TECHPROED

PROFESSIONAL TECHNOLOGY EDUCATION

WELCOME TO TECHPROED JAVA TUTORIAL

Testi baslatmak icin asagidaki adımları takip ediniz

Go to www.socrative.com

Click on Login

Click on Student Login

***Room Name*: A L P T E K I N 3 5 2 3**

Kayıtta kullandığınız ismi tam olarak yazınız

Time: 11 Minutes

TECHPROED
PROFESSIONAL TECHNOLOGY EDUCATION

ArrayList

ArrayList nedir?

ArrayList length'i esnek olan bir Array'dir.

ArrayList'e niçin ihtiyac duyarız?

Biz array oluştururken length'ing en basta belirlemek zorundayız ve daha sonra length'ini değiştiremeyiz.

Bu durum bizim esnek calışmamıza engel olur.

Bir array'in uzunluğunu değiştirmek istediğimizde yeni bir array oluşturmamız gerekir, ArrayList de gerekmez.

Bir array'den bir eleman silmek istediğimizde yeni bir array oluşturmamız gerekir, ArrayList de gerekmez.

ArrayList Oluşturma

```
ArrayList<String> list1 = new ArrayList<String>();
ArrayList<String> list2 = new ArrayList<>();
List<String> list3 = new ArrayList<>();
```

En çok bu kullanılır

! `ArrayList<String> list4 = new List<>();` // Compile Time Error verir, eşitliğin sağ tarafında ArrayList kullanmak zorundayız

ArrayList'i ekrana yazdırmak çok kolaydır.

```
List<String> list3 = new ArrayList<>();
System.out.println(list3);
```

ArrayList Method'ları

1) add()

add() method ArrayList'e eleman eklemek icin kullanılır.

Örnek:

```
List<String> hayvan = new ArrayList<>();
```

A) add() method index olamadan

```
hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
```

B) add() method index kullanarak

```
hayvan.add(1, "kartal");   // [kedi, kartal, yılan]
hayvan.add(0, "sinek");   // [sinek, kedi, kartal, yılan]
hayvan.add(1, "aslan");   // [sinek, aslan, kedi, kartal, yılan]

System.out.println(hayvan); // [sinek, aslan, kedi, kartal, yılan]
```

2) size()

size() method ArrayList'de kaç eleman olduğunu gösterir.

```
List<String> hayvan = new ArrayList<>();
System.out.println(hayvan.size()); // 0

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
System.out.println(hayvan.size()); // 2
```

3) isEmpty()

isEmpty() method ArrayList'in boş olup olmadığını kontrol eder.
Boş ise ekrana "true", boş değil ise "false" yazdırır.

```
List<String> hayvan = new ArrayList<>();
System.out.println(hayvan.isEmpty()); // true

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
System.out.println(hayvan.isEmpty()); // false
```

4) remove()

remove() methodu ArrayList'den belli bir elemanı silmek icin kullanılır.

A) remove() index kullanarak

Index'li remove() methodu ArrayList'de verilen index'deki elemanı siler.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
hayvan.remove(1);
System.out.println(hayvan); //[kedi]
```

Not: Index'li remove() u direkt System.out.println() icine koyarsak silinen elemanı ekrana yazdırır.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
System.out.println(hayvan.remove(1)); //yılan
```

B) remove() index'i degil elemanı kullanırsak kullandığımız elemanın ilk görünümünü siler.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
hayvan.add("kedi");  // [kedi, yılan, kedi]
hayvan.remove("kedi")
System.out.println(hayvan); //[yılan, kedi]
```

Not: Index'siz remove() u direkt System.out.println() icine koyarsak ekrana true veya false yazdırır.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
System.out.println(hayvan.remove("kedi")); //true yani kedi eleman olarak vardi ve sildim
System.out.println(hayvan.remove("tavsan")); //false yani tavsan eleman olarak yoktu ve silemedim
```

5) set()

set() methodu ArrayList'de var olan bir elemanı değiştirmeye yarar.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
hayvan.set(1, "tavşan");
System.out.println(hayvan); //[kedi, tavşan]
```

6) clear()

clear() methodu ArrayList'deki tüm elemanları silmek icin kullanılır.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
System.out.println(hayvan); //[kedi, tavşan]
hayvan.clear();
System.out.println(hayvan); //[]
```

7) contains()

contains() methodu ArrayList'de bir elemanın var olup olmadigini kontrol eder. Eleman varsa true, yoksa false return eder.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("kedi");   // [kedi]
hayvan.add("yılan");   // [kedi, yılan]
System.out.println(hayvan.contains("kedi")); //true
System.out.println(hayvan.contains("tavşan")); //false
```

8) sort()

sort() methodu ArrayList'deki elemanları küçükten büyüğe veya alfabetik sıraya göre dizer.

```
List<String> hayvan = new ArrayList<>();

hayvan.add("yılan");   // [yılan]
hayvan.add("kedi");   // [yılan, kedi]
hayvan.add("tavsan");   // [yılan, kedi, tavsan]
System.out.println(hayvan); //[yılan, kedi, tavsan]
Collections.sort(hayvan);
System.out.println(hayvan); //[kedi, tavsan, yılan]
```

```
List<Integer> numbers = new ArrayList<>();

hayvan.add(3);   // [3]
hayvan.add(7);   // [3, 7]
hayvan.add(1);   // [3, 7, 1]
System.out.println(numbers); //[3, 7, 1]
Collections.sort(numbers);
System.out.println(hayvan); //[1, 3, 7]
```

equals()

equals() methodu iki listteki aynı indexteki elemanların aynı olup olmadığını kontrol eder.
Aynı indexteki tum elemanlar aynı ise true return eder, farklı ise false return eder.

```
List<String> one = new ArrayList<>();
List<String> two = new ArrayList<>();

System.out.println(one.equals(two));    // true

one.add("a");          // [a]
System.out.println(one.equals(two));    // false

two.add("a");          // [a]
System.out.println(one.equals(two));    // true

one.add("b");         // [a,b]
two.add(0, "b");    // [b,a]
System.out.println(one.equals(two));    // false
```

Ödev

1) Elemanları A, C, E, ve F olan bir String ArrayList oluşturup ekrana yazdırınız.

2) İndexsiz add() methodunu kullanarak, B'yi ekleyiniz.
İndex'li add() methodunu kullanarak, L'yi 1 numaralı index'e ekleyiniz.
ArrayList'i ekrana yazdırınız, list şöyle olmalı; A, L, C, E, F, B.

3) set() methodu kullanarak, E'yi D yapınız.
ArrayList'i ekrana yazdırınız, list şöyle olmalı; A, L, C, D, F, B.

4) remove() methodu kullanarak, F'yi siliniz.
ArrayList'i ekrana yazdırınız, list şöyle olmalı; A, L, C, D, B.

5) sort() methodu kullanarak, elemanları alfabetik sıraya diziniz.
ArrayList'i ekrana yazdırınız, list şöyle olmalı; A, B, C, D, L.

6) contains() methodu kullanarak, L'nin list'de var oldugunu ve M'nin list'de
var olmadığını doğrulayınız.

7) size() methodu kullanarak, list'in kaç eleman olduğunu ekrana yazdırınız.

8) clear() methodu kullanarak, list'deki tum elemanları siliniz.

9) isEmpty() methodu kullanarak, list'deki tum elemanların silindiğini doğrulayınız.

Array'i ArrayList'e Çevirmek

```
String[] arr = { "hawk", "robin" };              // [hawk, robin]

List<String> list = Arrays.asList(arr);          // array to uzunluğu değiştirilemeyen bir list'e cevirir.
                                                 // Yani; yeni oluşturulan listte add(), remove() ve clear() methodlarını kullanamazsınız.

System.out.println(list.size());                 // 2

System.out.println(list);                        // [hawk, robin]
```

Note: Eğer array'deki bir elemanı değiştirirseniz listteki eleman da otomatik olarak değişir.
Listteki bir elemanı değiştirirseniz array de otomatik olarak değişir.

```
list.set(1, "test");                             // [hawk, test]

arr[0] = "new";                                  // [new, test]

System.out.println(Arrays.toString(arr));        // [hawk, robin]

System.out.println(list);                        // [hawk, robin]
```

Note:

```
list.add("Ali");
list.remove(1);
list.clear();
// methodlari Run Time Error verir. UnsupportedOperation Exception
```

ArrayList'i Array'e Çevirmek

```
List<String> list = new ArrayList<>();
list.add("hawk");
list.add("robin");
System.out.println(list);                          // [hawk, robin]
```

1. Yöntem

```
String arr[ ] = list.toArray(new String[0]);
System.out.println(arr.length);                    // 2
System.out.println(Arrays.toString(arr));          // [hawk, robin]
```

2. Yöntem

```
Object arr[ ] = list.toArray( );
System.out.println(arr.length);                    // 2
System.out.println(Arrays.toString(arr));          // [hawk, robin]
```

For-each Loop / Enhanced For Loop

Faydalar:

Kodun daha okunabilir olmasını saglar.
Hata yapma ihtimalini azaltır.

```
public static void main(String args[ ]){

    int arr[ ]={12,13,14,44};

    for( int i : arr ) {
      System.out.print(i + " ");
    }

}
```

For-each Loop / Enhanced For Loop

```
public static void main(String args[ ]){

    List<String> list=new ArrayList<>();
     list.add("Ali");
     list.add("Veli");
     list.add("Can");

    for( String s : list ) {
      System.out.print(s + " ");
    }

}
```

For-each loop Soru 1:

Bir integer array oluşturunuz ve bu array'deki tum sayıların çarpımını For-each loop kullanarak bulunuz. Sonucu ekrana yazdırınız.

For-each loop Soru 2:

Bir integer list oluşturunuz ve bu list'deki tum sayıların karesinin toplamını For-each loop kullanarak bulunuz. Sonucu ekrana yazdırınız.

For-each loop Soru 3:

İki String array oluşturunuz ve bu array'lerdeki ortak elemanları For-each loop kullanarak bulunuz. Sonucu ekrana yazdırınız.
Ortak eleman yoksa ekrana "Ortak eleman yok" yazdırınız

For-each loop Soru 4:

Bir String oluşturunuz, bu String'deki character'leri yukarıdan aşağıya for-each loop kullanarak yazdırınız.
İpucu: split()