8.

(a)
```
void Leftist::Merge(Leftist &rhs)
{
   if(this == &rhs)
      return;
   root = Merge(root, rhs.root);
   rhs.root = NULL;
}

LeftistNode *Leftist::Merge(LeftistNode*h1, LeftistNode*h2)
{
   if(h1==NULL) return h2;
   if(h2==NULL) return h1;
   if(h1->element < h2->element) return Merge(h1,h2);
   else return Merge(h2,h1);
}
```

(b)
```
void LeftList::reclaimMemory(LeftListNode *t)
{
    if(t!NULL)
    {
        reclaimMemory(t->left);
        reclaimMemory(t->right);
        delete t;
    }
}

LeftListNode *LeftList::clone(LeftListNode *t)
{
    if(t==NULL) return NULL;
    else
        return new LeftListNode(t->element, clone(t->left), clone(t->right),
                                t->dist);
```

(c) O(1). O(log n).