**Set<T> S(V,n);**

{   
 init(); //O(1)  
 for(int i = 0; i < n; i++){ //O(n)  
 insert(temp, a[i]); //O(1)  
 }

Svar: T(n) = O(1) + O(n) + O(1) = O(n)

**S1=S2**

{  
 this->clear(); //O(n)   
 while(temp != b.tail){ //O(n)  
 insert(this->tail, temp->value);//O(1)  
 temp = temp->next;

}  
 return \*this;  
}

Svar: T(n) = O(n) + O(1) + O(n) = O(n)

**S.cardinality();**

{   
 while(temp != this->tail){ //O(n)  
 counter++;  
 temp = temp->next;  
 }  
 return counter;  
}

Svar: T(n) = O(n)

**k+S;**

while(thisTemp != this->tail || bTemp != b.tail){ //O(n)

if(thisTemp == this->tail){ //O(1)  
 while (bTemp != b.tail) { //O(n)

newSet.insert(newSet.tail, bTemp->value); //O(1)

bTemp = bTemp->next;

}

break;

}

}

Svar: T(n) = O(n) + O(n) = O(n)

**S1<=S2;**

{

while(thisTemp != this->tail || bTemp != b.tail){ //O(n)

if(thisTemp == this->tail){ //O(1)

break;

}

if(bTemp == b.tail){ //O(1)

return false;

}

if(thisTemp->value > bTemp->value ){ //O(1)

continue;

}

else if(thisTemp->value < bTemp->value){

return false;  
 }

else if(thisTemp->value == bTemp->value){

continue;

}

}

return true;

}

Svar: T(n) = O(5n) = O(n)

**S1+S2;**

while(thisTemp != this->tail || bTemp != b.tail){ //O(n)

if(thisTemp == this->tail){ //O(1)

while (bTemp != b.tail) { //O(n)

newSet.insert(newSet.tail, bTemp->value);

}

break;

}

else if(bTemp == b.tail){ //O(1)

while (thisTemp != this->tail){ //O(n)

newSet.insert(newSet.tail, thisTemp->value);//O(1)

}

break;

}

else if(thisTemp->value > bTemp->value ){ //O(1)

newSet.insert(newSet.tail, bTemp->value); //O(1)

continue;

}

else if(thisTemp->value < bTemp->value){ //O(1)

newSet.insert(newSet.tail, thisTemp->value); //O(1)

continue;

}

else if(thisTemp->value == bTemp->value){ //O(1)

thisTemp = thisTemp->next;

continue;

}

}

return newSet;

}

Svar: T(n) = O(5n) + O(4n) = O(n)