p.183   5.29

Merge the two binomial queues in Figure 5.59.

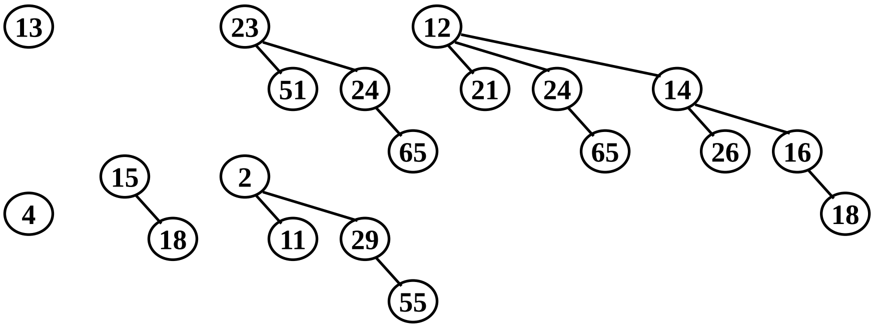
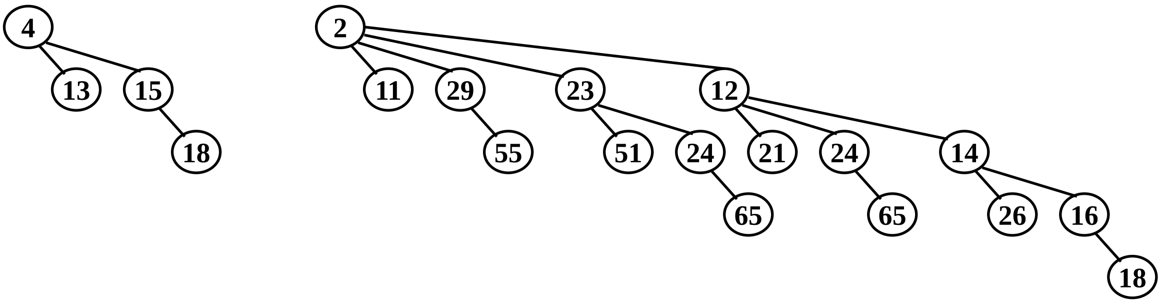


Figure 5.59

Solution:



p.183   5.31

Write an efficient routine to perform *Insert* using binomial queues. Do not call *Merge*.

```
BinQueue   Insert( ElementType   X,   BinQueue   H )
{
     /* insert X into H with a simplified version of Merge */
     BinTree    Carry;    /* the tree carried from the previous step */
     int    i;    /* index of the current tree in H */

     if ( H->CurrentSize + 1 > Capacity )
          Error( "Insertion would exceed capacity" );
     H->CurrentSize ++;    /* update the size of H */

     /* initialize Carry to be a single-node tree */
     Carry = malloc( sizeof( struct BinNode ) );
     if ( !Carry )
```

```
        FatalError( “Out of Space!!!” );

    else {    /* begin insertion */
        /* Initialize Carry to contain X */
        Carry->Element = X;
        Carry->LeftChild = Carry->NextSibling = NULL;

        i = 0; /* start from the first tree of H */
        while ( H->TheTrees[ i ] ) { /* if B_i exists */
            /* merge Carry with B_i, and carry the result to the next step */
            Carry = CombineTrees( Carry, H->TheTrees[ i ] );
            /* reset this B_i and continue to the next tree*/
            H->TheTrees[ i ++ ] = NULL;
        }    /* end – while */
        H->TheTrees[ i ] = Carry; /* find the first nonexistent B_i and insert Carry */
    }    /* end – else */
    return    H;
}
```