Universidad del Valle
Sede Tuluá

Tema 07

ARBOLES ROJINEGROS

Universidad del Valle
Sede Tuluá

QUÉ SON?

- Un árbol rojinegro es un árbol de búsqueda binaria con un bit extra de almacenamiento por nodo: su color, el cual puede ser rojo o negro

- Lo anterior indica que cada nodo del árbol contiene los campos: ***color***, ***key***, ***left***, ***right***, y ***p***

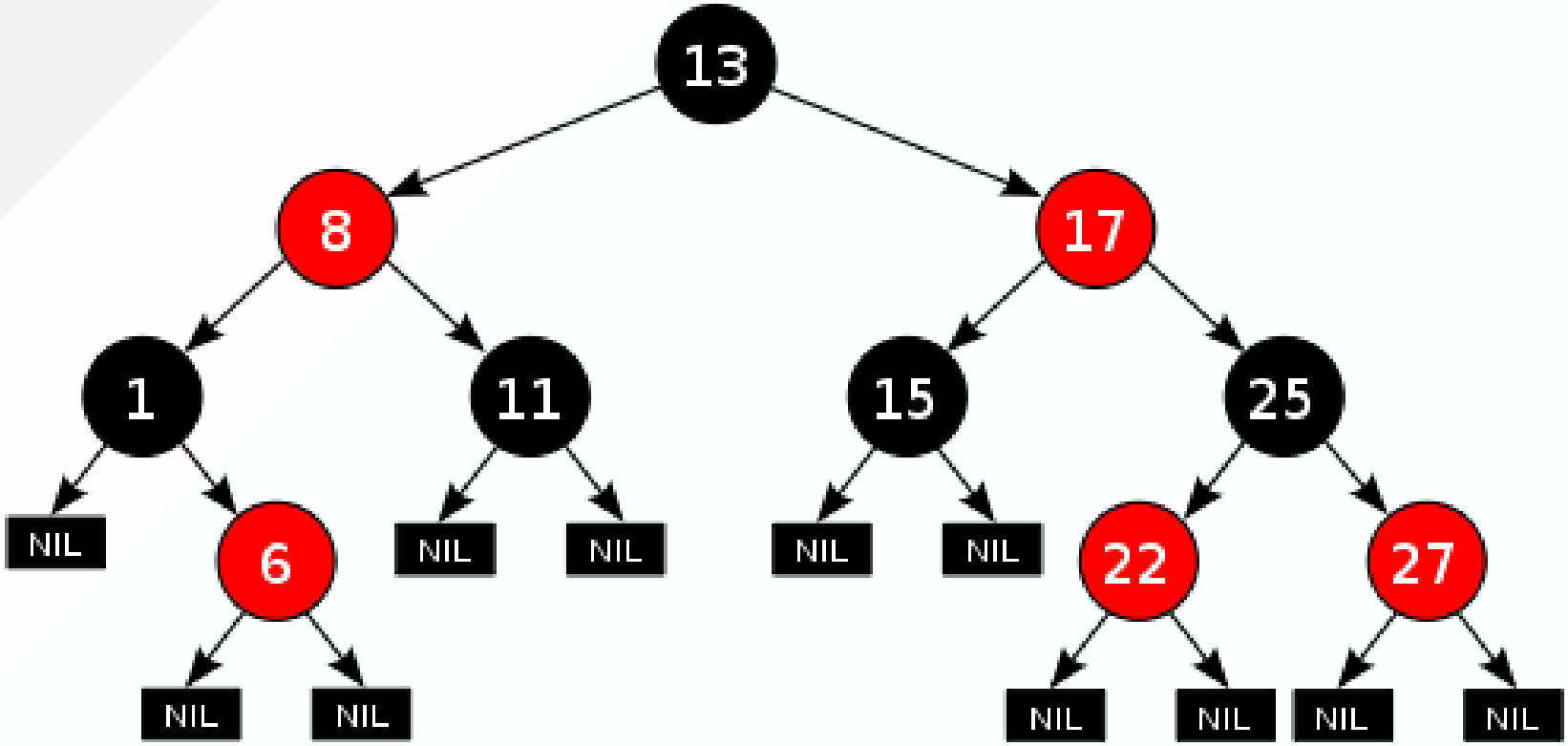
Universidad del Valle
Sede Tuluá

PROPIEDADES

- Todo nodo es rojo o negro.
- Toda hoja (*nil*) es negra.
- La raíz es negra.
- Si un nodo es rojo, entonces sus hijos son negros
- Cada ruta simple de un nodo a un hoja descendente contiene el mismo número de nodos negros.

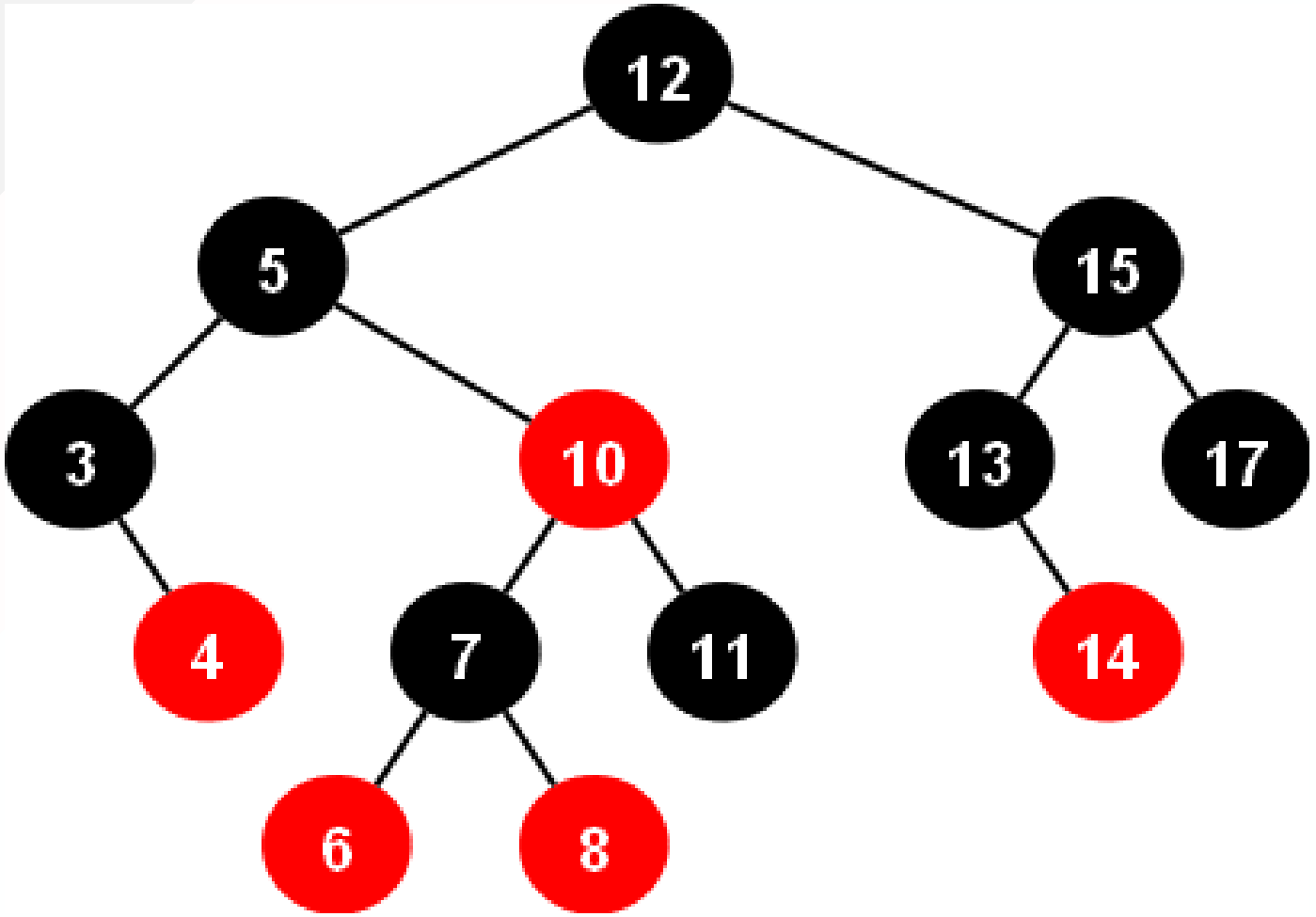
Universidad del Valle
Sede Tuluá

EJEMPLO



Universidad del Valle
Sede Tuluá

EJEMPLO



Universidad del Valle
Sede Tuluá

ALTURA NEGRA

- Se denomina ***altura negra*** del nodo x, ***bh(x)***, al número de nodos negros en cualquier camino desde el nodo x (sin incluirlo) hasta una hoja.

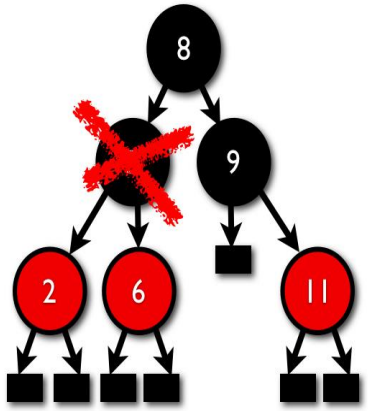
- Un árbol rojinegro con n nodo internos tiene una altura de
  $2 \log (n + 1)$, como máximo.

- Las operaciones Search, Minimum, Maximum, Successor y Predecessor pueden ser implementadas en un tiempo O(log n) en árboles rojinegros

ROTACIONES SIMPLES

- Hacer inserciones o eliminaciones sobre un árbol rojinegro podría hacer violar sus propiedades.

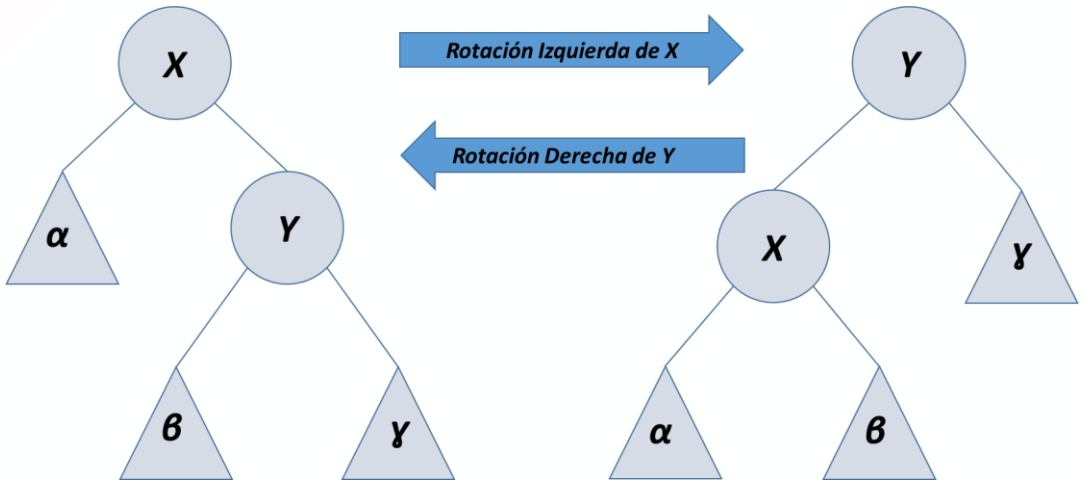
- Qué pasa si hacemos esta eliminación?



Universidad del Valle
Sede Tuluá

ROTACIONES SIMPLES

- En estos casos se hace necesario “rotar” uno o varios nodos del árbol con el fin de preservar las propiedades de los rojinegros.
- Las rotaciones se pueden visualizar así:

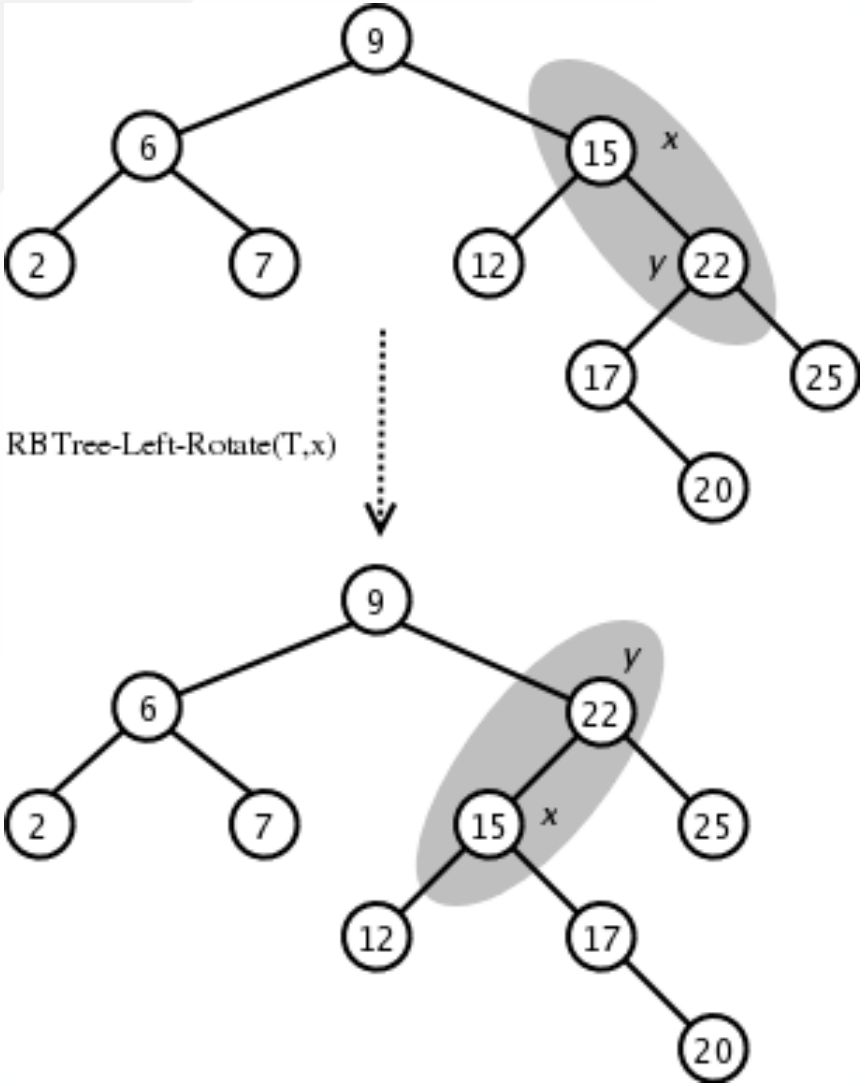


ROTACIONES SIMPLES

- RBTREE-LEFT-ROTATE (T,x)

```
1    y ← right[x]
2    right[x] ← left[y]
3    p[left[y]] ← x
4    p[y] ← p[x]
5   if p[x]= NIL
6        then root[T] ← y
7        else if x=left[p[x]]
8                  then left[p[x]] ← y
9                  else right[p[x]] ← y
10    left[y] ← x
11    p[x] ← y
```

ROTACIONES SIMPLES



El algoritmo para rotar a la derecha es similar

Ambos algoritmos corren en tiempo *O(1)*

Universidad del Valle
Sede Tuluá

INSERCIÓN

- Primero, Se inserta el nodo usando la misma técnica para inserción en un Árbol de Búsqueda Binaria

- Segundo, Se colorea de rojo el nodo insertado.

- Tercero, Se modifica el árbol recoloreando nodos y haciendo rotaciones para garantizar las propiedades de los rojinegros.

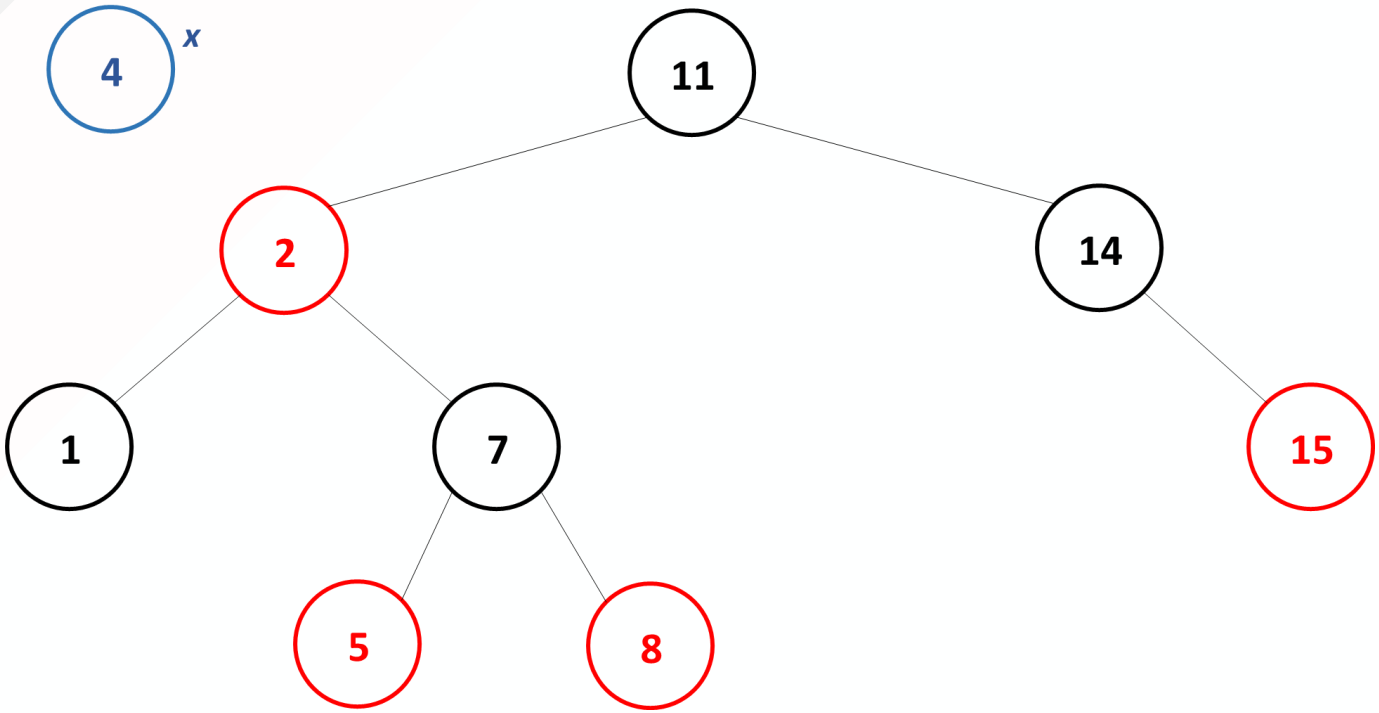
INSERCIÓN

```
RB-INSERT(T, z)
1.  y ←NIL
2.  x ← root[T]
3.  while(x ≠ NIL)
4.      do  y ← x
5.    if(key[z] < key[x])
6.              then x ← left[x]
7.              else  x ← right[x]
8.   p[z] = y
9.   if( y == NIL)
10.   then  root[T] ← z
11.   else if (key[z] < key[y])
12.            then left[y] ← z
13.            else right[y] ← z
14. right[z] ←NIL
15. left[z] ←NIL
16. color[z] ←"RED"
17. RB-INSERT-FIXUP(T, z)
```

```
RB-INSERT-FIXUP(T,z)
1. while color[p[z]] = RED
2.       do if p[z] = left[p[p[z]]]
3.             then y ← right[p[p[z]]]
4.                   if color[y] = RED
5.                      then color[p[z]] ← BLACK                    case 1
6.                              color[y] ← BLACK                                case 1
7.                              color[p[p[z]]] ← RED                            case 1
8.                               z ← p[p[z]]                        case 1
9.                      else if z = right[p[z]]
10.                             then  z ← p[z]                                  case 2
11.                                    LEFT-ROTATE(T , z)           case 2
12.                                    color[p[z]] ← BLACK          case 3
13.                                    color[p[p[z]]] ← RED         case 3
14.                                    RIGHT-ROTATE(T , p[p[z]])    case 3
15.            else (same as then clause with "right" and "left" exchanged)
16. color[root[T]] ← BLACK
```

EJEMPLO D INSERCIÓN

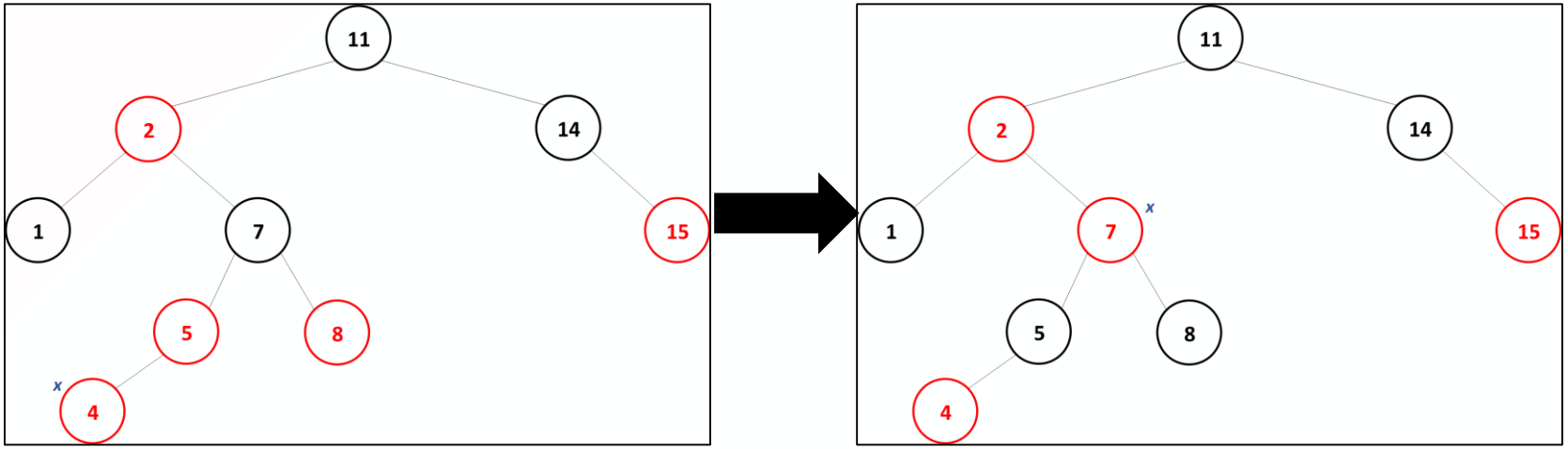
- Insertar el nodo con Key=***4*** en el siguiente árbol:



CASOS DE INSERCIÓN

- ***CASO 1:***

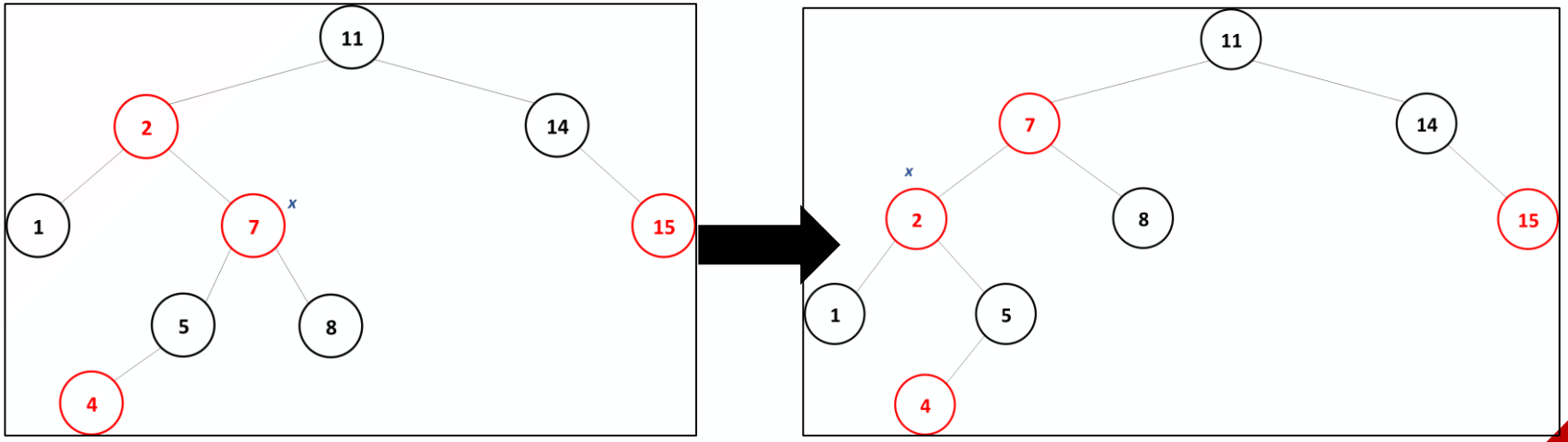
  Sea *x* el nodo rojo con padre rojo, si el tío de *x* es rojo, se colorean de negro su padre y su tío, y se colorea de rojo a su abuelo.



CASOS DE INSERCIÓN

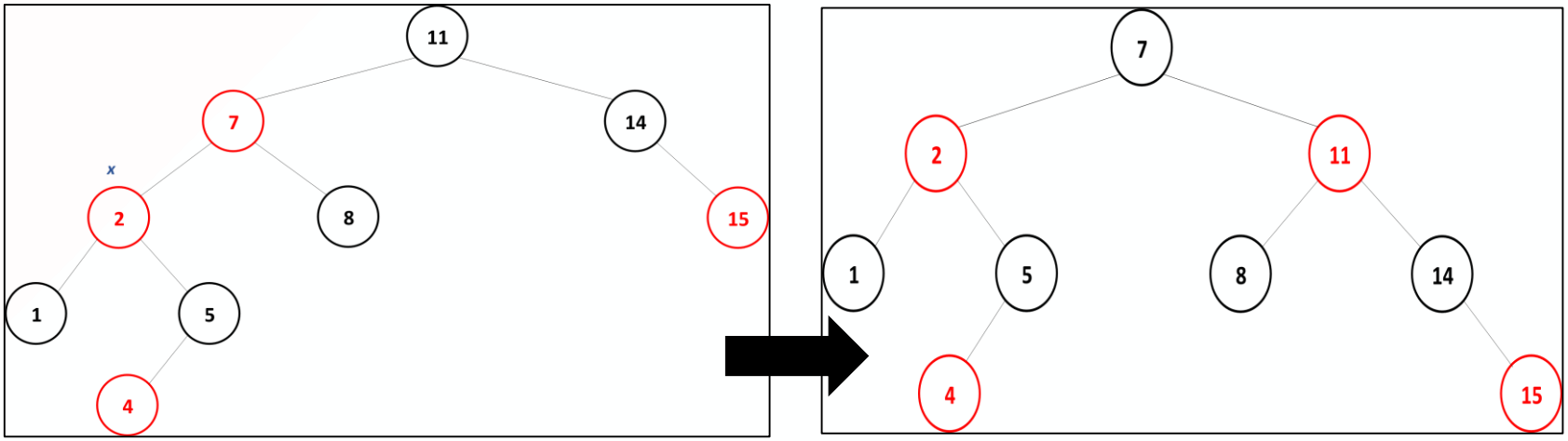
- ***CASO 2:***

  Sea *x* el nodo rojo con padre rojo, si el tío de *x* es negro y *x* es un hijo derecho, entonces se hace una rotación izquierda del padre de *x*.



CASOS DE INSERCIÓN

- ***CASO 3:*** Sea *x* el nodo rojo con padre rojo, si el tío de *x* es negro y *x* es un hijo izquierdo, entonces se colorea de negro a su padre, se colorea de rojo a su abuelo y se hace una rotación derecha del abuelo de *x*.



CASOS DE INSERCIÓN

- NOTA:

  1. Estos casos aplican cuando el padre de *x* es un hijo izquierdo.

  2. En el caso en que el padre de *x* sea un hijo derecho, se aplican los mismos casos **intercambiando** ***derecha*** e ***izquierda***.

ELIMINACIÓN

- RBTREE-DELETE (T,z)

```
1   if  left[z]= NIL or right[z]= NIL
2         then  y ← z
3         else   y ←  TREE-SUCCESSOR (z)
4   if  left[y]≠ NIL
5         then  x ←  left[y]
6         else   x ←  right[y]
7    p[x] ←  p[y]
8   if  p[y]= NIL
9         then  root[t] ←  x
10          else  if  y=left[p[y]]
11                      then  left[p[y]] ←  x
12                      else   right[p[y]] ←  x
13    if  y≠z
14          then  key[z] ←  key[y]
15    if  color[y]=BLACK
16          then  RBTREE-DELETE-FIXUP (T,x)
17    return  y
```

ELIMINACIÓN

```
RBTREE-DELETE-FIXUP (T,x)
1   while  x≠root[T] and color[x]=BLACK
2           do if  x=left[p[x]]
3                  then  w ← right[p[x]]
4                        if  color[w]=RED
5                            then  color[w] ← BLACK
6                                  color[p[x]] ← RED
7                                  RBTREE-LEFT-ROTATE (T,p[x])
8                                  w ← right[p[x]]
9                        if  color[left[w]]=BLACK and color[right[w]]=BLACK
10                           then  color[w] ← RED
11                                 x ← p[x]
12                           else  if  color[right[w]]=BLACK
13                                     then  color[left[w]] ← BLACK
14                                           color[w] ← RED
15                                           RBTREE-RIGHT-ROTATE (T,w)
16                                              w ← right[p[x]]
17                                 color[w] ← color[p[x]]
18                                 color[p[x]] ← BLACK
19                                 color[right[w]] ← BLACK
20                                 RBTREE-LEFT-ROTATE (T,p[x])
21                                 x ← root[T]
22                  else  Mirror opposite of "then" clause
23  color[x] ← BLACK
```

Caso 1

Caso 2

Caso 3

Caso 4

Universidad del Valle
Sede Tuluá

Preguntas