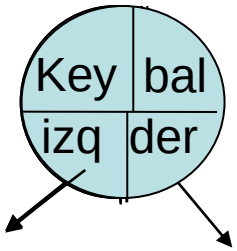
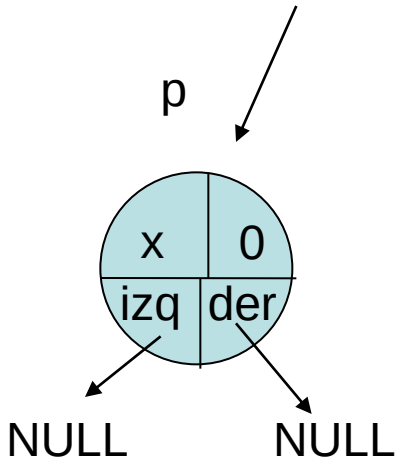
Arbol Balanceado

```
struct nodo
{  int key;
   int bal;
   struct nodo *izq,*der;
};
typedef struct nodo *punt;
```



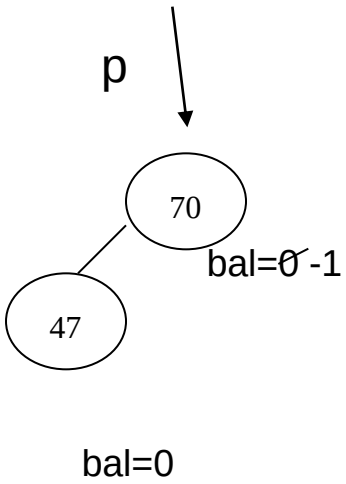
Arbol Balanceado

```
void insertar(punt &p,int x,int &h)
{ punt p1,p2;
if(p==NULL)/*insertar*/
{ p=(nodo*)malloc(sizeof(nodo));
  h=1;
  p->key=x;
  p->izq=NULL;
  p->der=NULL;
  p->bal=0;
}  else  ….
```



Arbol Balanceado

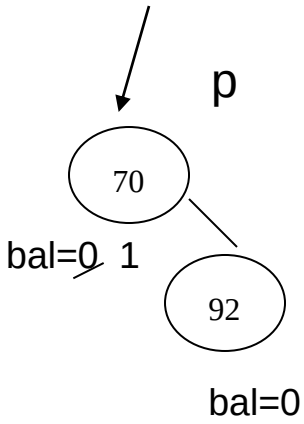
```
…….
if(p->key>x)
 { insertar(p->izq,x,h);
   if(h)/*La rama izquierda creció*/
   switch(p->bal)
{      case 1:{p->bal=0;
                  h=0;
                  break;
                 }
       case 0:{ p->bal=-1;
                  break; }
       case-1:{/*Rebalancear*/
                 p1=p->izq;
                 if(p1->bal==-1)/*RSI
                  { p->izq=p1->der;
                    p1->der=p;
                    p->bal=0;
                    p=p1; }
```




```
else/*Rotacion doble lado derecho*/
                 {p2=p1->der;
                  p1->der=p2->izq;
                  p2->izq=p1;
                  p->izq=p2->der;
                  p2->der=p;
                  if(p2->bal==-1)
                    p->bal=1;
                  else
                    p->bal=0;
                  if(p2->bal==1)
                    p1->bal=-1;
                  else
                    p1->bal=0;
                  p=p2;
                  }
                 p->bal=0;
                 h=0;
                }/*fin Case*/
  }/*fin switch*/
 }/*fin if*/
else ….
```

Arbol Balanceado

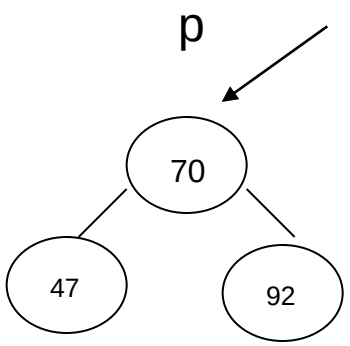
```
else
 if(p->key<x)
  { insertar(p->der,x,h);
    if(h)/*La rama derecha crecio*/
     switch(p->bal)
     {case -1:{p->bal=0;
                 h=0;
                 break;}
      case 0:{p->bal=1;
                 break;}
      case 1:{/*Rebalancear*/
                 p1=p->der;
                 if(p1->bal==1)/*RSLD*/
                  {p->der=p1->izq;
                   p1->izq=p;
                   p->bal=0;
                   p=p1;}
```




```
else/*Rotacion doble lado izq*/
  {p2=p1->izq;
   p1->izq=p2->der;
   p2->der=p1;
   p->der=p2->izq;
   p2->izq=p;
   if(p2->bal==1)
    p->bal=-1;
   else
     p->bal=0;
   if(p2->bal==-1)
     p1->bal=1;
   else
    p1->bal=0;
   p=p2; }
 p->bal=0;
 h=0;
}/*fin case*/
}/*fin switch*/
}/*fin if*/
```

Arbol Balanceado

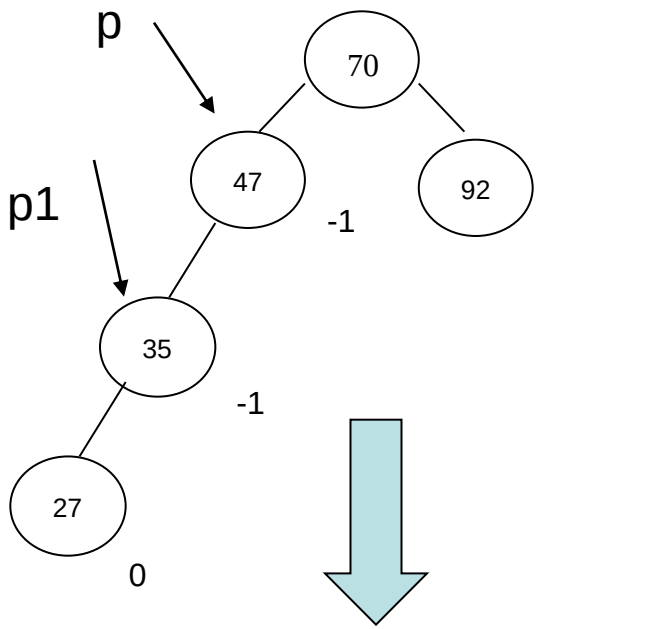
```
if(p->key>x)

    case 1:{p->bal=0;
                    h=0;
                  break;
                    }
```

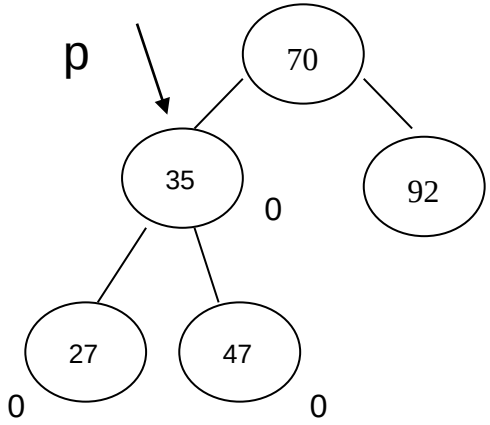



```
if(p->key<x)

    case -1:{p->bal=0;
                    h=0;
                  break;}
```

Arbol Balanceado

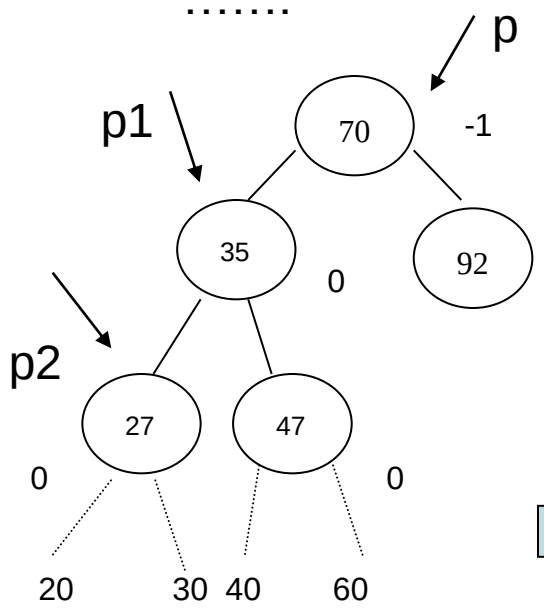
```
case-1:{/*Rebalancear*/
          p1=p->izq;
          if(p1->bal==-1)/*RSI
            { p->izq=p1->der;
              p1->der=p;
              p->bal=0;
              p=p1; }

          ……
            p->bal=0;
                 h=0;
```

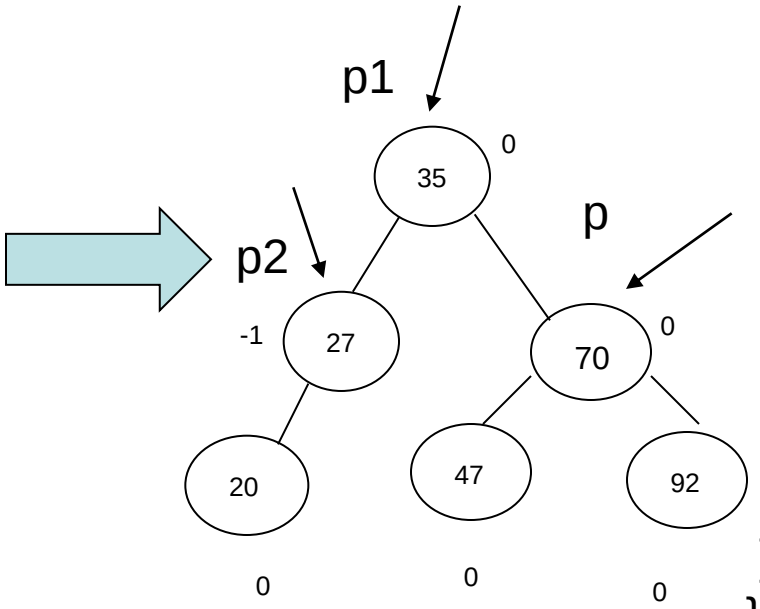




Arbol Balanceado

```
case-1:{/*Rebalancear*/
        p1=p->izq;
        if(p1->bal==-1)/*RSI
          …….
```

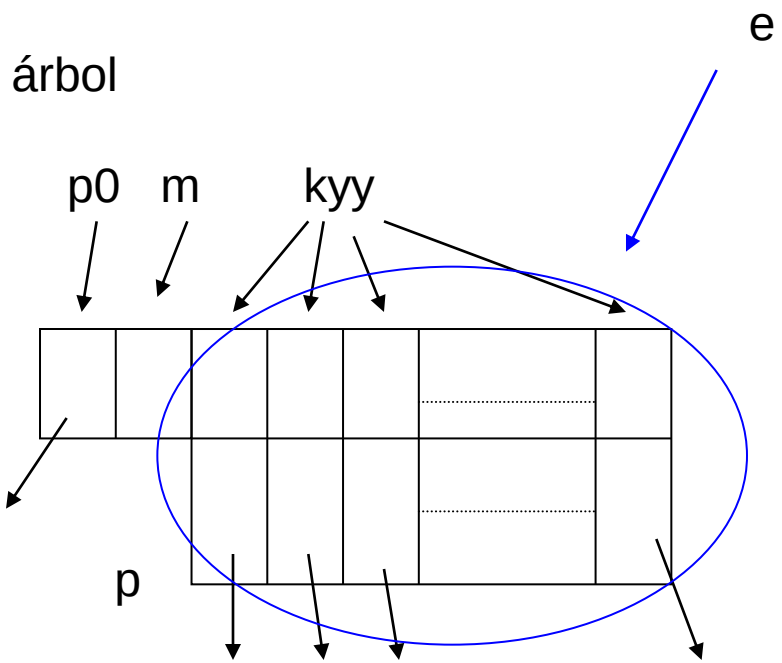





```
else/*Rotacion doble */
 {p2=p1->der;
  p1->der=p2->izq;
  p2->izq=p1;
  p->izq=p2->der;
  p2->der=p;
  if(p2->bal==-1)
      p->bal=1;
  else
      p->bal=0;
  if(p2->bal==1)
      p1->bal=-1;
  else
      p1->bal=0;
  p=p2;
  }
 p->bal=0;
 h=0;
 }/*fin Case*/
 }/*fin switch*/
 }/*fin if*/
else ….
```

Arbol B

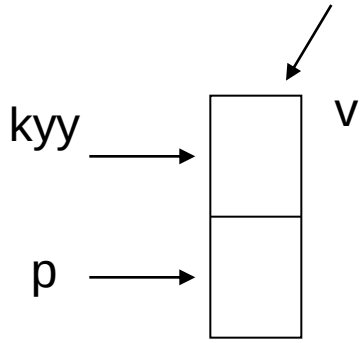
```
struct item
                {
                 int kyy;
                 struct page *p;
                 };
struct page{int m;
                  struct page *p0;
                  item e[2*n];  //n orden del árbol
                  };
typedef struct page *puntero;
```



Arbol B

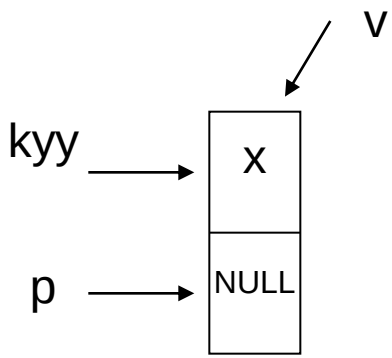
```
 void main (void)
 { puntero raiz,q;
   item v;
   int op,x,h;
  raiz=NULL;
   ……….
printf("\n Ingrese clave a insertar(Finaliza con -1) ");
scanf("%d",&x);
while(x >=0)
 {
   insertar(x,raiz,h,v);
   if (h)
     {
       q=raiz;
       raiz=new(page);
       raiz->m=1;
       raiz->p0=q;
       raiz->e[0]=v;
     }
   printf("\n Ingrese clave a insertar (Finaliza con -1) ");
   scanf("%d",&x);
 }
```

raiz → NULL



**

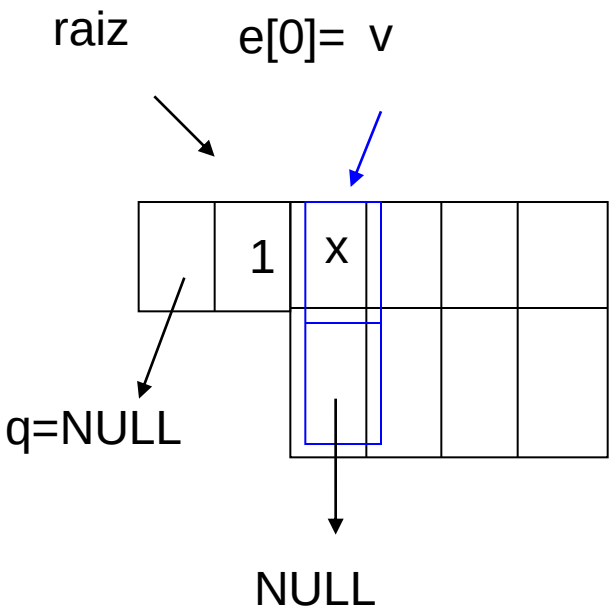
```
void insertar(int x,puntero &a,int &h,item &v)
{int l,i,r;puntero b;item u;
if(a==NULL)   //carga el registro con la clave
{
 h=1;
 v.kyy=x;
 v.p=NULL;
 }
 else  ……
```

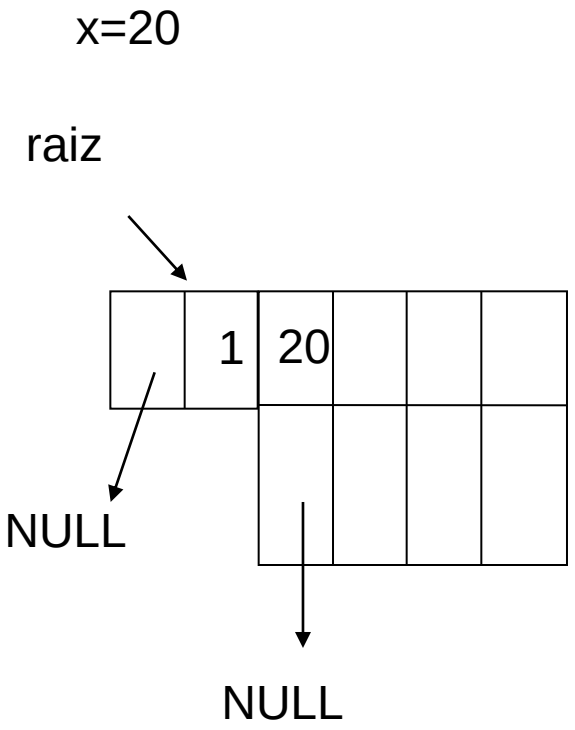


Arbol B

**

q=raiz (NULL)

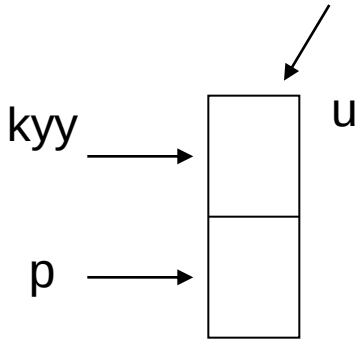




Arbol B

```
printf("\n Ingrese clave a insertar(Finaliza con -1) ");
 scanf("%d",&x);
 while(x >=0)
  {
    insertar(x,raiz,h,v);
```

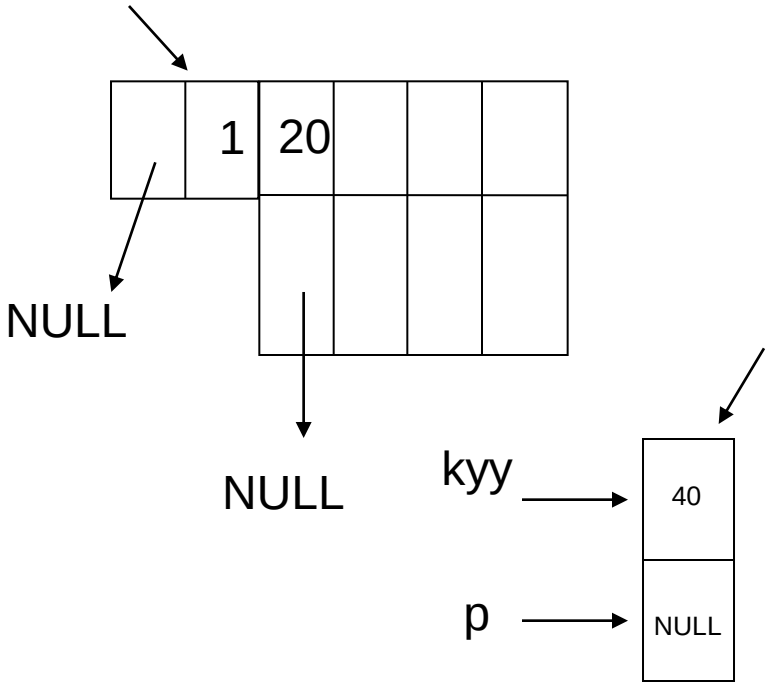
X=40




```
void insertar(int x,puntero &a,int &h,item &v)
{int l,i,r;puntero b;item u;
if(a==NULL)   //carga el registro con la clave
```

```
 else  // Busca donde insertar una nueva clave
 {l=1;   r=(a->m);
  while (l<r)
   {i=(l+r)/2;
    if((a->e[i].kyy<=x))
       l=i+1;
    else
       r=i;
    }
   r--;  ........
```

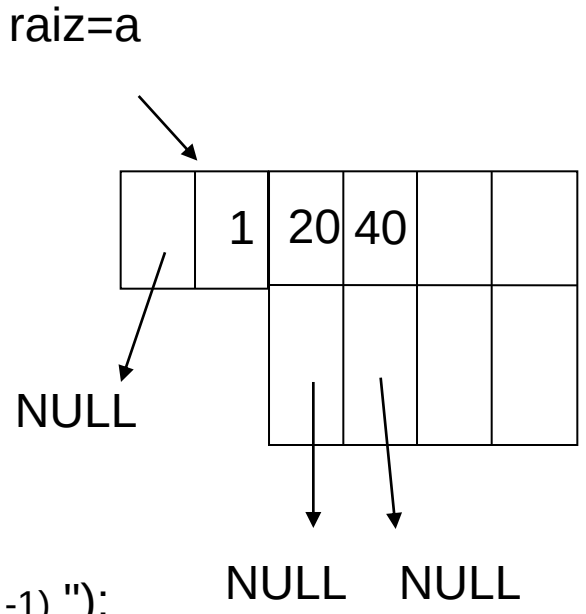
raiz=a




```
if ((r==0)&&(a->e[r].kyy>x)) // los mas chicos
    insertar(x,a->p0,h,u);
else
    insertar(x,a->e[r].p,h,u); // los mayores que...
```

Arbol B

```
if (h)
 if(a->m<2*n)  {hay lugar}
   {h=0;
    a->m++;
    for(int i=a->m-1;i>=(r+1);i--) // corrimiento para insertar
       a->e[i]=a->e[i-1];
    if (a->e[r].kyy>x)// se ubica a el nuevo elemento
       a->e[r]=u;
    else
       a->e[r+1]=u;          }
```




```
printf("\n Ingrese clave a insertar(Finaliza con -1) ");
scanf("%d",&x);
while(x >=0)
 {
   insertar(x,raiz,h,v);
   if (h)
    {
      q=raiz;
      raiz=new(page);
      raiz->m=1;
      raiz->p0=q;
      raiz->e[0]=v;
```

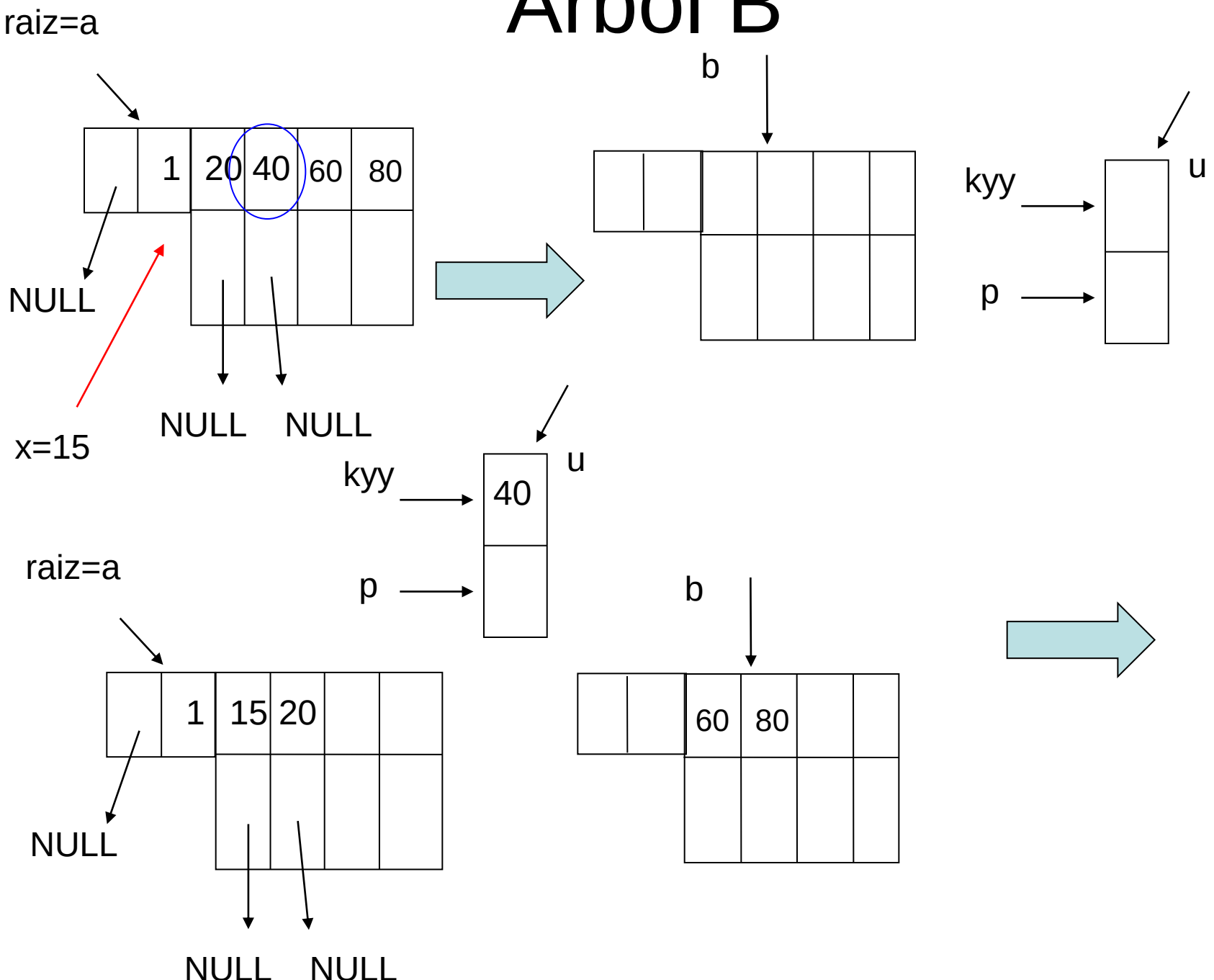
Arbol B

```
 else
{//no hay lugar pedir una nueva página
    b=new(page);
    if(r<=n)
     {  for(int i=0;i<n;i++)
              b->e[i]=a->e[i+n];
       if(r==n)
       {  v=a->e[n];
          b->e[0]=u;}
       else
       { if (a->e[n-1].kyy < x)
            v=u;
         else
          { v=a->e[n-1];
            if (a->e[0].kyy < x)
                     a->e[n-1]=u;
            else
             {  for(int i=n-1;i>=(r+1);i--)
                       a->e[i]=a->e[i-1];
                a->e[r]=u;  }
            }
         }
       }
```

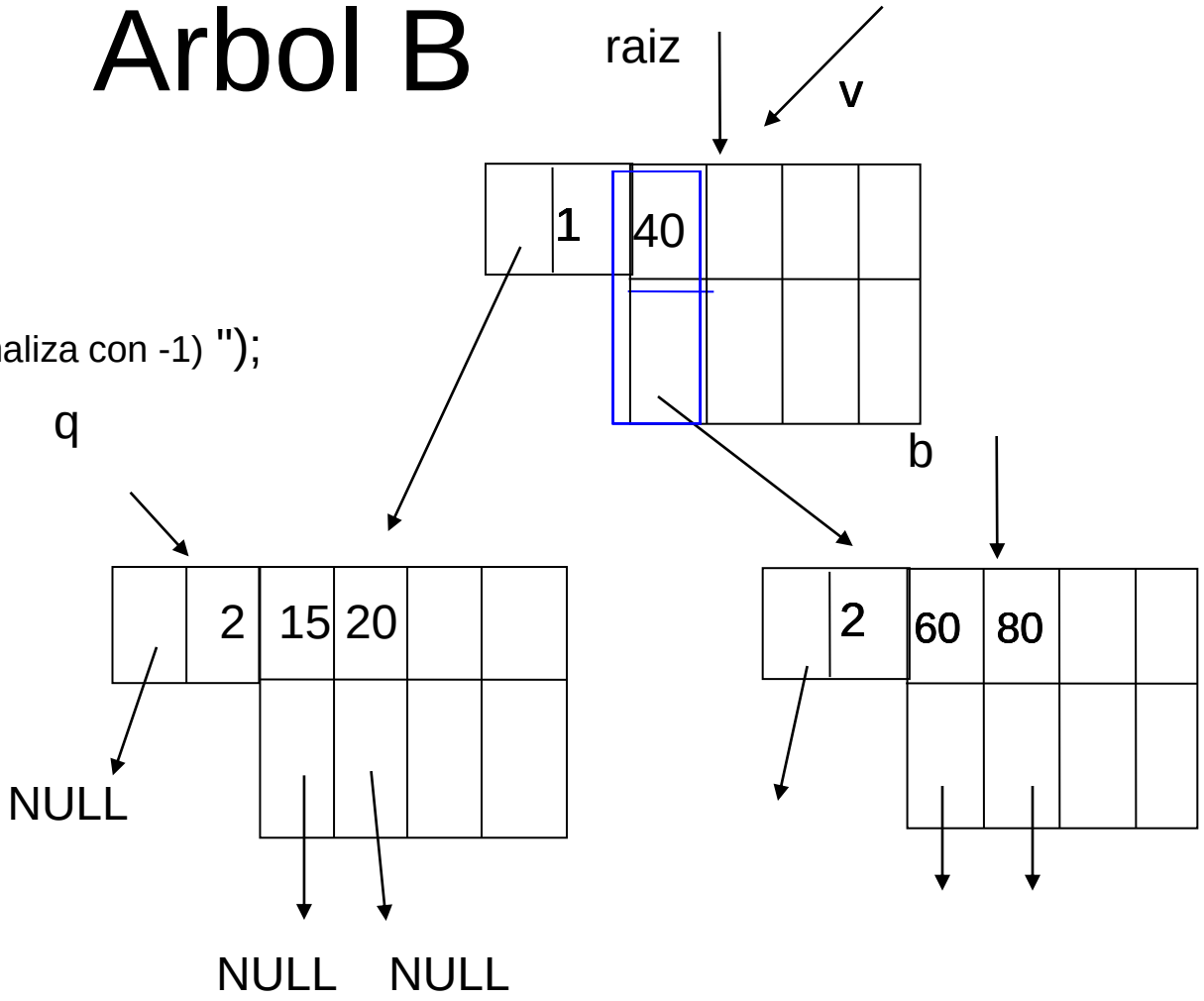
```
else
       {r=r-n;
        v=a->e[n];
        for(int i=0;i<r;i++)
        b->e[i]=a->e[i+n+1];
        b->e[r]=u;
        for(int i=r+1;i<=n;i++)
            b->e[i]=a->e[i+n];
        }
     a->m=n; // se especifican los valores de
m para las páginas
     b->m=n;
     b->p0=v.p;// y los punteros
correspondientes
     v.p=b;
```

x=15

Arbol B


raiz=a
1
20
40
60
80
NULL
NULL
NULL
x=15
b
kyy
u
p
40
raiz=a
1
15
20
NULL
NULL
NULL
b
60
80

Arbol B




```
printf("\n Ingrese clave a insertar(Finaliza con -1) ");
scanf("%d",&x);
while(x >=0)
 {
   insertar(x,raiz,h,v);
   if (h)
     {
       q=raiz;
       raiz=new(page);
       raiz->m=1;
       raiz->p0=q;
       raiz->e[0]=v;
      }
   printf("\n Ingrese clave a insertar (Finaliza con -1) ");
   scanf("%d",&x);
 }
```