CSE 222 HW03 REPORT

131044042_AhmetEmin_Kaplan

Ödev pdf inde belirtilen metodlari implement etmeden önce metodlarin kendi içinde karsilastirma metodu olan compareTo() 'ya ihtiyaç duyacağını düşündüm.Primitive tipler haricindeki sınıfların birbirleriyle karşılaştırılması için Comparable interface ini implement etmiş olması gerektiğini düşündüm ve bu kullanımı sağlamak için sınıfın generic olma durumunu Comparable olmakla sınırlandırdım. Veriler üzerinde gezinmenin iterator yapısını kullanarak daha performanslı olacağını düşündüm ondan dolayı metodlari implement ederken iterator ve listIterator yapılarını kullandım.

public boolean addAllAtHead(Collection<? extends E> c)

-Bu metodu implement ederken parametre olarak aldigim c değişkeninin tipi collection olduğu için bütün collectionlarin ortak özelliği olan iterator yapısıyla parametre değişkenimin üzerinde gezerek her bir elemani tek tek SpecList objesinin ilk elemanina ekledim.SpecList objesinin iteratorunun add metodu constant zaman aldigi için bu metodun performansini c parametresinin boyutu cinsinden lineer zaman olarak ifade edebiliriz.C değişkeninin boyutu n ise best case ile worst case ayni olacagi icin Theta(n) ile ifade edebiliriz

public List<E> getIntersectList(Collection<? extends E> c)

-Bu metodu implement ederken yine iterator yardimiyla c değişkeninin butun yapilari üzerinde gezmek Lineer time alacaktir .C değişkeninin boyutu n ise en dıştaki döngü n kere dönecektir.

Yani parametre olarak aldigim c değişkeninin her bir elemanini teker teker K uzunluğundaki SpecList objemin içinde aradigim için 2.döngünün dönme miktari K olacaktır.Ortak elemanlari eklediğim result listesinde duplication olmamasi için Ortak elemani eklemeden önceki kontrol T boyutlu result listesi için 3.döngü T kez dönecektir.

Kabaca metodun performansi O(N *K*T) Best case durumu için 3.döngünün içindeki break mekanizmasi Theta olarak ifade etmemizi engellemektedir.

public List<E> sortList (boolean mode)

-Bu metod çalışma mantığı itibariyle otomobillerin hız göstergelerine benziyor iteratorler sağa doğru giderken maximum elemani sağa taşıyorlar , sola doğru giderken minimum elemani sola taşıyorlar veya tam tersi formatta da çalışabilir.

En dıştaki döngü size/2 defa dönecektir n elemanlı bir listede n/2 defa dönecektir.İçteki döngüler ise içteki 2 döngünün dönme miktarı ise her bir iterasyonda azalmaktadır olmakla beraber ilk iterasyonlarinda size-2 defadır.Genel bir takımla metodun O(n^2) performansında çalışacağını ifade edebiliriz.Best case n defa döneceği için performans Omega(n) ile ifade edilebilir.

*Bu ödev sayesinde Iterator kullanımını pekiştirdiğimi düşünüyorum.

Yazım hatalarım için özür dilerim.