```
// Function to decrease the value of old_val
// to new_val
void decreasekeyBHeap (Node *H, int old_val,
                                  int new_val)
{
 // First check element present or not
   Node *node = findNode (H, old_val);

 // return if Node is not present
 if (node == NULL)
 return;

 // Reduce the value to the minimum
 node -> val = new_val;
Node *parent = node -> parent;

// Update the heap according to reduced value
while (parent ! = NULL && Node -> val < parent -> val)
 {
   swap (node -> val, parent -> val);
   node = parent;
  parent = parent -> parent;
   }
 }.
```

```
// Function to delete an element
Node *binomialHeapDelete(Node *h, int val)
{
   //Check if heap is empty or not
  if (h == NULL)
     return NULL;

  //Reduce the value of element to minimum
   decreaseKeyBHeap(h, val, INT_MIN);

  // Delete the minimum element from heap
    return extractMinBHeap(h);
 }
```