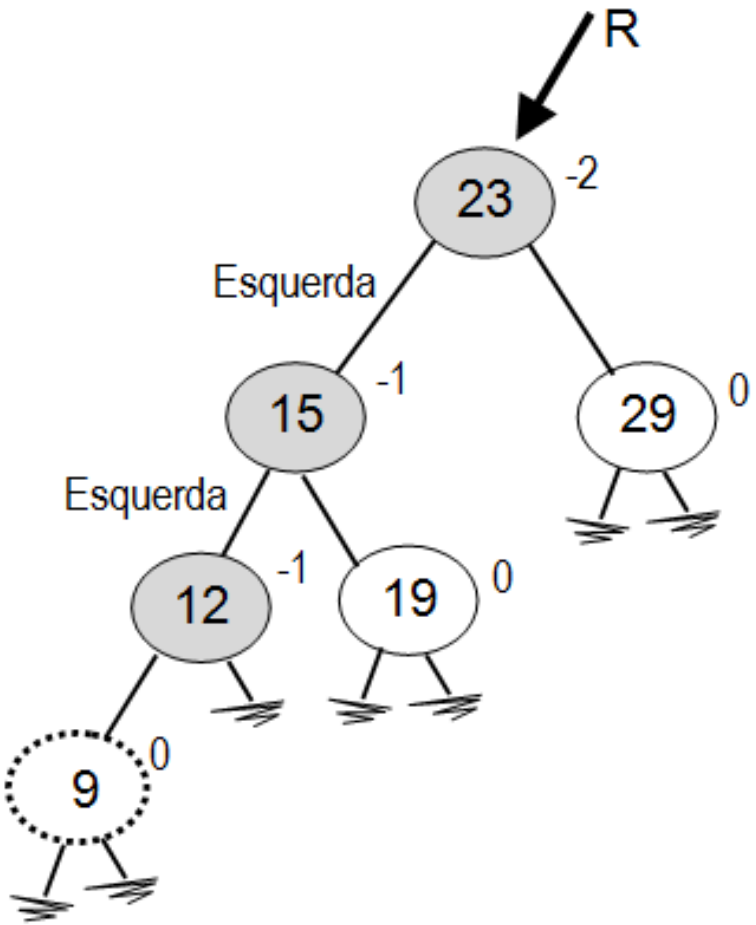


(a) Inseriu e Desbalanceou



(b) Achar os Três Nós Principais

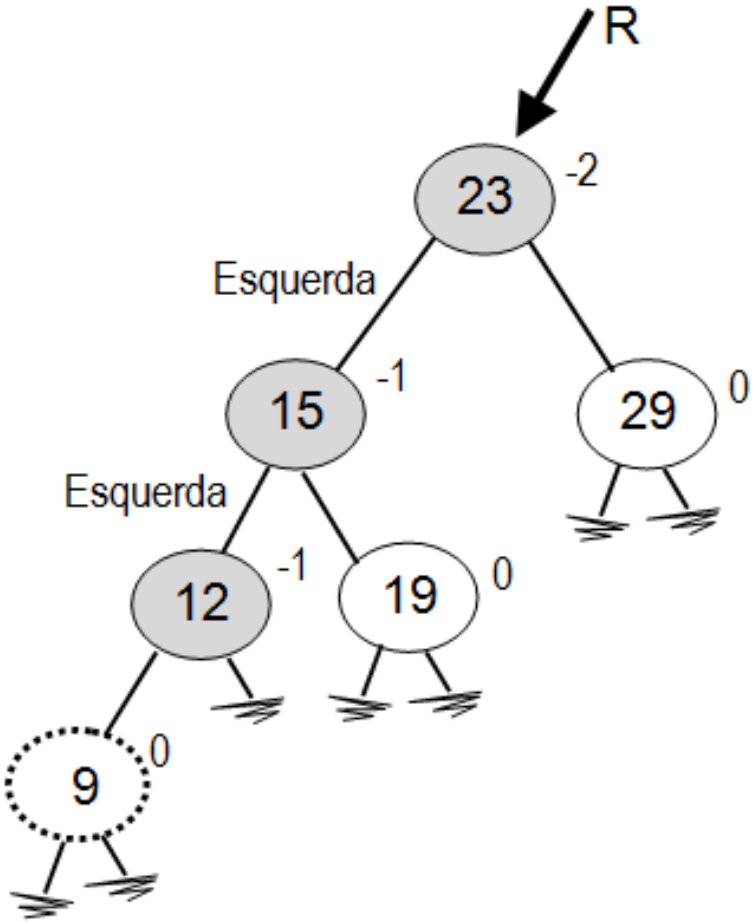
Caso 1 - Rotação Simples EE - Insere



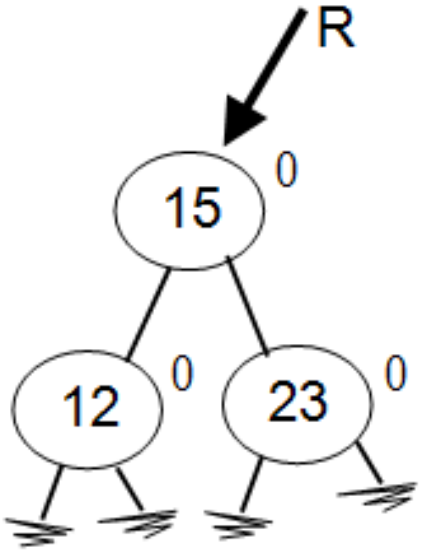
(b) Achar os Três Nós Principais

(c) Reposicionar os Três Nós

Caso 1 - Rotação Simples EE - Insere

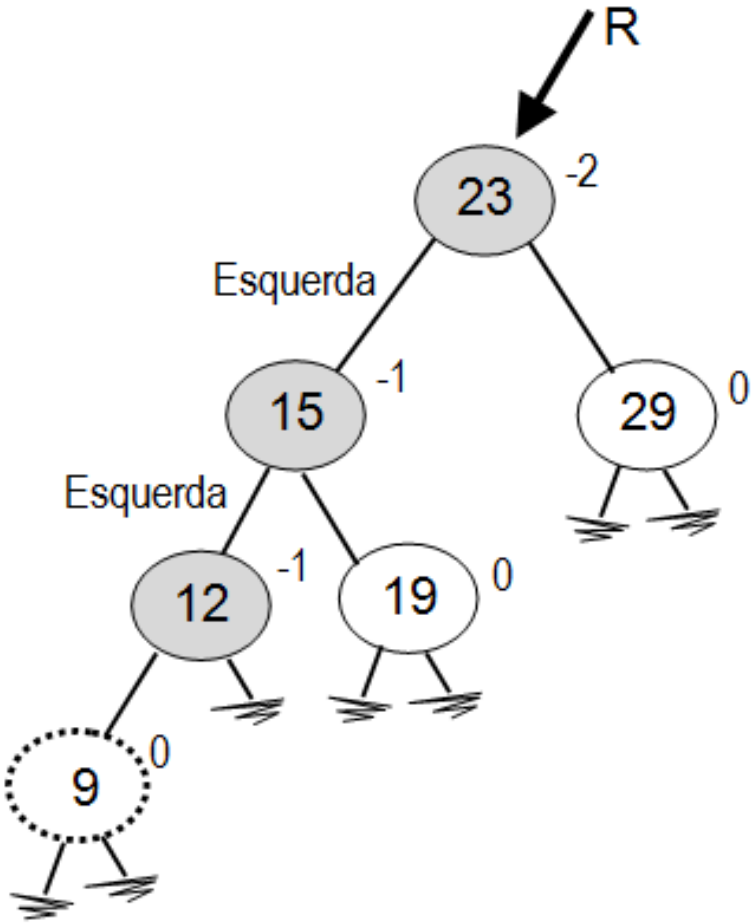


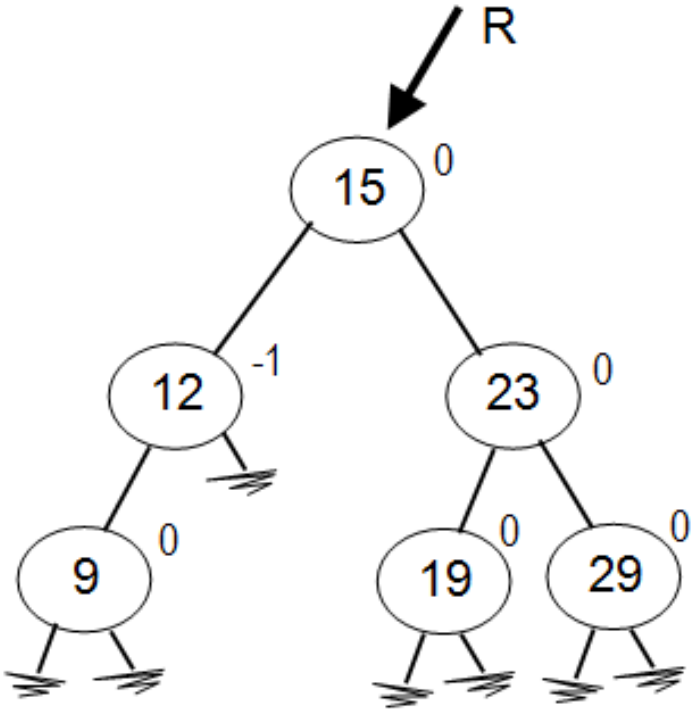
(b) Achar os Três Nós Principais



(c) Reposicionar os Três Nós

Caso 1 - Rotação Simples EE - Insere





(d) Reposicionar os Demais Nós

Para Rebalancear uma Árvore Manualmente

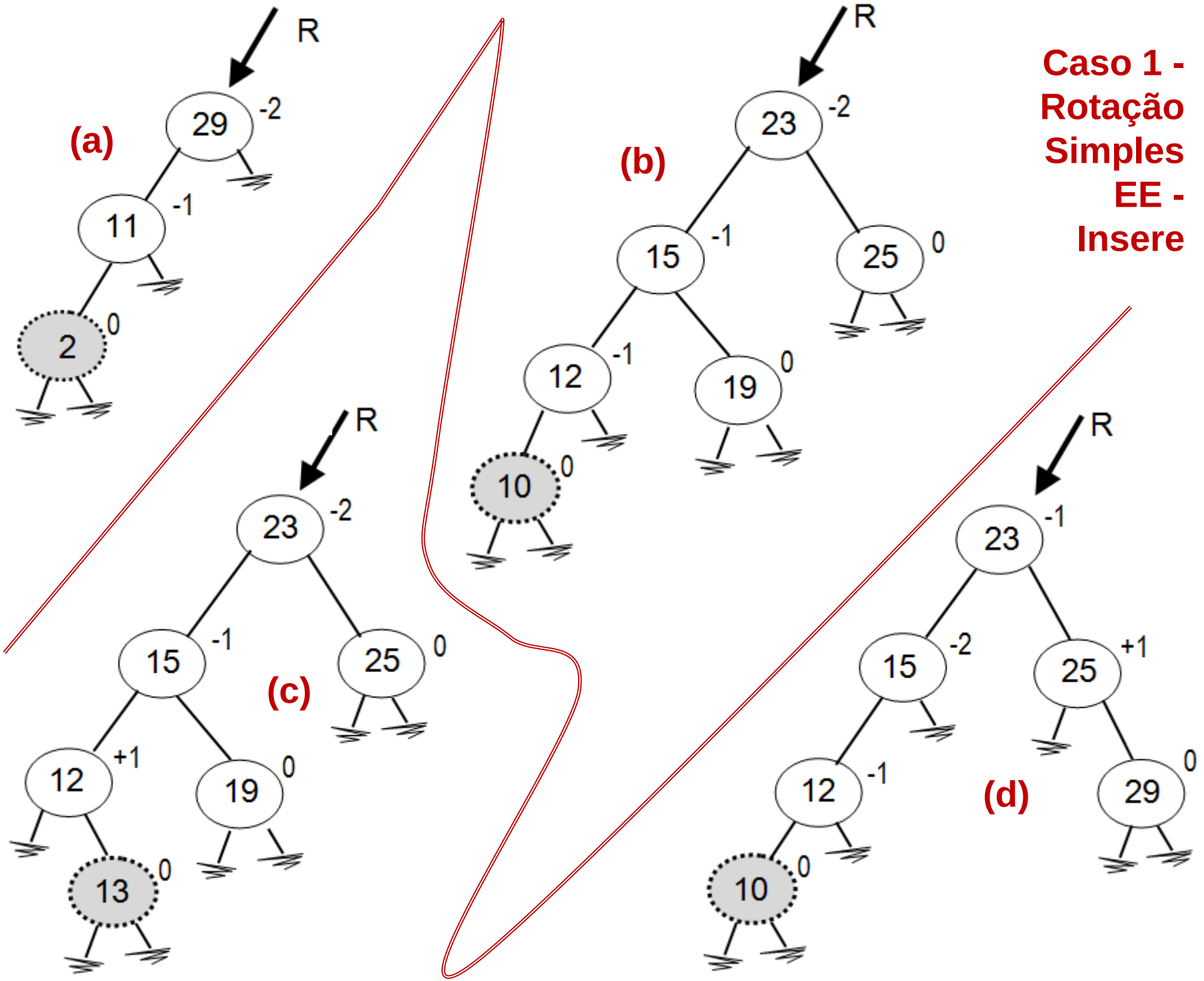
Passo 1: Identificar os Três Nós Principais.

Passo 2: Reposicionar os Três Nós Principais.

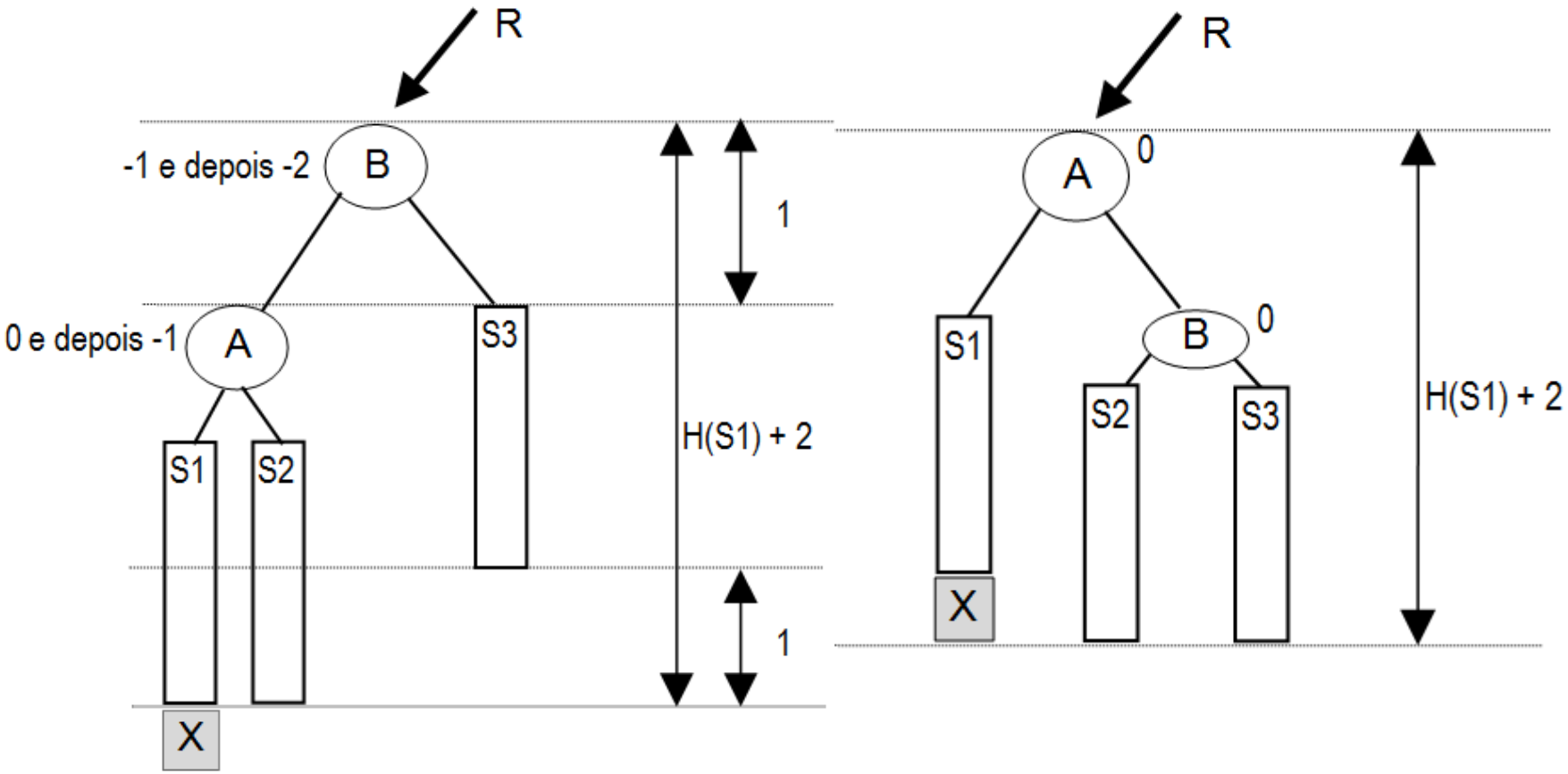
Passo 3: Reposicione os demais valores, respeitando o critério que define uma ABB.

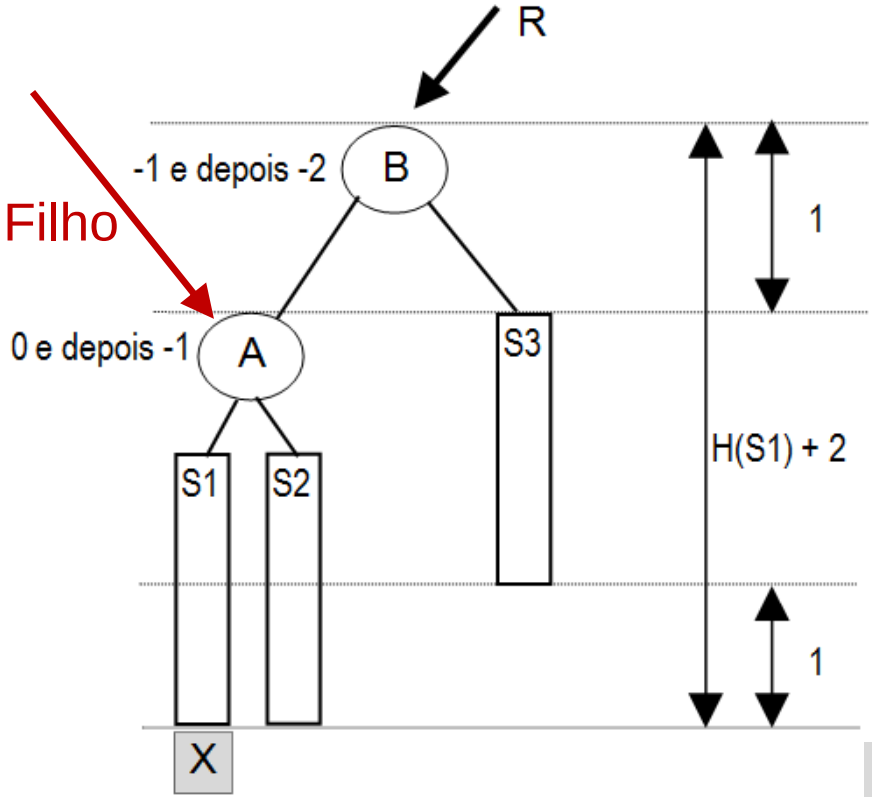
Passo 4: Atualize o Fator de Balanceamento de Cada Nó.

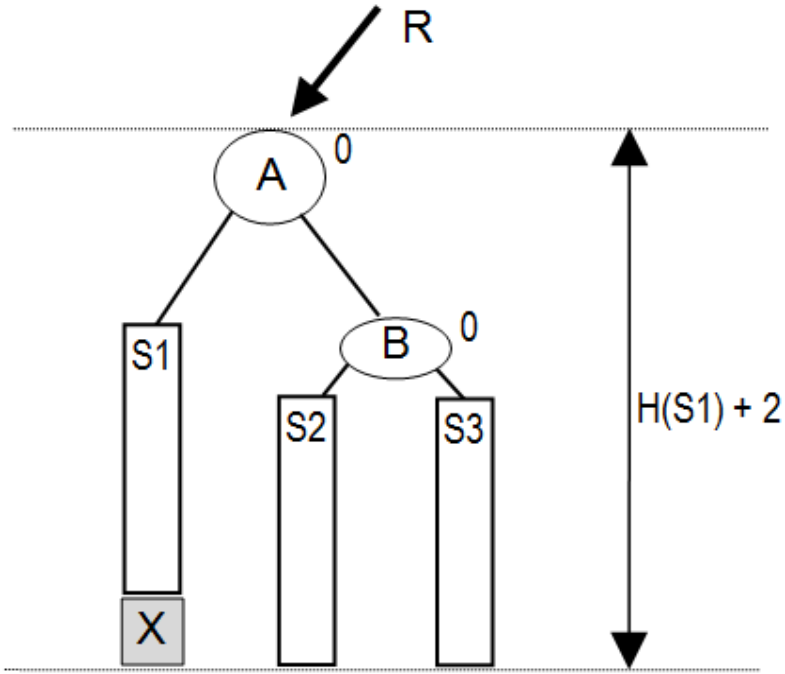
Caso 1 - Rotação Simples EE - Insere



Generalização do Caso 1: Rotação Simples EE - Insere







Algoritmo - Caso 1: Rotação Simples EE - Insere

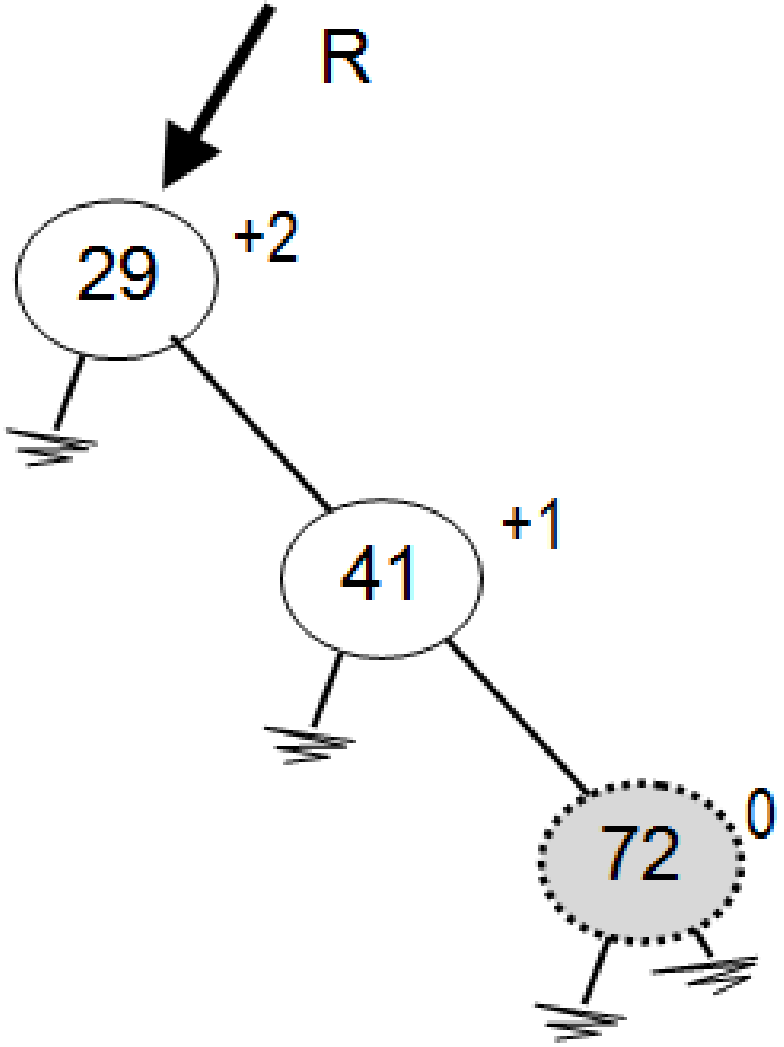
```
Variavel Filho do tipo NodePtr;
Filho = R→Esq;                  // 'A'
R→Esq = Filho→Dir;              // S2
Filho→Dir = R;                  // 'B'
```

```
R→Bal = 0;
Filho→Bal = 0;
```

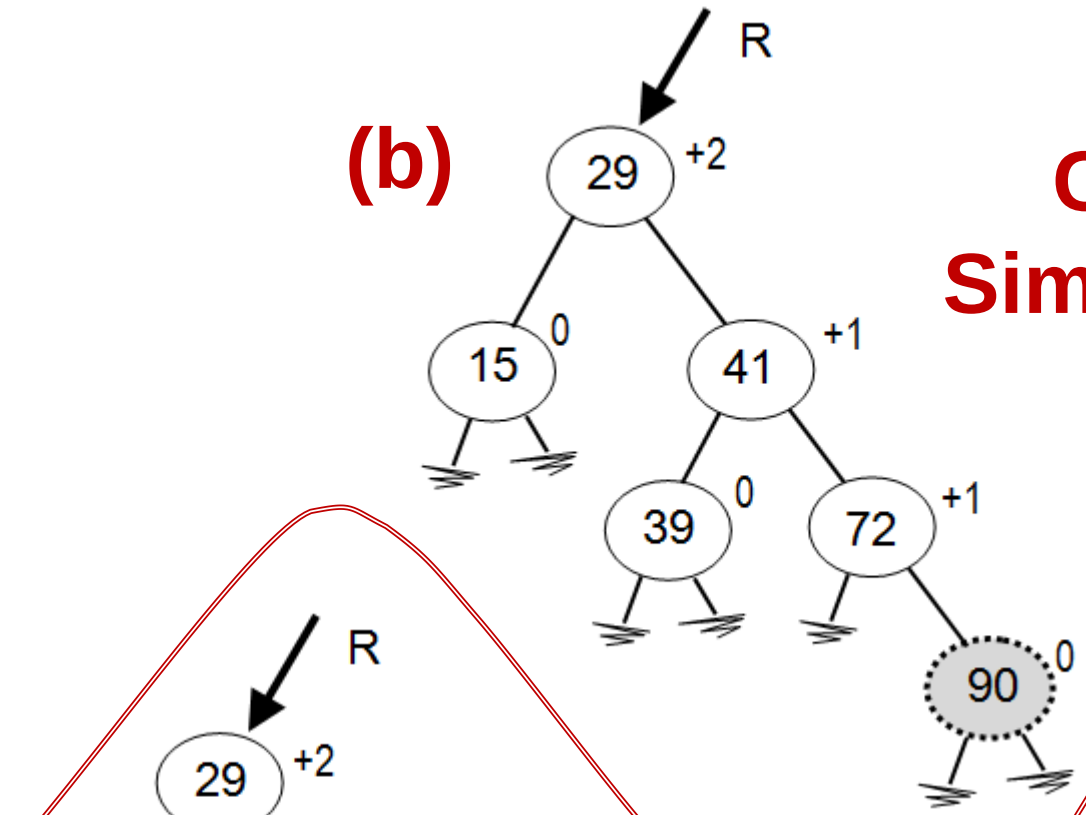
```
R = Filho;                      // 'A'
MudouAltura = Falso;            // H(S1) + 2.
```

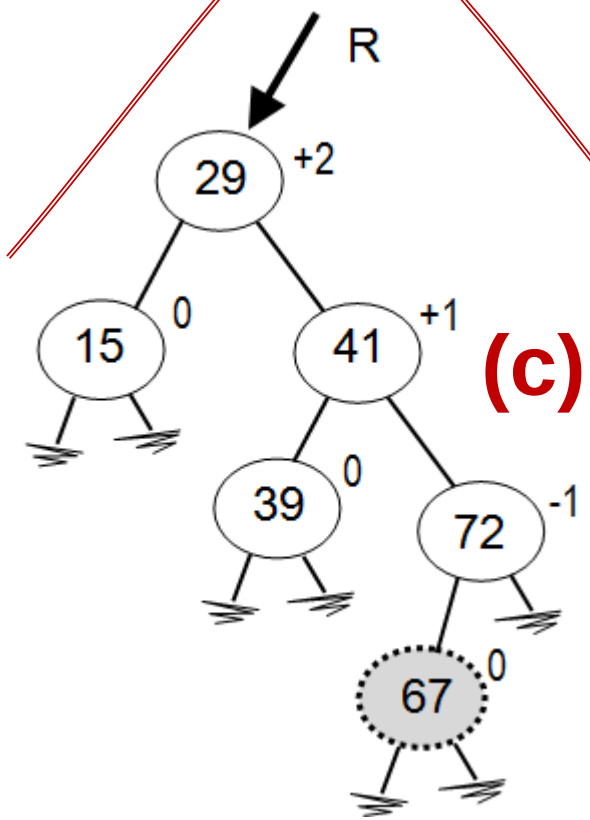
Como Rebalancear?

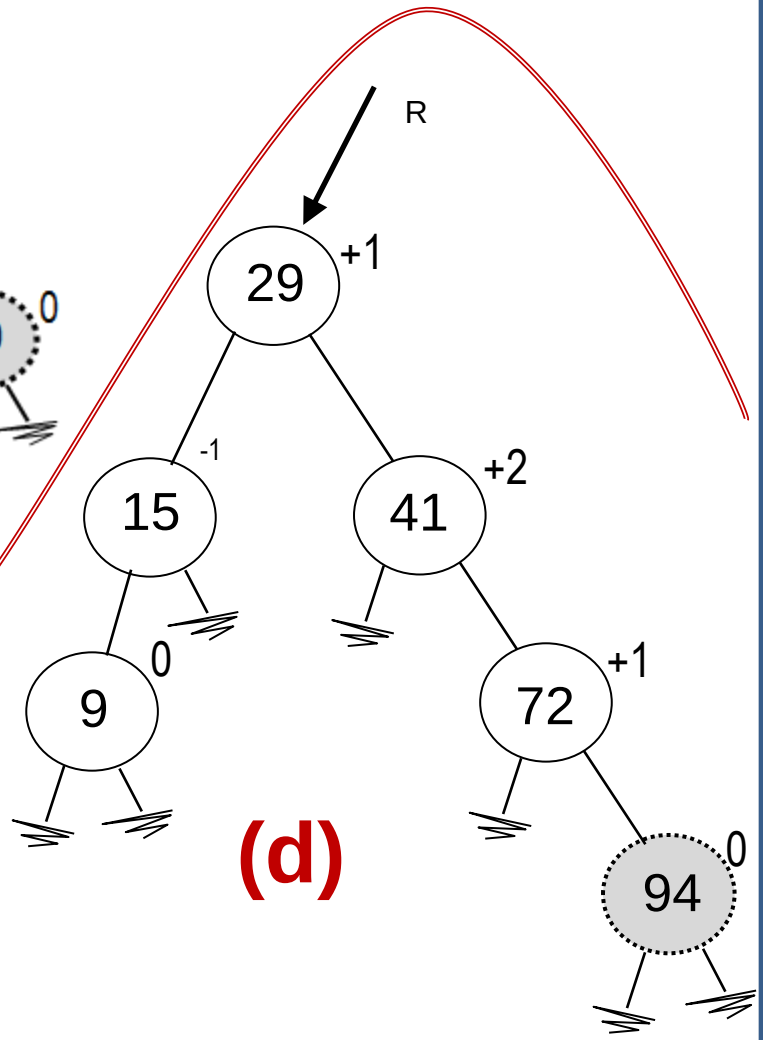
(a)



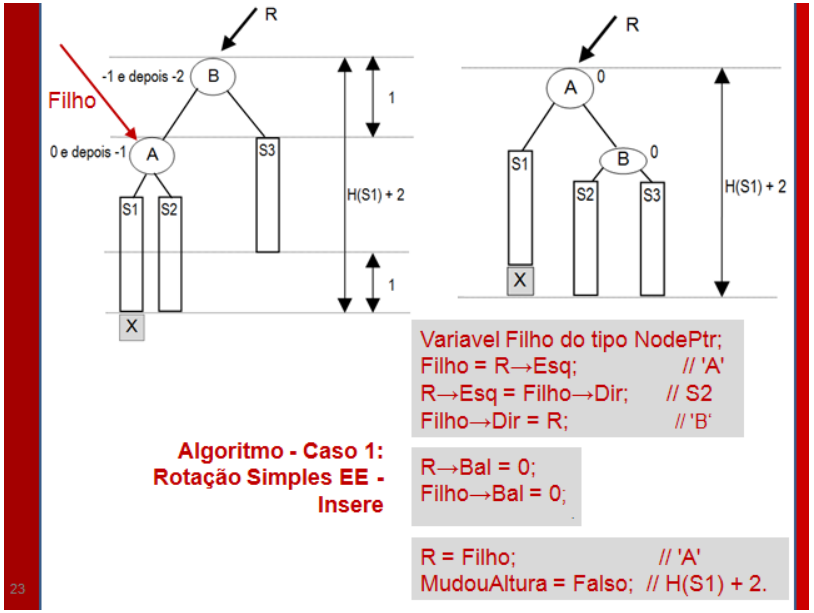
Exercício 9.7
Caso 2 - Rotação Simples DD - Insere







Exercícios – DD Insere

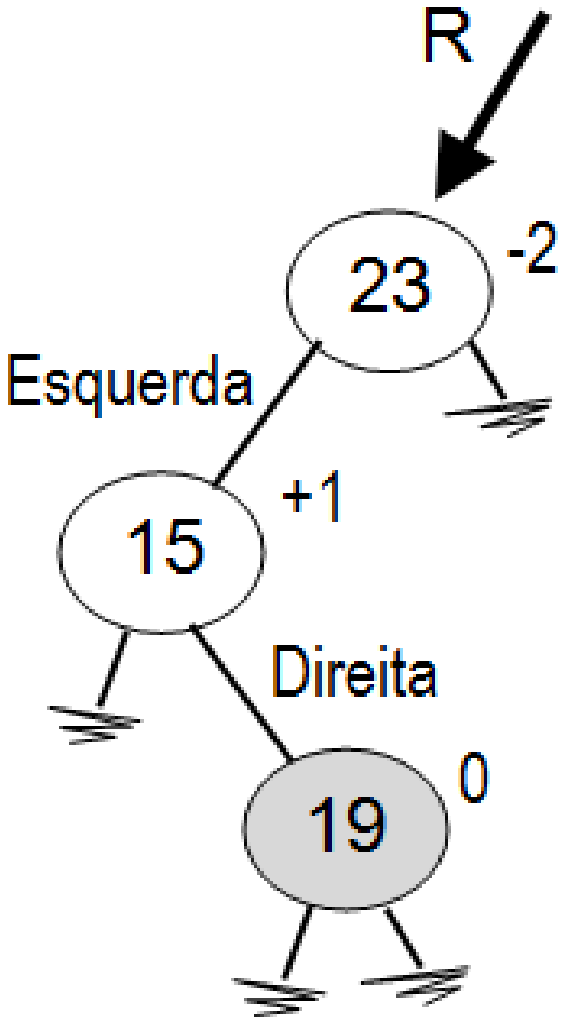


Exercício 9.8 Diagrama - Caso 2: Rotação Simples DD

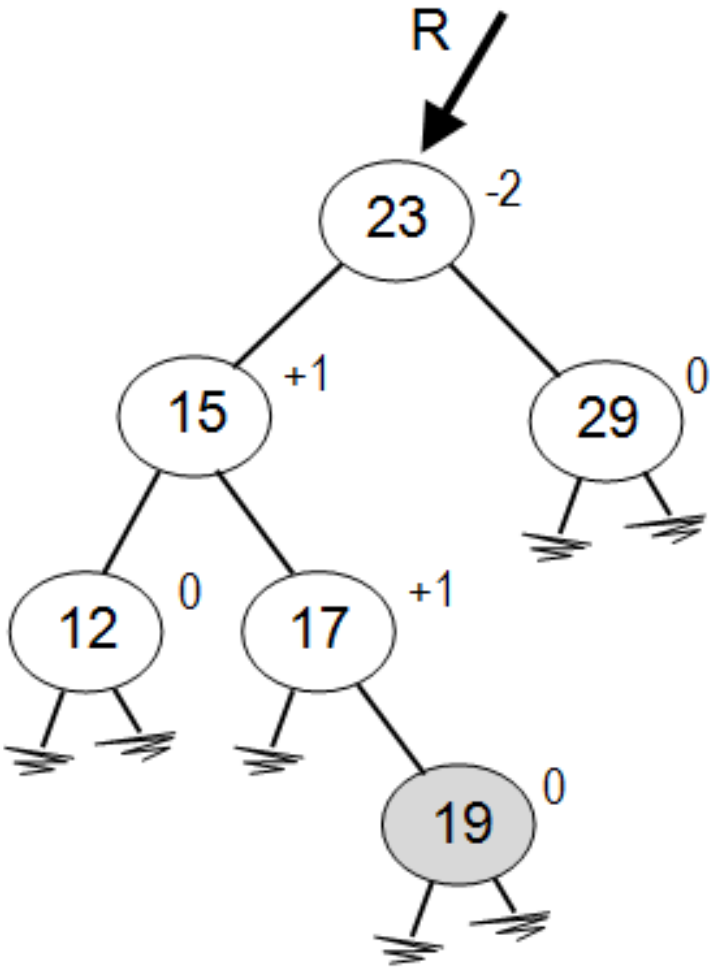
Exercício 9.9 Algoritmo - Caso 2: Rotação Simples DD

Como Rebalancear?

(a)

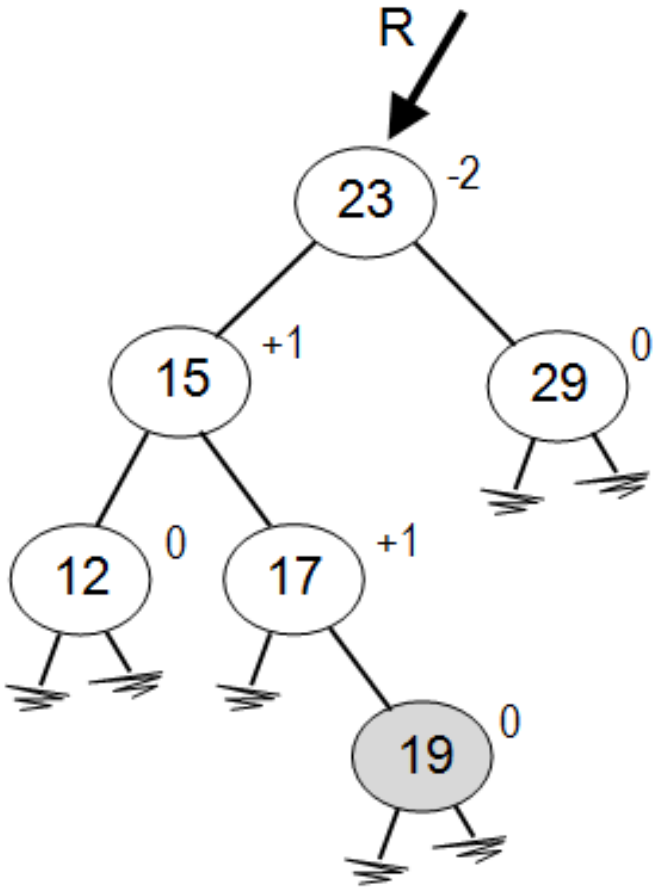


Caso 3 - Rotação Dupla ED - Insere

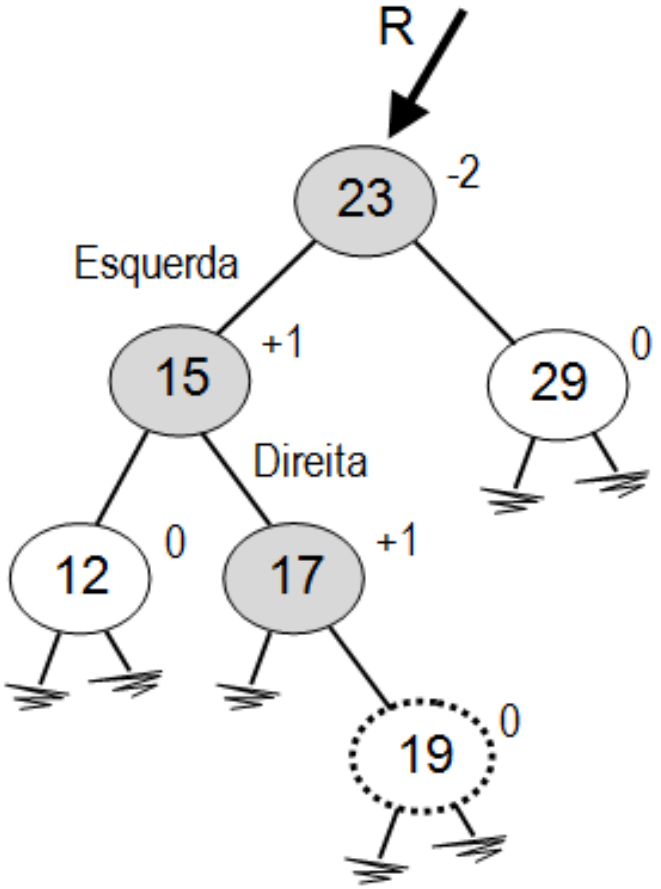


(a) Inseriu e Desbalanceou

Caso 3 - Rotação Dupla ED - Insere

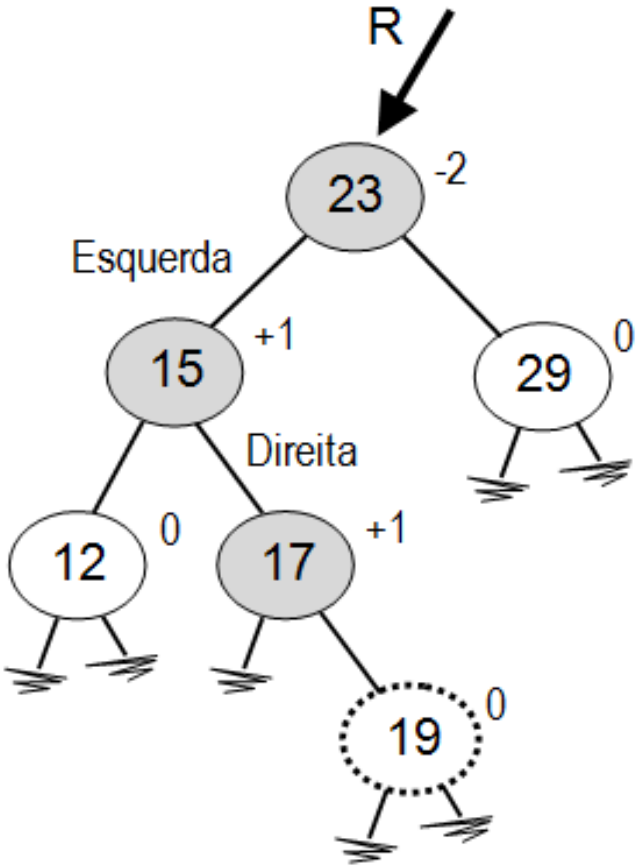


(a) Inseriu e Desbalanceou



(b) Achar os Três Nós Principais

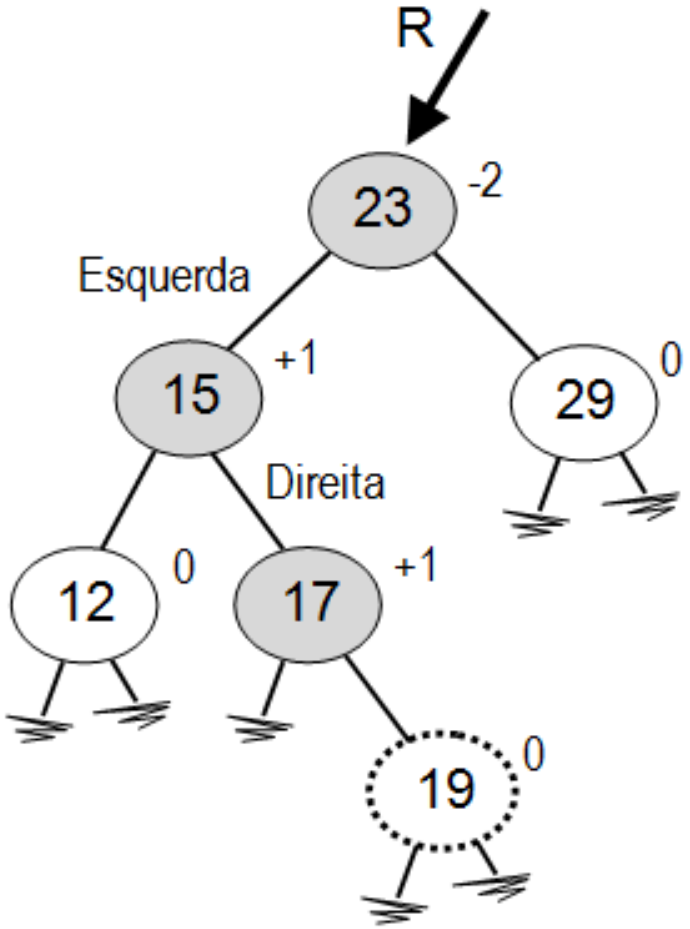
Caso 3 - Rotação Dupla ED - Insere



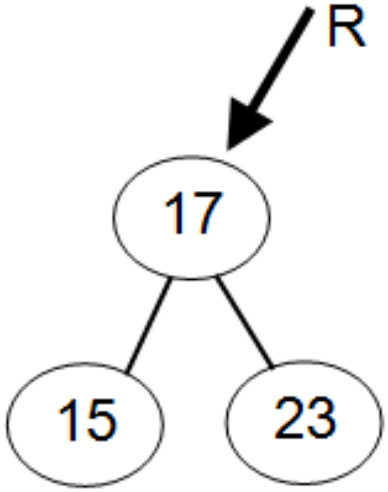
(b) Achar os Três Nós Principais

(c) Reposicionar os Três Nós

Caso 3 - Rotação Dupla ED - Insere

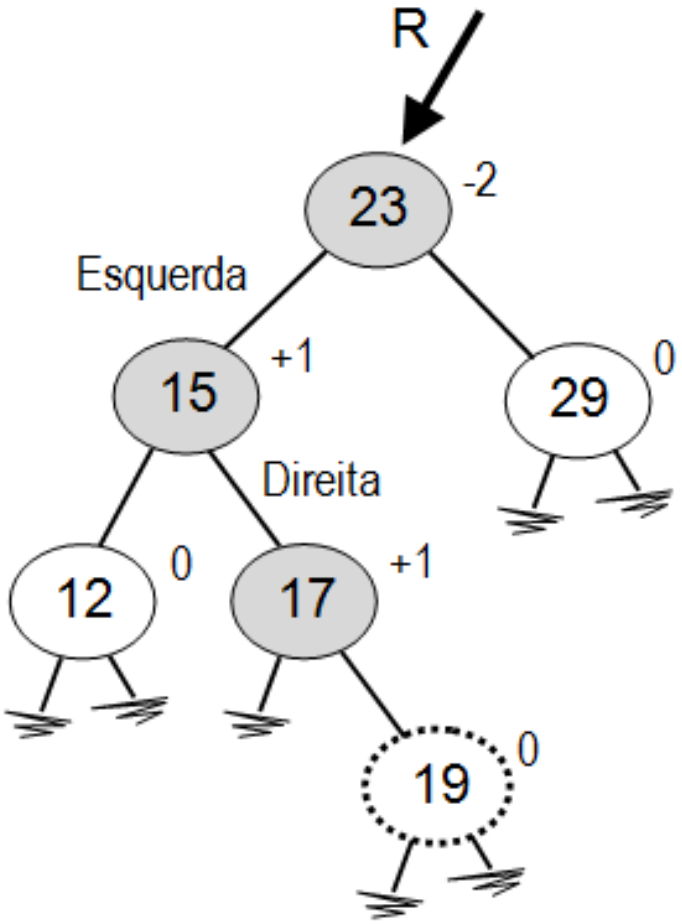


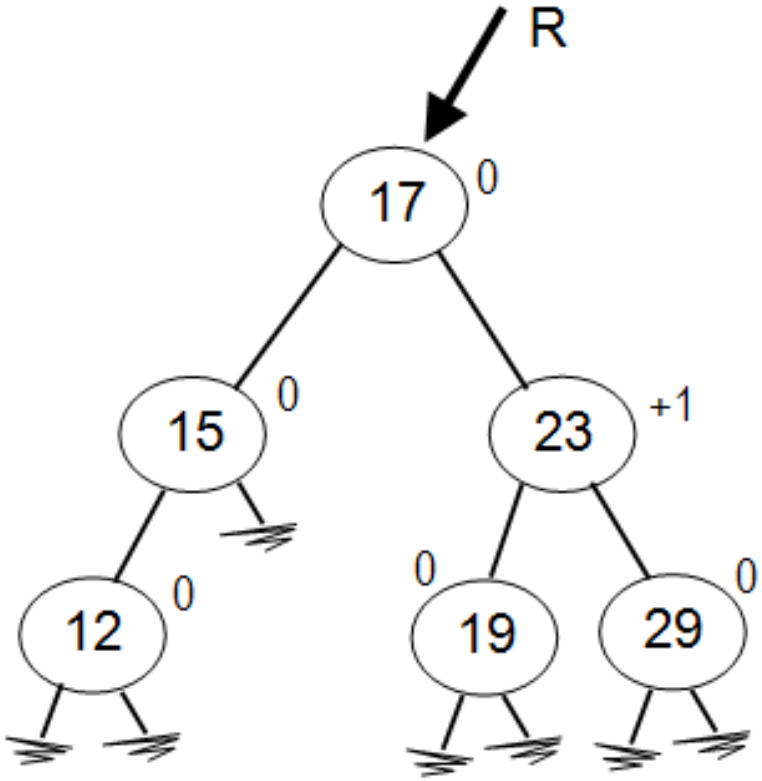
(b) Achar os Três Nós Principais



(c) Reposicionar os Três Nós

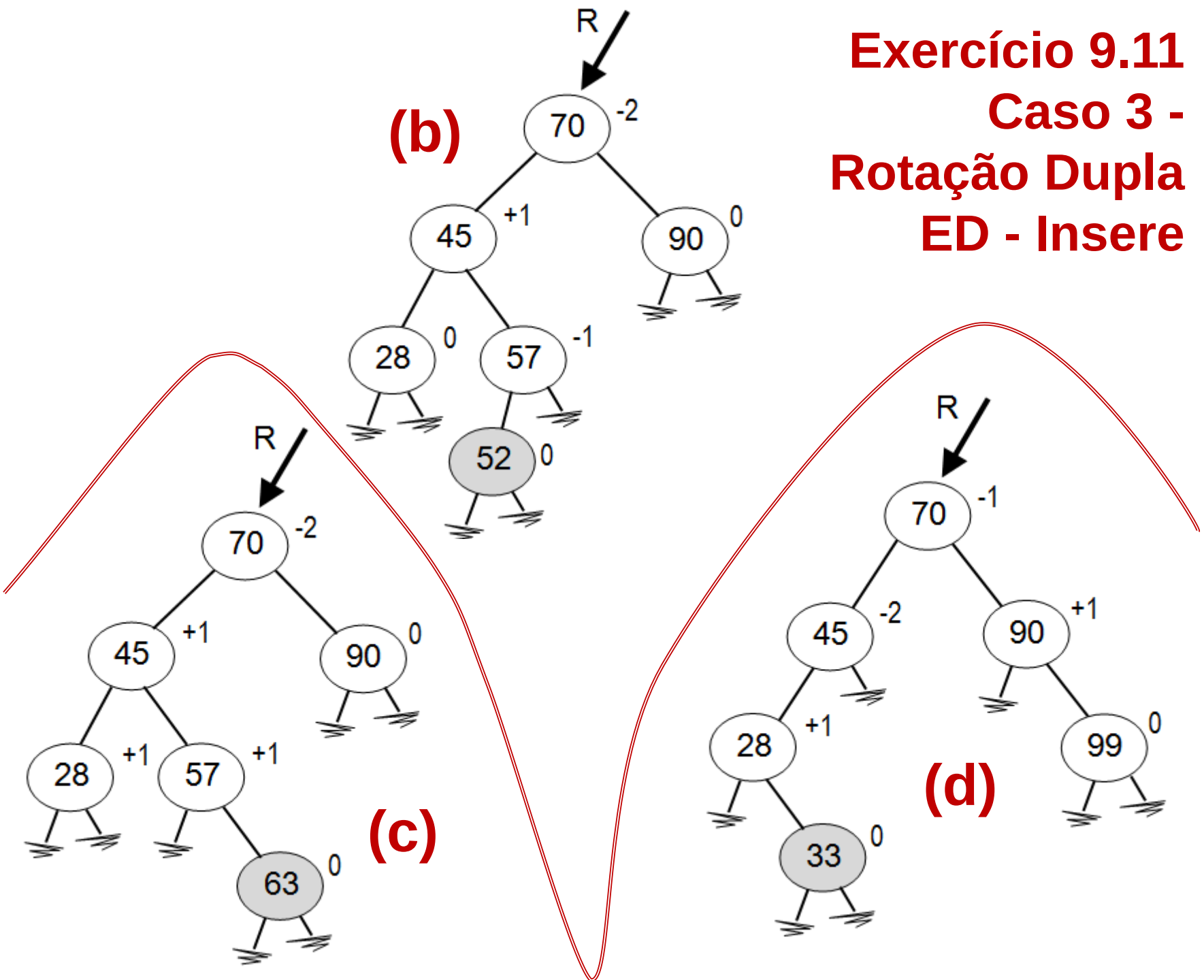
Caso 3 - Rotação Dupla ED - Insere



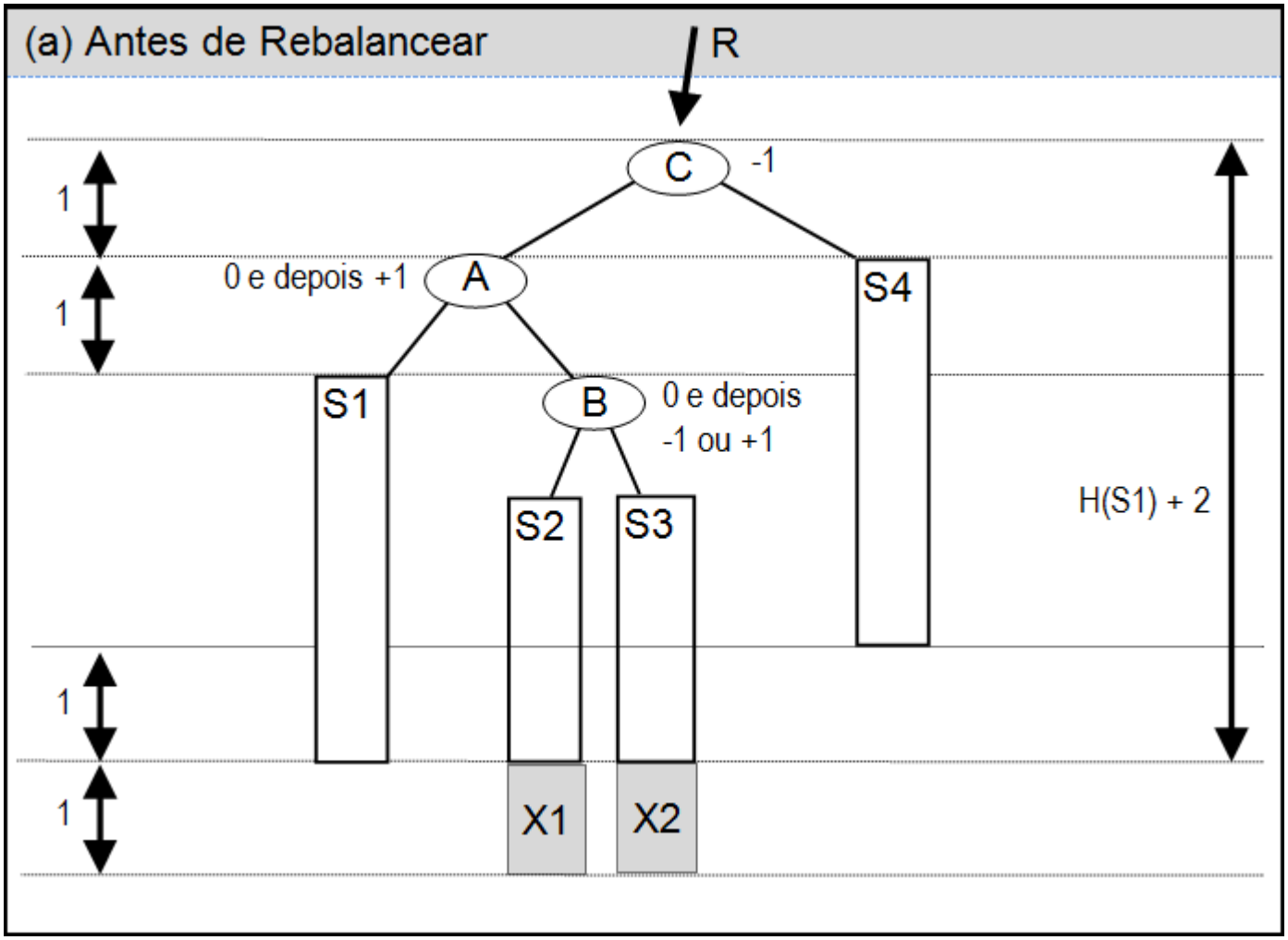


(d) Reposicionar os Demais Nós

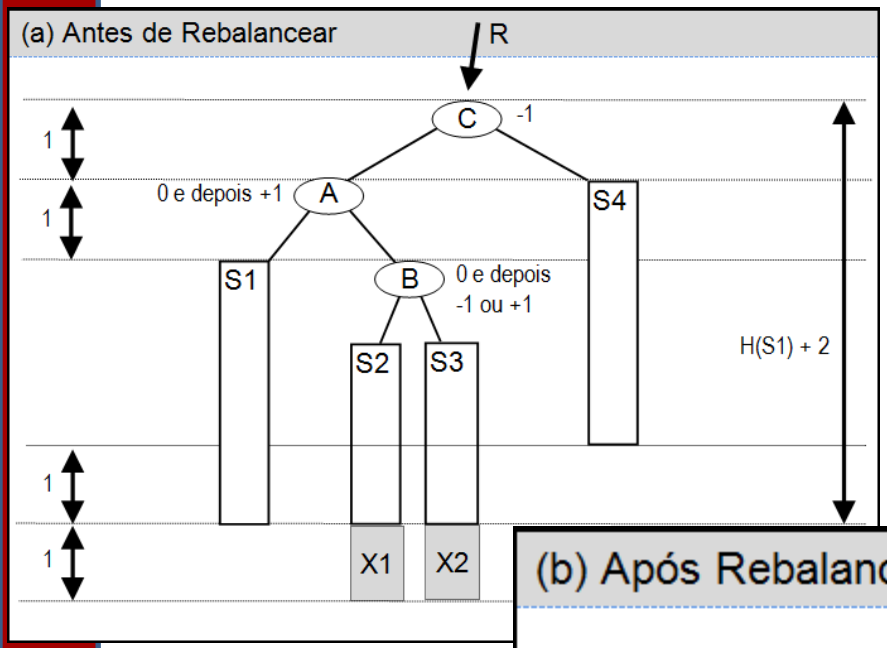
Exercício 9.11
Caso 3 - Rotação Dupla ED - Insere

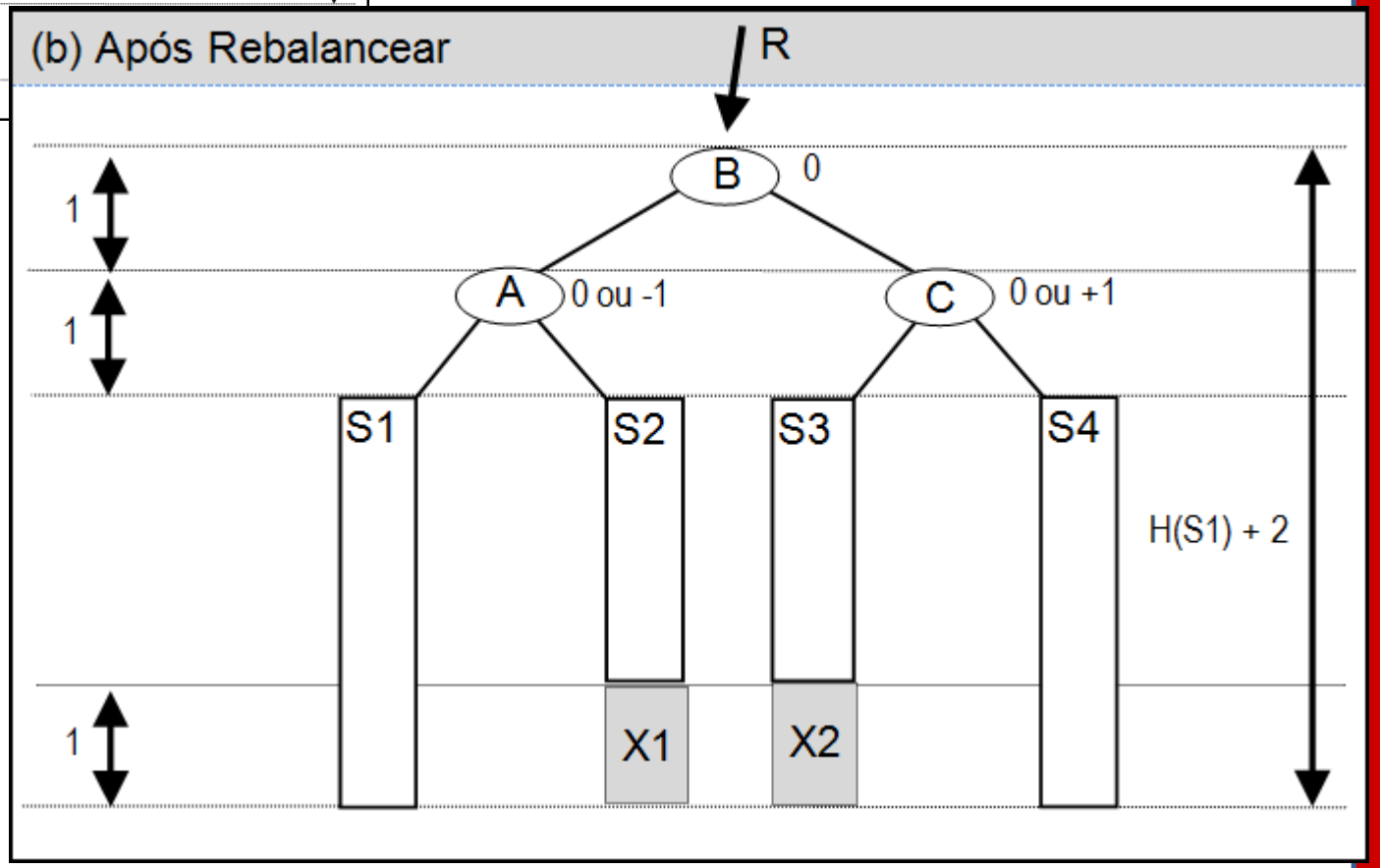


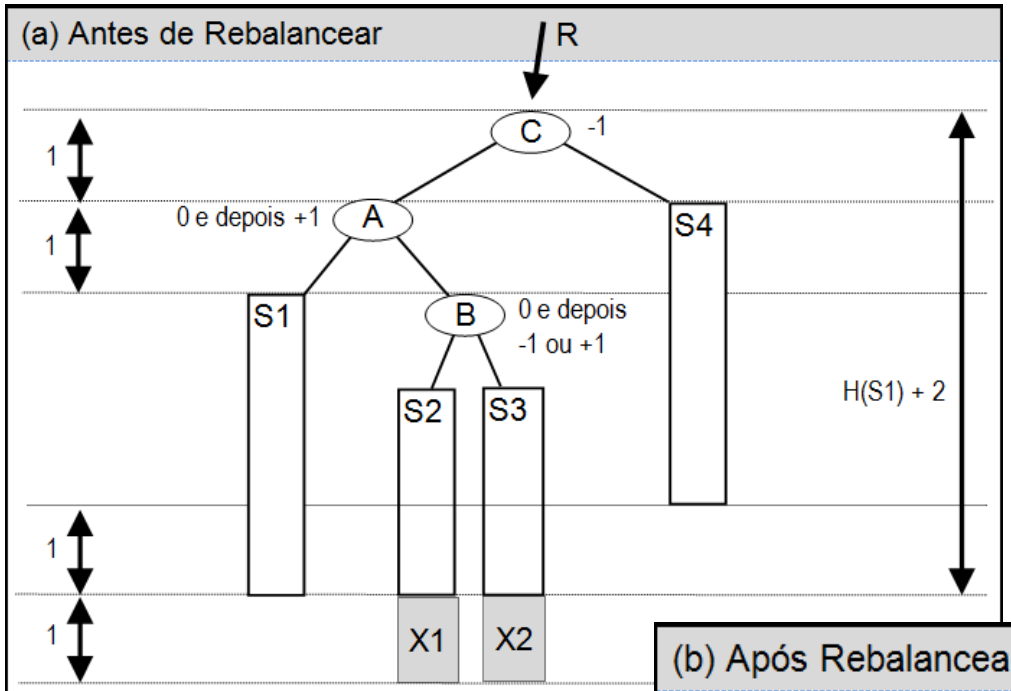
Generalização do Caso 3: Rotação Dupla ED - Insere



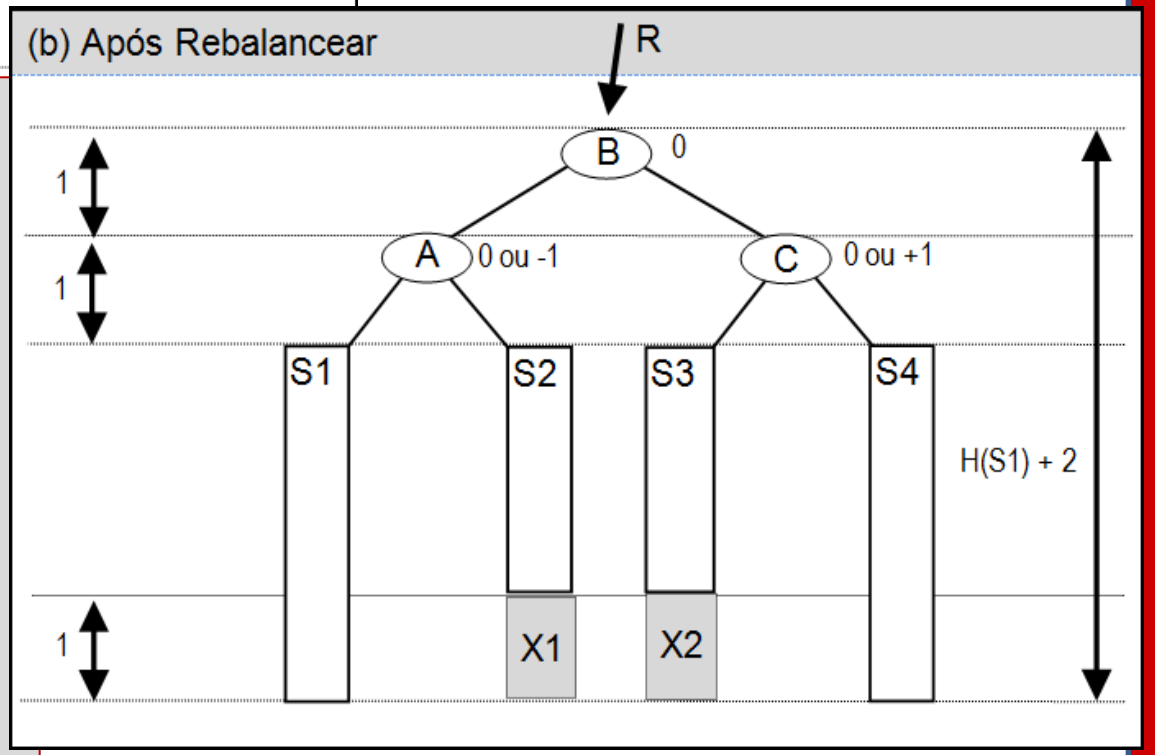
Generalização do Caso 3: Rotação Dupla ED - Insere





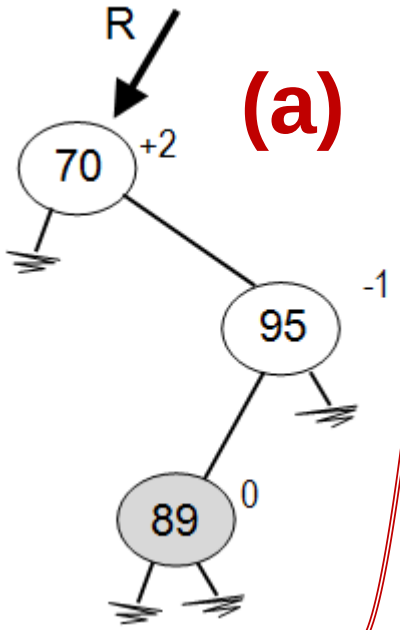


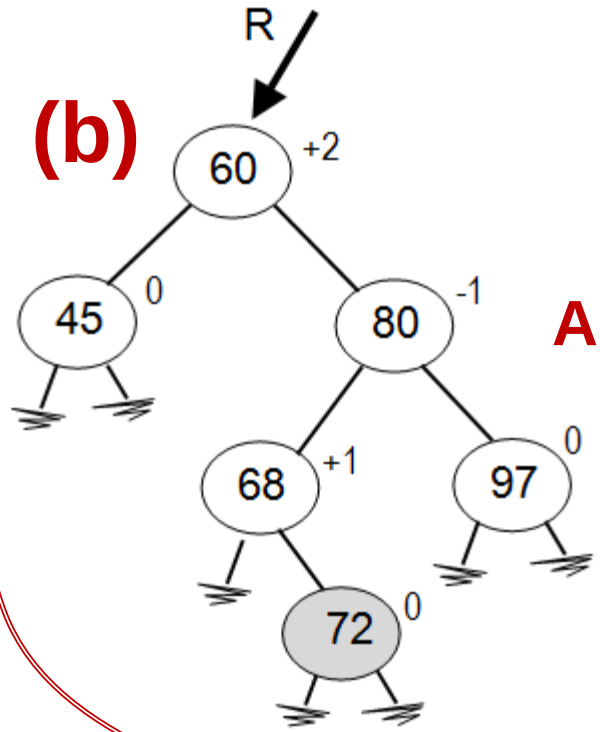

```
variavel Filho do tipo NodePtr;
variavel Neto do tipo NodePtr;
Filho = R→Esq;                 // 'A'
Neto = Filho→Dir;              // 'B'

Filho→Dir = Neto→Esq;  // S2
Neto→Esq = Filho;              // 'A'

R→Esq = Neto→Dir;              // S3
Neto→Dir = R;                  // 'C'
```

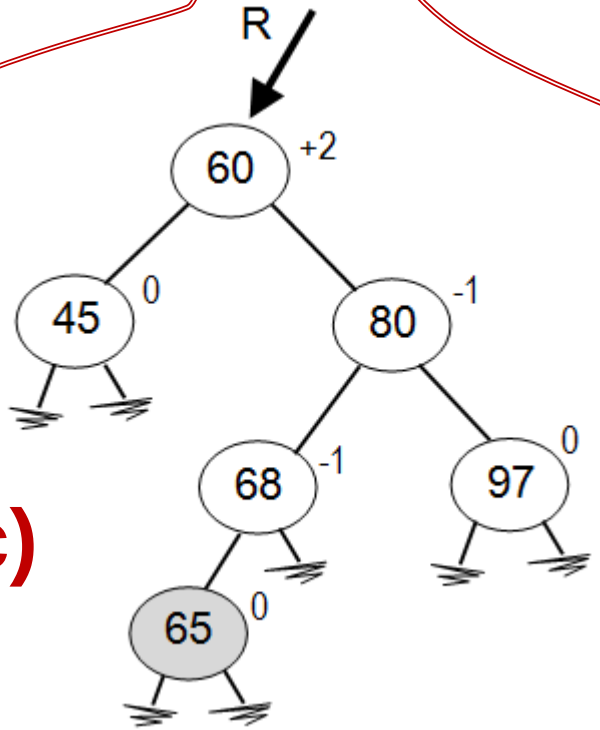


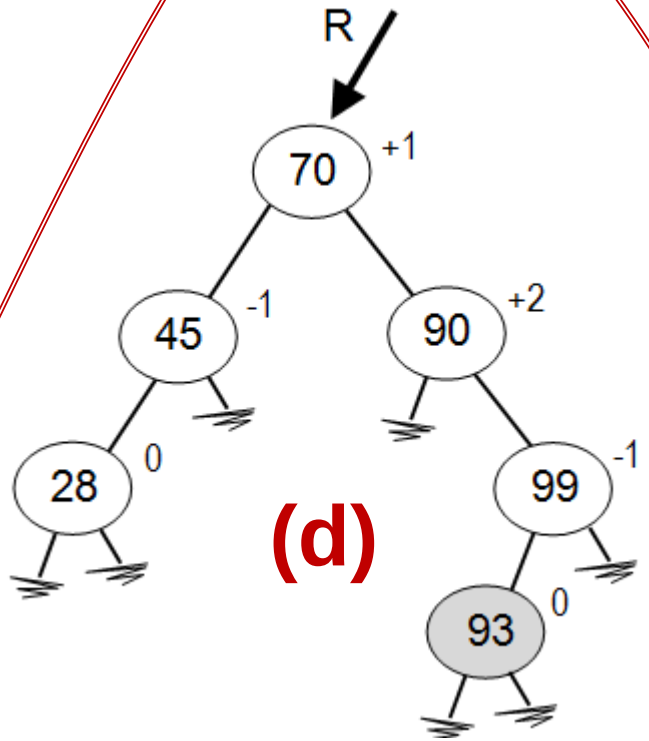

```
Caso Neto→Bal for
    -1 :  { R→Bal = 1;      // inseriu X1
            Filho→Bal =  0;
            Neto→Bal = 0; }
    +1 : { R→Bal = O;       // inseriu X2
            Filho→Bal =  -1;
            Neto→Bal = 0; }
     0 :   { R→Bal = 0;  //  inseriu 'B'
             Filho→Bal = 0;
             Neto→Bal = 0; }
R = Neto;  // 'B'
MudouAltura = Falso;  //  H(S1) + 2.
```

Exercícios 9.12 a 9.14 Exemplos, Diagrama e Algoritmo - Caso 4: DE Insere





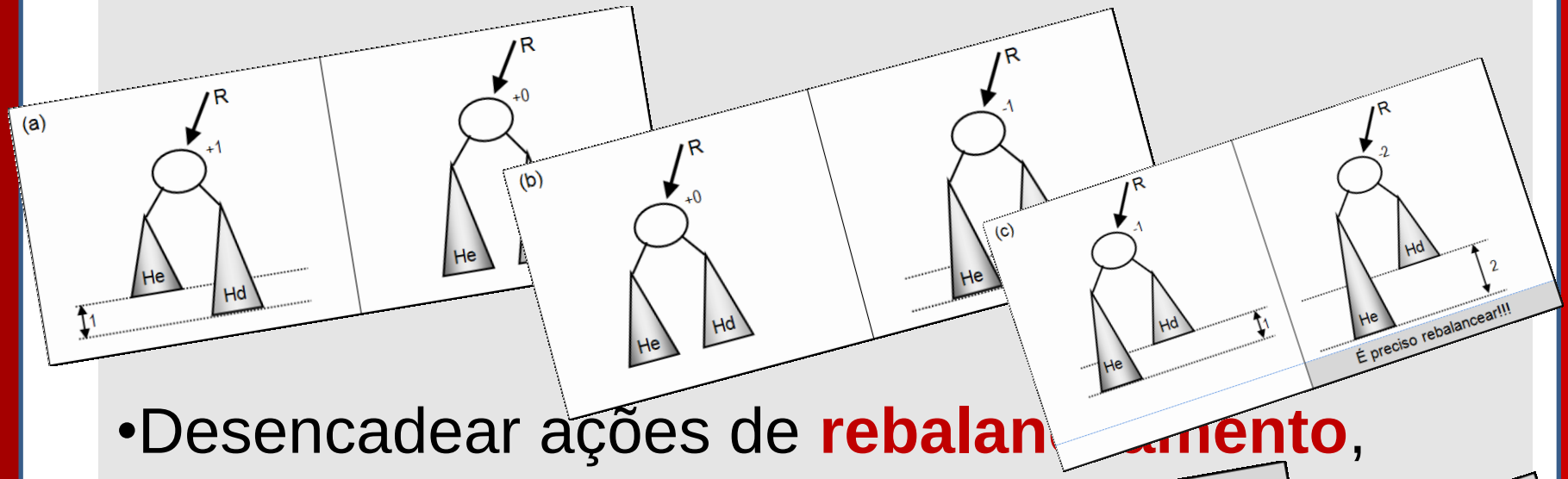




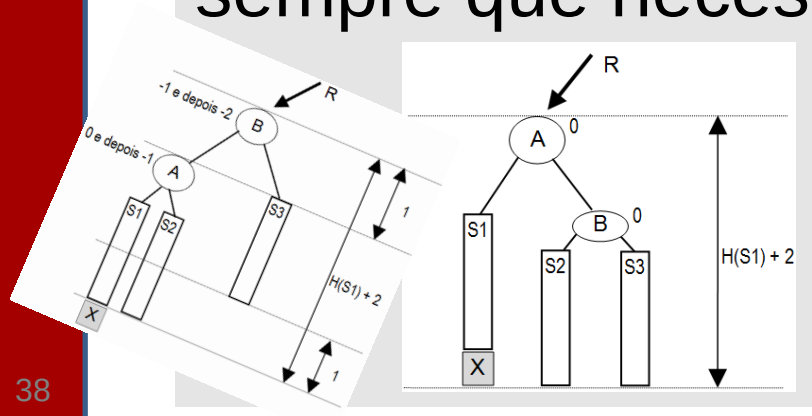
Para Manter uma ABB Balanceada:

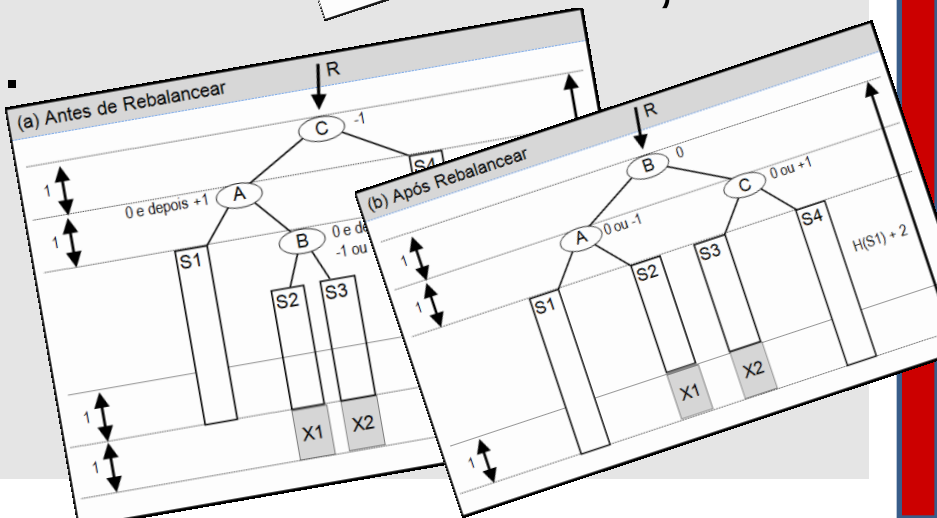
Alterar algoritmos de **inserção** e eliminação para:

- **Monitorar** o Balanceamento da Árvore; e



- Desencadear ações de **rebalanceamento**, sempre que necessário.





Exercício 9.15 Adaptar o Algoritmo Insere - ABB para ABBB

Insere (parâmetro por referência **R** do tipo ABB, parâmetro **X** do tipo Inteiro, parâmetro por referência **Ok** do tipo Boolean);

Variável P do tipo NodePtr;

```
Se (R == Null)
Então {   P = NewNode;      // Caso 1: Achou o lugar; insere e acaba
          P→Info = X;
          P→Dir = Null;
          P→Esq = Null;
          R = P;
           P = Null;
          Ok = Verdadeiro; }
Senão {  Se (X == R→Info)
           Então  Ok = Falso; // Caso 2: X já está na árvore; não insere;
           Senão { Se (R→Info> X)
                      Então Insere (R→Esq, X , Ok)  // Caso 3: tenta na Es
                      Senão Insere(R→Dir, X, Ok); // Caso 4: tenta na Dir
                     } // fim senão
         } // fim senão
} // fim Insere ABB
```

Exercício 9.15 Insere - ABBB

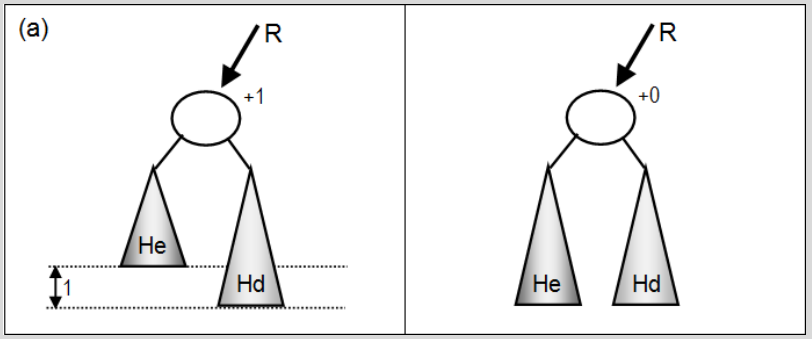
Insere (parâmetro por referência **R** do tipo ABB, parâmetro **X** do tipo Inteiro, parâmetro por referência **Ok** do tipo Boolean);

Variável P do tipo NodePtr;

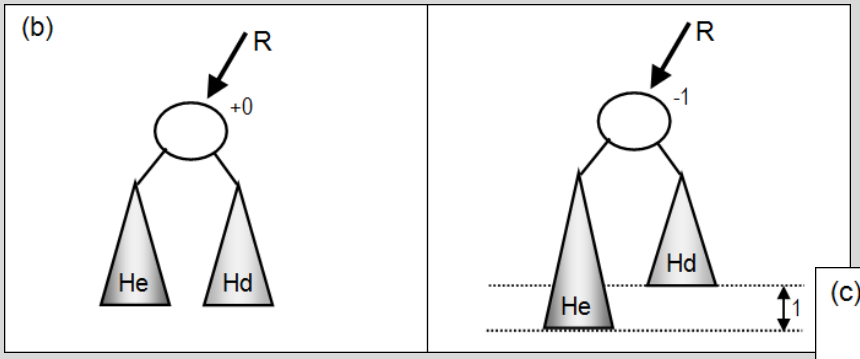
```
Se (R == Null)
Então {   P = NewNode;  P→Info = X; P→Dir = Null; P→Esq = Null; R = P;
          P = Null; Ok = Verdadeiro; P→Bal = 0; }
Senão {  Se (X == R→Info)
          Então  Ok = Falso;
          Senão { Se (R→Info> X)
                    Então { Insere (R→Esq, X , Ok);
                            // Monitora balanceamento voltando de inserir
                            // na Subárvore Esquerda de R. Se for
                            // preciso, desencadeia Rebalanceamento  }
                    Senão { Insere(R→Dir, X, Ok);
                            // monitora balanceamento voltando de inserir
                            // na Subárvore Direita de R. Se for
                            // preciso, desencadeia Rebalanceamento  }
                  }
        }
}
```

Monitora balanceamento voltando de inserir na Esquerda...

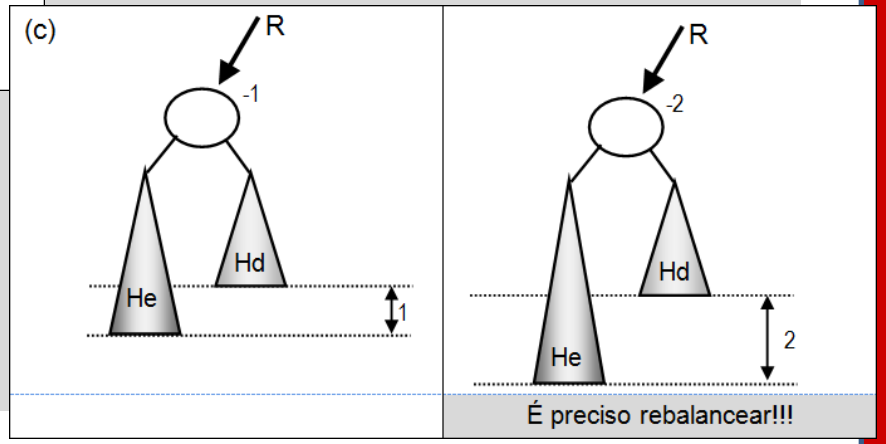
Insere (**R→Esq,** X , Ok);
Se MudouAltura // Se a Subárvore esq cresceu..
Então Caso R→Bal for:
+1: { R→Bal = 0;
MudouAltura = Falso;}



0: R→Bal= -1; // MudouAltura continua Verdadeiro



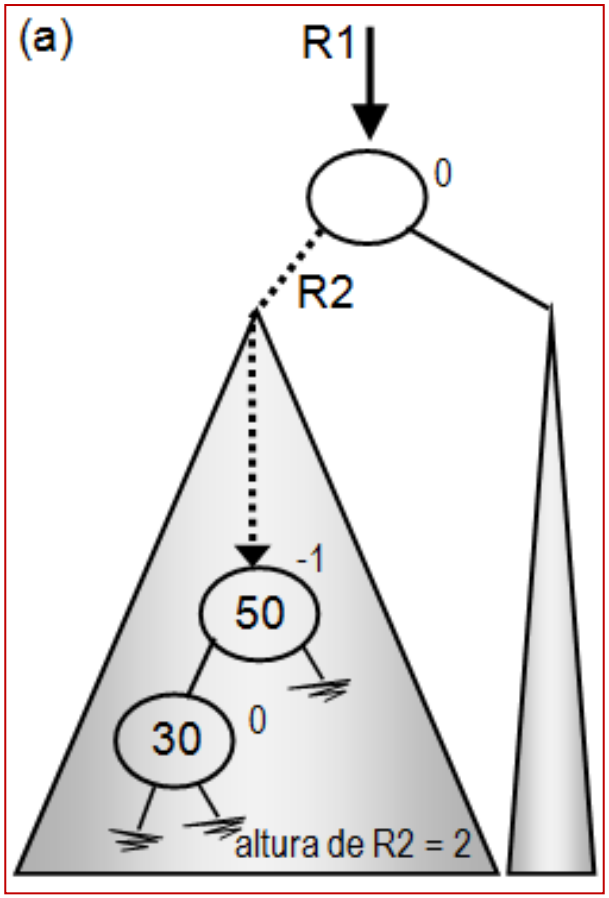
-1: /* É preciso rebalancear! */
{ Filho = R→Esq;
Se (Filho→Bal == +1)
Então RotDuplaEDInsere;
Senão RotSimplesEEInsere; }

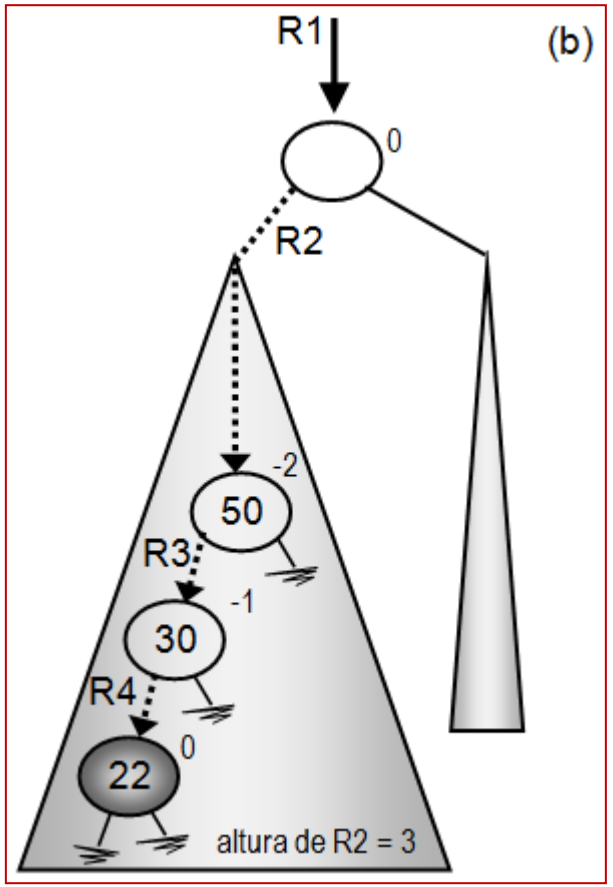


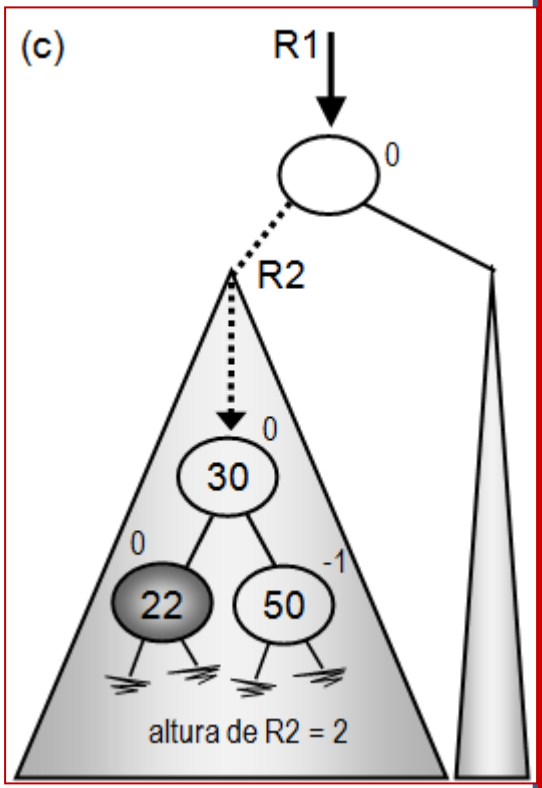
Monitora balanceamento voltando de inserir na Subárvore Direita. Se for preciso, desencadeia Rebalanceamento

```
Insere (R→Dir, X , Ok);
Se MudouAltura // Se a Subárvore Direita cresceu..
Então Caso R→Bal for:
Se MudouAltura   // se a Subárvore Direita cresceu...
então Caso R→Bal for:
        -1: { R→Bal = 0; MudouAltura = Falso; };

         0: R→Bal=1; // MudouAltura continua Verdadeiro

        +1: { Filho = R→Dir;      // É preciso rebalancear!!
              Se (Filho→Bal = -1)
              Então RotDuplaDEInsere;
              Senão RotSimplesDDInsere;}
```

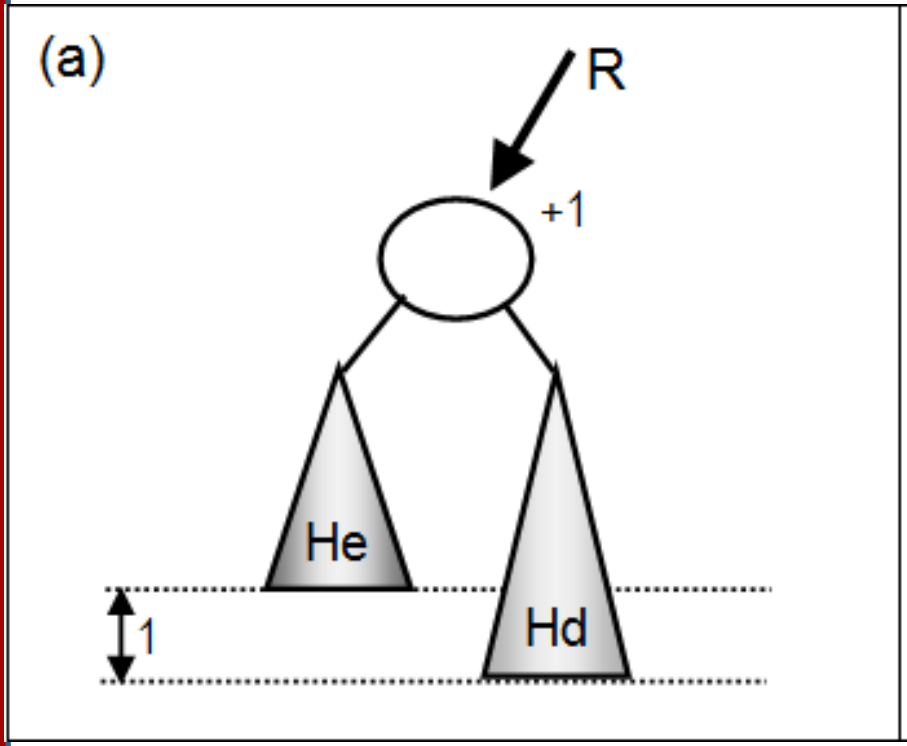
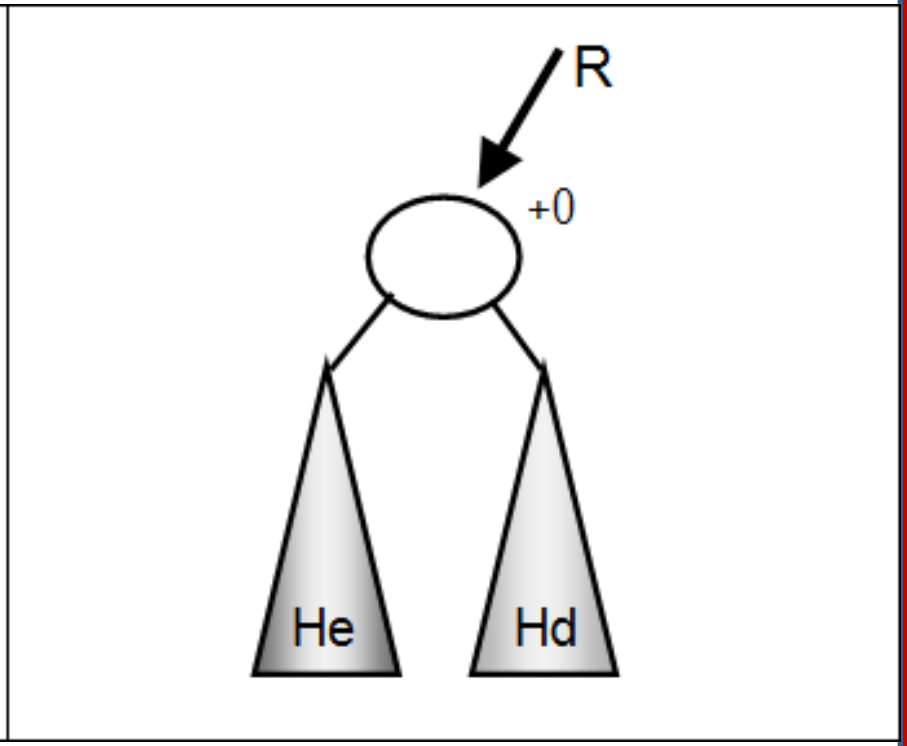
Insere em ABBB: Exemplo de Execução



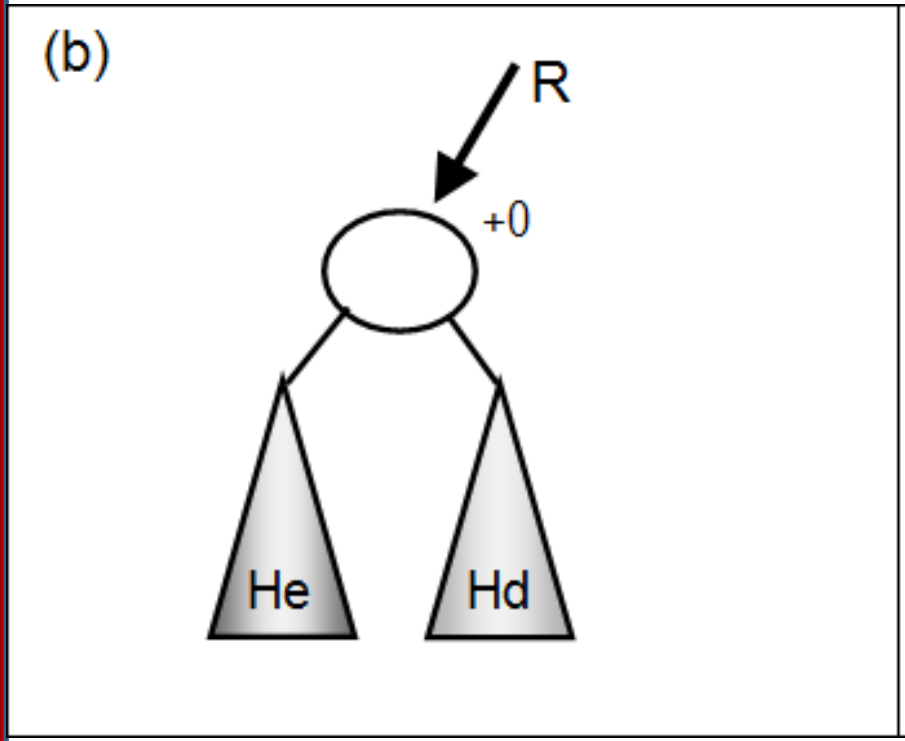
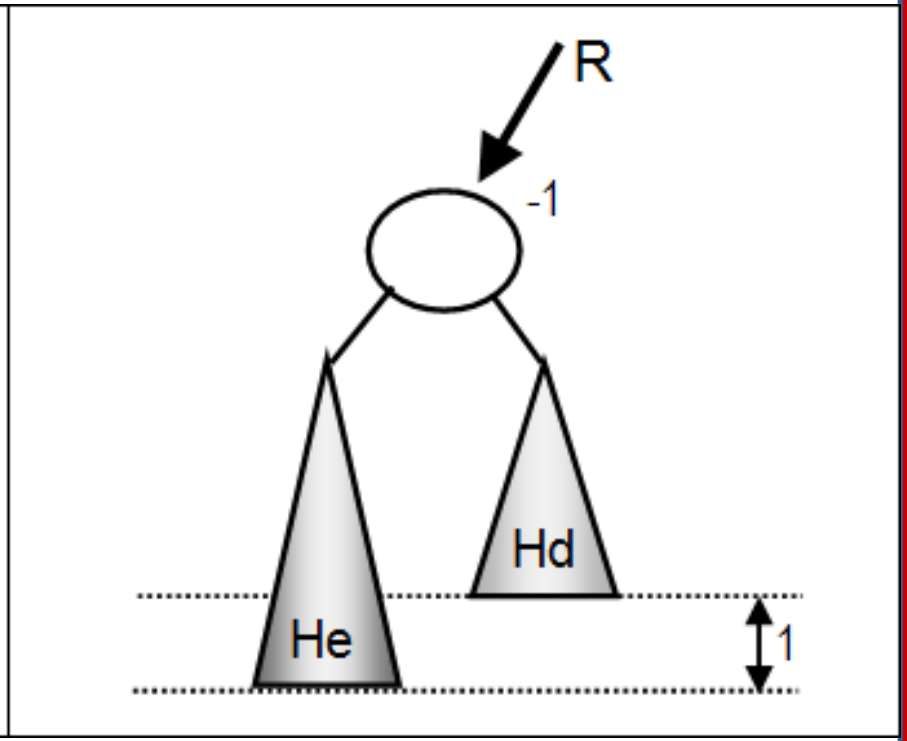




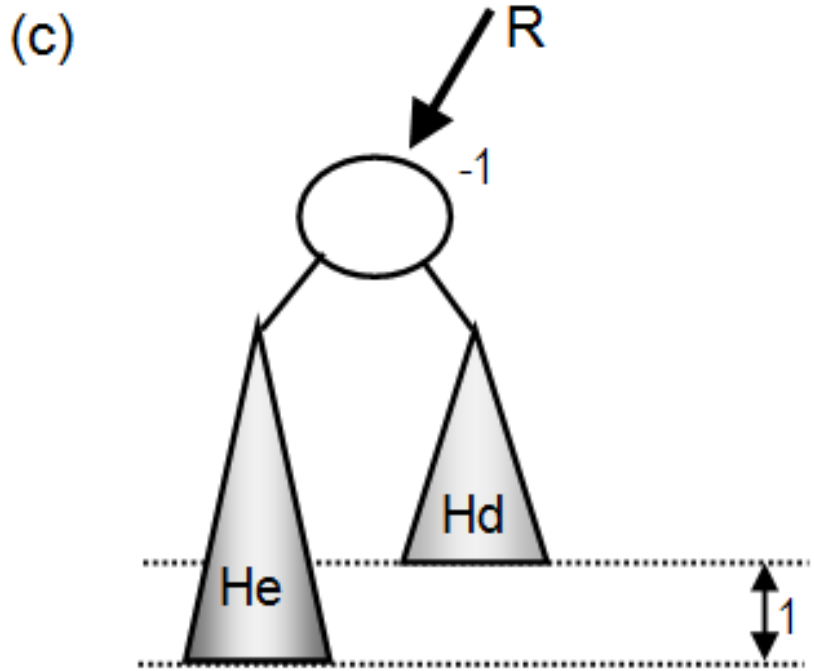
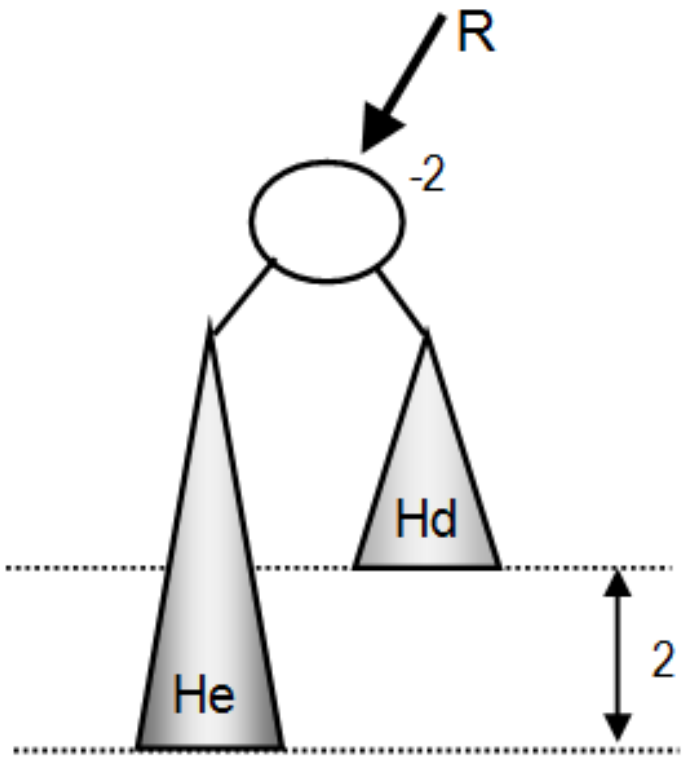
Monitorando o Balanceamento ao Remover Elementos na Subárvore Direita

Antes da Remoção	Após a Remoção
	

Monitorando o Balanceamento ao Remover Elementos na Subárvore Direita

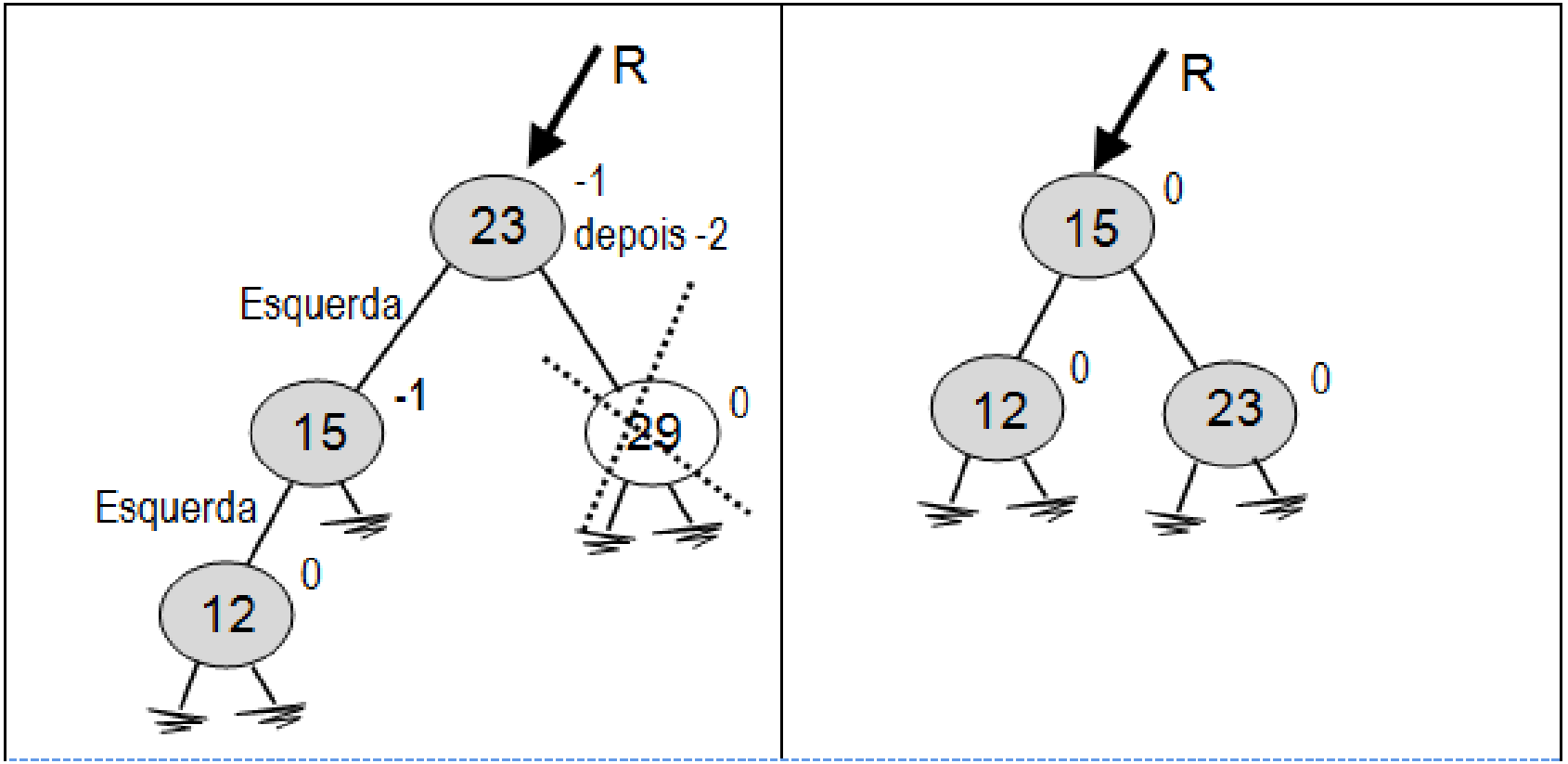
Antes da Remoção	Após a Remoção
	

Monitorando o Balanceamento ao Remover Elementos na Subárvore Direita

Antes da Remoção	Após a Remoção
	
	É preciso rebalancear!!!

Caso 1 - Rotação Simples EE - Remove

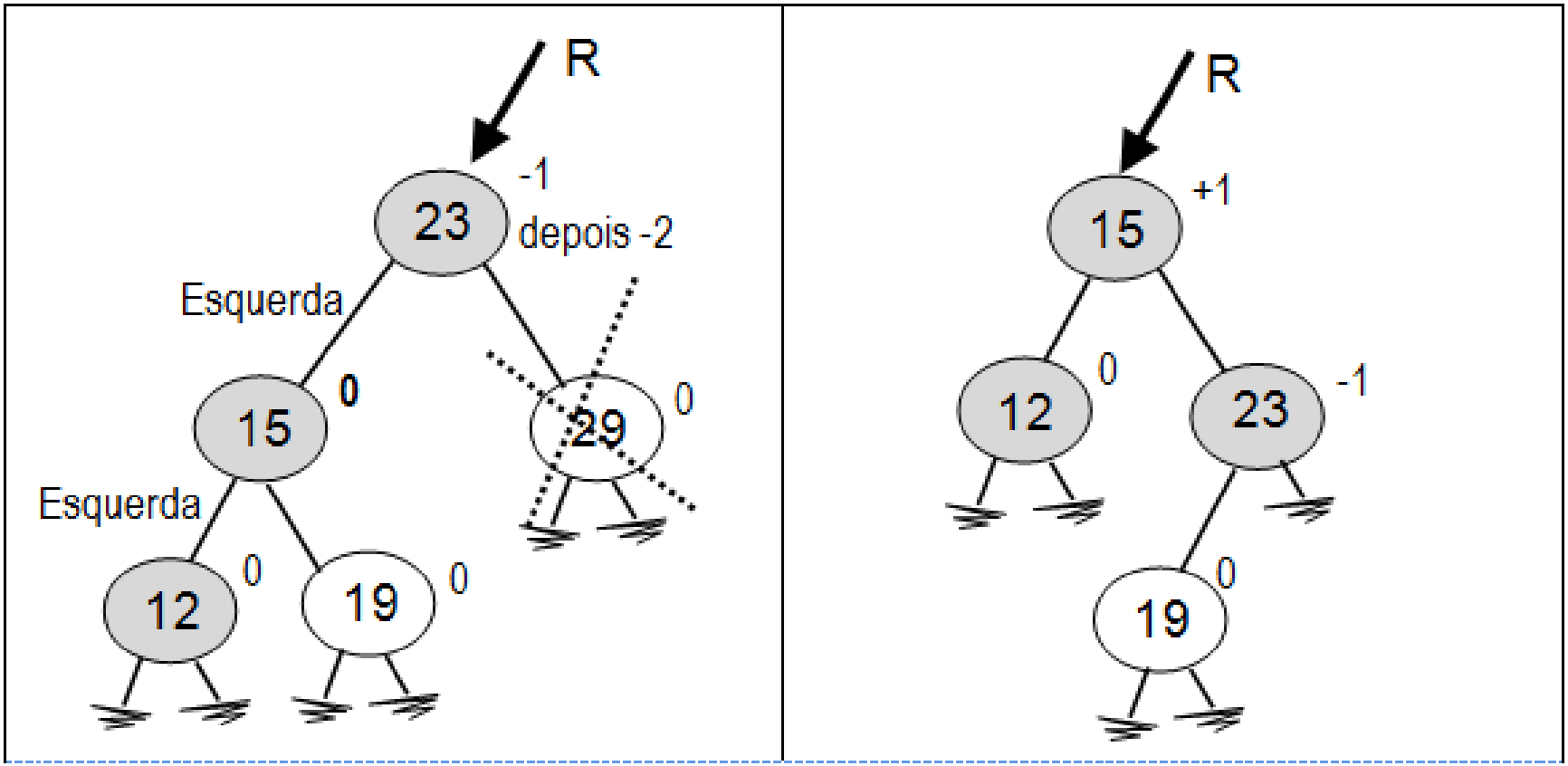
(Filho -> Bal = -1)



(a) Caso EE mas MudouAltura não será atualizada para Falso, como no Insere

Caso 1 - Rotação Simples EE - Remove

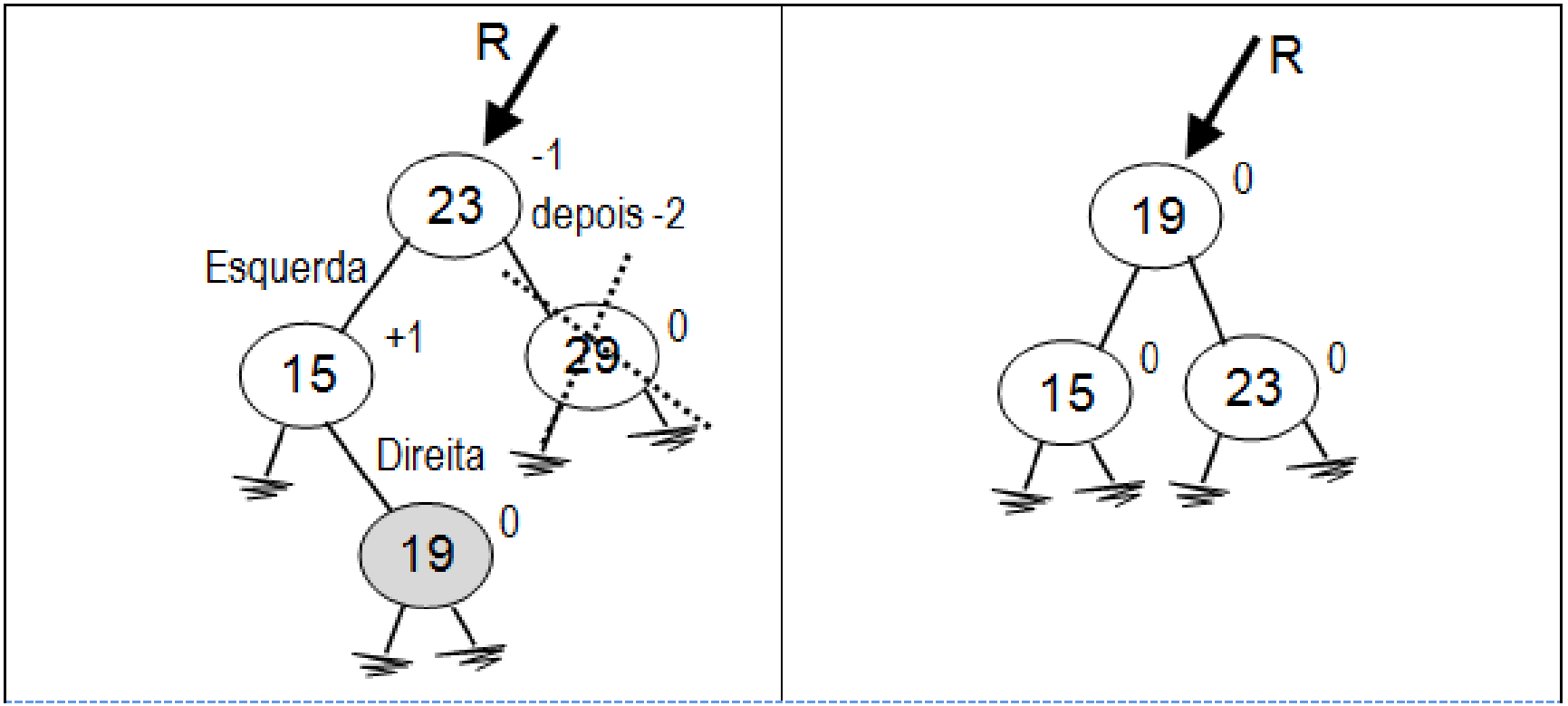
(Filho -> Bal = 0)



(b) Caso EE mas Fatores de Balanceamento não serão atualizados para Zero, como no Insere

Caso 1 - Rotação Simples EE - Remove

(Filho -> Bal = +1)



(c) Caso ED mas MudouAltura não será atualizada para Falso, como no Insere

Exercícios

Exercícios 9.16 e 9.17 Remove ABBB - Rebalanceamento Manual casos EE e ED

Exercícios 9.18 a 9.21 Generalização dos Casos 5, 6, 7 e 8 EE, DD, ED e DE do Remove - Diagrama e Algoritmo

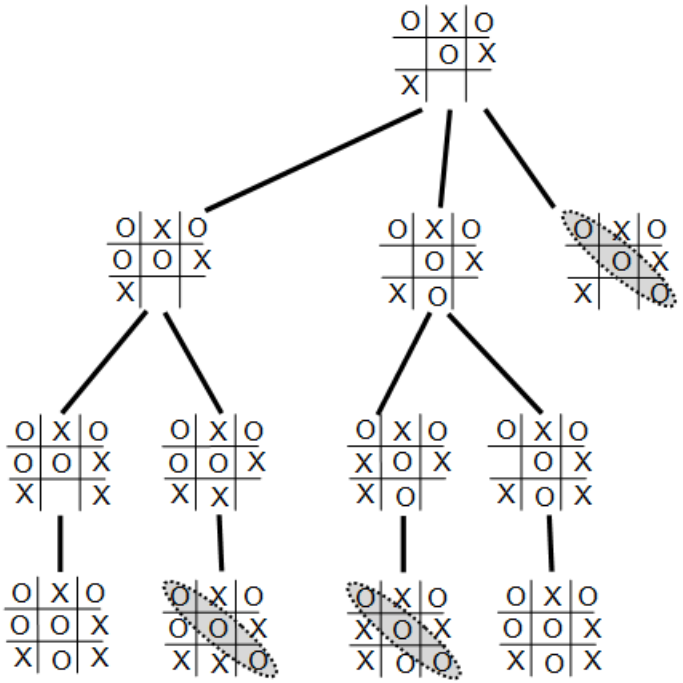
Exercício 9.22 Algoritmo Remove para uma Árvore Binária de Busca Balanceada – ABBB (ajustar o Remove de ABB)

Exercício 9.23 Implemente uma Árvore Binária de Busca Balanceada - ABBB em uma Linguagem de Programação

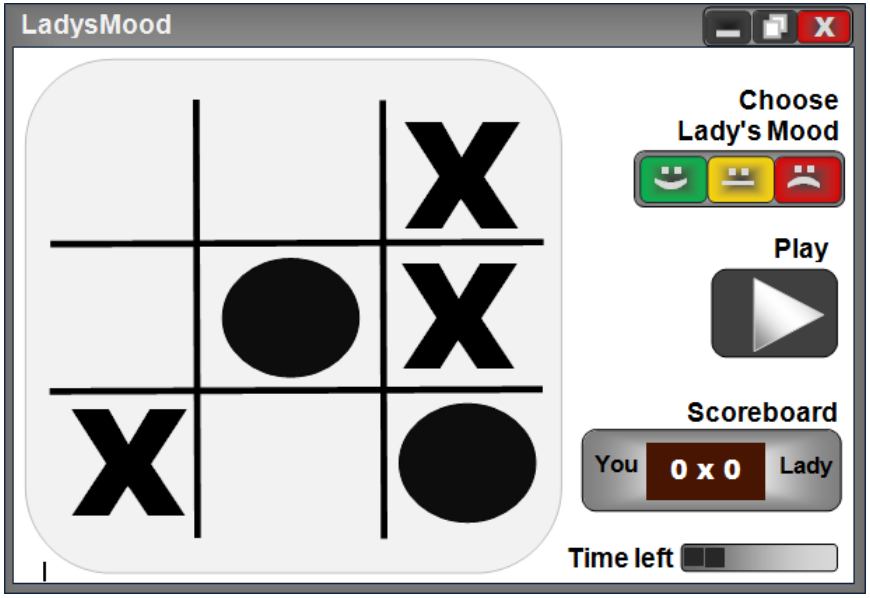
Avanço de Projeto

Exercício 8.12 Discutir Aplicações de Árvores em Jogos

Exercício 8.13 Avançar o Projeto do Desafio 4: Defina Regras, Escolha um Nome e Inicie o Desenvolvimento do Seu *Jogo*



Comece a Desenvolver Seu Jogo Agora!



Dê personalidade e inteligência ao seu jogo! Mostre para os seus amigos!

Estruturas de Dados com Jogos

Aprender a programar pode ser divertido!