Bölüm 6: Liste

Veri Yapıları

Liste Veri Yapısı

- Liste, birden çok öğeyi bir arada saklamak için kullanılan bir veri yapısıdır.
- Listeler, verilerin sıralı bir şekilde depolanmasını sağlar.

Liste Kavramları

- **Öğe (Element):** Listede depolanan her bir veri parçası.
- **İndeks (Index):** Liste içindeki her öğenin sırasını belirleyen sayısal değer. İndeks genellikle 0'dan başlar.
- **Boş Liste (Empty List):** Hiçbir öğe içermeyen bir liste.
- **Uzunluk (Length):** Listenin içinde bulunan öğe sayısı.
- **Dizi (Array):** Liste öğelerini depolamak için kullanılan veri yapısı.

Liste Özellikleri

- **Sıralıdır:** Listeler öğelerin eklenme sırasına göre sıralanır.
- **Değiştirilebilirdir (Mutable):** Öğeler eklenip çıkarılabilir, güncellenebilir.
- **İndeksleme (Indexing):** Her öğe, bir sayısal indeksle ulaşılabilir.
- **Döngülerle Kullanılabilir:** Liste öğeleri üzerinde döngülerle işlemler yapılabilir.

Liste İşlemleri

- **Ekleme (Append):** Yeni bir öğeyi listenin sonuna ekler.
- **Silme (Remove):** Belirli bir öğeyi listeden çıkarır.
- **İndeksleme (Indexing):** Belirli bir öğeye indeksle erişim sağlar.
- **Dilimleme (Slicing):** Liste içindeki bir aralığı seçme.
- **Uzunluk (Length):** Listenin öğe sayısını döndürür.

Karmaşıklık Analizi

- Ekleme: O(1)
- Silme: O(n)
- İndeksleme: O(1)
- Dilimleme: O(k)
- Uzunluk: O(1)

Java'da Liste (List) Veri Yapısı

- Liste (List), birden fazla öğeyi sıralı bir şekilde saklamak için kullanılan veri yapısıdır.
- Java'da, java.util paketi içindeki List arabirimini veya bu arabirimi uygulayan sınıfları kullanarak listeler oluşturabiliriz.

Liste Arabiriminin Ana Metodları

- Eleman Ekleme (Add)
  - **add(E e):** Liste sonuna bir eleman ekler.
  - **add(int index, E element):** Belirli bir indekse eleman ekler.
- Eleman Silme (Remove)
  - **remove(Object o):** Belirli bir elemanı listeden kaldırır.
  - **remove(int index):** Belirli bir indeksteki elemanı kaldırır.
- Eleman Erişim (Get)
  - **get(int index):** Belirli bir indeksteki elemanı döndürür.
- Liste Uzunluğu (Size)
  - **size():** Listenin uzunluğunu döndürür.
- Döngülerle Kullanım
  - Liste elemanları üzerinde döngülerle işlem yapabiliriz.

Liste Arayüzünü Uygulayan Popüler Sınıflar

- ArrayList:
- LinkedList:
- Vector:
- Stack:
- CopyOnWriteArrayList:
- Arrays.asList():

Liste Arayüzünü Uygulayan Popüler Sınıflar

- **ArrayList:**
  - İhtiyaca göre büyüyebilen veya küçülebilen dinamik dizi.
  - Öğelere hızlı rastgele erişim sağlar.
  - Sık sık ekleme veya silme gerektirmeyen senaryolar için uygundur.
- LinkedList:
- Vector:
- Stack:
- CopyOnWriteArrayList:
- Arrays.asList():

Liste Arayüzünü Uygulayan Popüler Sınıflar

- ArrayList:
- **LinkedList:**
  - Çift yönlü bağlı liste uygular, her öğe önceki ve sonraki öğelere bağlıdır.
  - Sık sık ekleme veya silme gerektiren senaryolar için uygundur.
  - Hızlı ekleme ve silme sağlar,
  - ancak ArrayList'e kıyasla rastgele erişim daha yavaştır.
- Vector:
- Stack:
- CopyOnWriteArrayList:
- Arrays.asList():

Liste Arayüzünü Uygulayan Popüler Sınıflar

- ArrayList:
- LinkedList:
- **Vector:**
  - ArrayList'e benzer, ancak senkronizedir (synchronized)
  - Çoklu iş parçacıklarında kullanıldığında güvenlidir.
  - Senkronizasyon nedeniyle performans sorunu yaşanabilir.
- Stack:
- CopyOnWriteArrayList:
- Arrays.asList():

Liste Arayüzünü Uygulayan Popüler Sınıflar

- ArrayList:
- LinkedList:
- Vector:
- **Stack:**
  - Yığın veri yapısını uygular, özel bir Liste uygulamasıdır.
  - Bir yığında kullanılan standart push ve pop işlemlerini destekler.
- CopyOnWriteArrayList:
- Arrays.asList():

Liste Arayüzünü Uygulayan Popüler Sınıflar

- ArrayList:
- LinkedList:
- Vector:
- Stack:
- **CopyOnWriteArrayList:**
  - Senkronizasyon yükü olmadan iş parçacıkları arası güvenlik sağlar.
  - Listenin sık gezildiği, nadiren değiştirildiği senaryolar için tasarlanmıştır.
  - Liste güncellendiğinde yeni bir kopya oluşturur,
  - Büyük listeler için hafıza ve performans açısından maliyetli olabilir.
- Arrays.asList():

Liste Arayüzünü Uygulayan Popüler Sınıflar

- ArrayList:
- LinkedList:
- Vector:
- Stack:
- CopyOnWriteArrayList:
- **Arrays.asList():**
  - Bir diziyi bir List'e dönüştürür.
  - Elde edilen List, sabit boyutludur ve değiştirilemez

ArrayList

- ArrayList, Java Koleksiyon Çerçevesi (Collection Framework) içinde uygulanan bir sınıftır.
- Dinamik bir dizi oluşturmak için kullanılır.
- Standart dizilere kıyasla;
  - Daha yavaş,
  - Boyutu dinamik olarak büyütülebilir.
  - Eleman eklemek veya çıkarmak kolaydır.
  - Elemanlarla daha fazla işlem yapma esnekliği sağlar.

ArrayList Kullanımı

- add(E e): Eleman ekleme
- remove(int index): Belirli bir indeksteki elemanı çıkarma
- get(int index): Belirli bir indeksteki elemana erişim
- size(): Listenin uzunluğunu alma

LinkedList

- LinkedList, Java Koleksiyon Çerçevesi'nde uygulanan bir sınıftır.
- Bağlı liste veri yapısını doğuştan uygular.
- Öğelerin ardışık konumlarda saklanmadığı bir lineer veri yapısıdır.
- Her öğe, veri kısmı ve adres kısmı olan ayrı bir nesnedir.
- Öğeler, işaretçi ve adresler kullanılarak birbirine bağlıdır.
- Her öğeye "düğüm" denir.
- Rastgele erişim performansı düşüktür, çünkü elemanlar bağlıdır ve indeksleme maliyetlidir.

LinkedList Kullanımı

- add(E e): Eleman ekleme
- remove(int index): Belirli bir indeksteki elemanı kaldırma
- get(int index): Belirli bir indeksteki elemana erişim
- size(): Listenin uzunluğunu alma

Vector

- Vector, Java Koleksiyon Çerçevesi'nde uygulanan bir sınıftır.
- Büyüyebilen bir nesneler dizisi gerçekler.
- Dinamik bir dizi gerçeklediği için ihtiyaca göre büyür veya küçülür.
- Bir diziyi andırır, tamsayı indeks kullanılarak erişilebilen bileşenleri içerir.
- Concurrent (eşzamanlı) işlemler için uygun değildir.

Vector Kullanımı

- add(E e): Eleman ekleme
- remove(int index): Belirli bir indeksteki elemanı çıkarma
- get(int index): Belirli bir indeksteki elemana erişim
- size(): Listenin uzunluğunu alma

Stack

- Stack, Java Koleksiyon Çerçevesi'nde uygulanan bir sınıftır.
- Stack sınıfı, vektör sınıfını genişletir ve Yığın (Stack) veri yapısını uygular.
- Temel işlem, son giren ilk çıkar (last-in-first-out) ilkesine dayanır.
- Geri alma (undo) işlemleri için kullanışlıdır.
- Çoğu işlem sadece yığının üstündeki elemanı etkiler.

Stack Kullanımı

- push(E e): Eleman eklemek
- pop(): En üstteki elemanı kaldırmak
- empty(): Yığın boş mu?
- search(Object o): Belirli bir elemanın indeksini bulma
- peek(): En üstteki elemana erişmek

CopyOnWriteArrayList

- CopyOnWriteArrayList, Java Koleksiyon Çerçevesi'nde uygulanan bir sınıftır.
- "Yazarken Kopyala" (Copy-on-Write) stratejisini kullanır.
  - Veriler üzerinde değişiklik yapıldığında orijinal veriyi kopyalar ve işlemleri kopya üzerinde gerçekleştirir.
- Eşzamanlı (concurrent) erişime karşı güvenli: Çoklu iş parçacıkları arasında güvenli bir şekilde kullanılabilir.
- Okuma işlemleri hızlıdır: Veri okuma işlemleri çok hızlıdır, çünkü herhangi bir kilitleme veya senkronizasyon olmadan yapılır.
- Yazma işlemleri maliyetli: Veriye yazma işlemleri oldukça maliyetlidir, çünkü her yazma işlemi için verinin kopyası oluşturulur.

Arrays.asList()

- Arrays.asList(), Java'da bir dizi (array) nesnesini liste (list) türünde bir koleksiyona dönüştüren bir fonksiyondur.
- Dizi ve liste arasında verilerin paylaşıldığı bir arayüz sunar.
- Dizi kullanmanın avantajlarından yararlanırken koleksiyonların işlevselliğini elde etmek istediğimizde kullanılır.
- Verileri diziye ekledikten sonra, bu verileri daha fazla koleksiyon işlevselliği kullanmak için bir Liste'ye dönüştürmek istediğimizde kullanılır.

Soru

Java'da List koleksiyonu, hangi temel arayüzün bir alt sınıfıdır?

A) ArrayList
B) Set
C) Collection
D) Map

Soru

Bir Java List koleksiyonu, aynı öğeyi kaç kez içerebilir?

A) Yalnızca bir kez
B) Birden fazla kez
C) Hiçbir zaman
D) Sonsuz sayıda

Soru

Bir Java List koleksiyonunda öğeler nasıl sıralanır?

A) Rastgele sıra
B) Eklenme sırası
C) Alfabetik sıra
D) Büyükten küçüğe sıra

Soru

Bir Java List koleksiyonu içindeki öğeleri bir dizine nasıl dönüştürebilirsiniz?

A) toArray() yöntemi kullanarak
B) remove() yöntemi kullanarak
C) add() yöntemi kullanarak
D) indexOf() yöntemi kullanarak

Soru

Bir Java List koleksiyonunda, bir öğenin belirli bir konumda olup olmadığını kontrol etmek için hangi yöntemi kullanırsınız?

A) contains()

B) add()

C) size()

D) get()

Soru

Java'da bir ArrayList nesnesi oluştururken, varsayılan başlangıç kapasitesi ne kadardır?

A) 0

B) 10

C) 50

D) 100

Soru

Bir Java List koleksiyonundan bir öğeyi kaldırmak için kullanılan yöntem hangisidir?

A) delete()

B) discard()

C) remove()

D) erase()

Soru

Java'da List koleksiyonları, hangi pakette bulunur?

A) java.util.list
B) java.collection
C) java.util
D) java.collection.list

Soru

Bir Java List koleksiyonundaki bir öğenin indeksini belirlemek için hangi yöntemi kullanırsınız?

A) find()

B) locate()

C) get()

D) indexOf()

Soru

Bir Java List koleksiyonundaki tüm öğeleri kaldırmak için hangi yöntemi kullanırsınız?

A) deleteAll()

B) removeAll()

C) clear().

D) eraseAll()

Soru

Java'da bir ArrayList nesnesi oluştururken, başlangıç kapasitesi neden önemlidir?

A) Bellek yönetimi için

B) Verilerin sıralanması için

C) Hızlı erişim için

D) Bellek kullanımını azaltmak için

Soru

Java List koleksiyonları, hangi veri yapısı temelinde çalışır?

A) İkili ağaç
B) İkili arama ağacı
C) Dizi
D) Bağlantılı liste

Soru

Bir Java List koleksiyonundaki öğeleri ters sırayla nasıl sıralayabilirsiniz?

A) reverse() yöntemi kullanarak
B) sort() yöntemi kullanarak
C) flip() yöntemi kullanarak
D) shuffle() yöntemi kullanarak

Soru

Bir Java List koleksiyonunda, hangi yöntemi kullanarak bir öğenin indeksini belirlediğinizde öğe bulunmazsa hangi değeri döndürür?

A) -1

B) null

C) 0

D) Exception fırlatır

SON