Ex2

public static int sumAll(BinarySearchTree<Integer> b)

 {

 int sum =0;

 while(!b.isEmpty())

 {

 sum+= b.minKey();

 b.delete(b.minKey());

 }

 return sum;

 }

 Ex3

 public static int sumGT20(BinarySearchTree<Integer> b)

 {

 int sum =0;

 while(!b.isEmpty())

 {

 if(b.minKey()>20)

 {

 sum+= b.minKey();

 }

 b.delete(b.minKey());

 }

 return sum;

 }

 Ex4

 public static int min(BinarySearchTree<Integer> b)

 {

 return b.minKey();

 }

 Ex5

 public static int Maxsum(BinarySearchTree<Integer> b1, BinarySearchTree<Integer> b2)

 {

 int sum1 =0;

 int sum2 =0;

 while(!b1.isEmpty())

 {

 sum1+= b1.minKey();

 b1.delete(b1.minKey());

 }

 while(!b2.isEmpty())

 {

 sum2+= b2.minKey();

 b2.delete(b2.minKey());

 }

 if(sum1>sum2)

 return sum1;

 else

 return sum2;

 }

 Ex7:

 public static int maxItemlength(BinarySearchTree<String> b)

 {

 int max = 0;

 while(!b.isEmpty())

 {

 if(b.minKey().length()>max)

 {

 max= b.minKey().length();

 }

 b.delete(b.minKey());

 }

 return max;

 }