Fonksiyonların asimptotik gösterimleri

1. addAllAtHead

```
for(E itr : c) {
    addFirst(itr);
}
```

→ iterator yardımıyla tüm elemanlar gezilecek ve bu da n(lineer) zamanda olur.

↓ farklı gösterim

```
for(ListIterator<E> itr = c.listIterator();
        itr.hasNext();
        itr.next(); )
{
    addFirst(itr);
}
```

→ addFirst daima başa eklediği için constant zamanda gerçekleşir.

→ $\sum_{i=0}^{i=size-1} 1 = \sum_{i=1}^{i=n} 1 = \underbrace{n}_{f(n)}$ diyebiliriz.

$f(n) = n$

$g(n) = n$

$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{n}{n} \overset{\text{L-hospital}}{=} 1 = c \quad 0 < c < \infty$ ✓

→ Her koşulda c listini sonuna kadar tarayacak. Worst ve best case n 'dir. limit aralığıda sağladığına göre

$$f(n) = \Theta(g(n))$$

$$n = \Theta(n)$$

* Method testleri (Junit4) ve javadoc yazıldı ✓

2. getIntersectList (c)

```
while ( itr.hasNext()){
  E node = itr.next();
  if ( c.contains(node)){
    intersectList.add(node)
  }
```

→ while döngüsü asıl (this) listemiz için lineer $\Theta(n)$ dönecek.

→ Best Case : Hemen bulacak (ilk eleman)

→ Worst Case : Son elemanda bulacak

Contains için
Best case constant, Worst case lineer $O(n)$ zamanda olur.

Toplam için $\sum_{P=1}^{P=n} n \rightarrow n^2$

$f(n) = n^2$
$g(n) = n^2$

→ $f(n)$ is in $O(g(n))$

$n^2 = c\,n^2$ $c, n \geq n_0, > 0$ ise

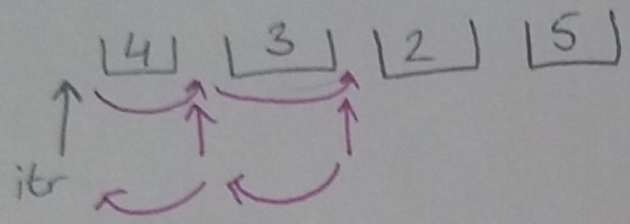
$c=1$ $n=1,2,3$ durumlarında sağlar.

→ $O(n^2)$ zamanda çalışır en kötü.

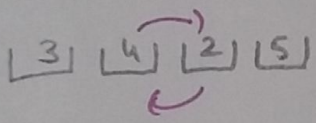
=> Junit4 and Javadoc ✓

3.

Algoritmamda list elemanlara ulaşımı hızlandırmak için iterator kullandım. get() methodu linked listte on) sürede çalışır. Iterator bu algoritma üzerinde constant sürede.



→ iterator 4'ün üzerinden 3'e sonra 3 ün arkasına geçer.
2 elemanda yedekle ve compareTo ile karşılaştırır.
2 geri 1 ileri yapa ve tekrar 4 ün üzerinden atla ve artık 4'ü değişebilir
4 değişir 1 ilerler 3'ü okur ve onuda değiştirir. Değişim tamam.
Sonra 1 geri gelerek yeni listede 4, 2 için aynısını yaparak ilerler.



=> Tüm method için

do {
while( ) } → hasnext ile lineer zamanda çalışır.
} → iterator ile constant zamanda

?
if( ) {
while() ?
} while() → lineer zamanda çalışacaktır.

En iyi durumda lineer zamanda hepsini kontrol edecektir.
En kötü durumda sıralama yapıp $O(n^2)$ zamanda çalışır.

3. devam

$$\sum_{i=1}^{n} n+n = 2n^2 = f(n)$$

$\rightarrow n^2$

$f(n) = O(g(n))$

$2n^2 \leq c\, n^2$

$c=2$ $n=1$ için $2 \leq 2$

$n=k$ için $2k^2 \leq 2k^2$

$n=k+1$ için $2(k+1)^2 \leq 2(k+1)^2$ ✓ $\angle n^2 \Rightarrow n^2 = O(n^2)$

$\Rightarrow$ 2'li Bubble sort gibi çalışacağından en kötü n^2'dir.
Bu yüzden dıştaki do-while lineer, içteki while lineer next ve prevler constant zamanda olacaktır.