

## Class Vector<E>

java.util.Vector,

Member of the Java Collections Framework.

- Assignment 6 makes use of the Vector class to store elements of the heap.
- The Vector class implements a growable array of objects. It is similar to an array, components can be accessed via an integer index.
- The size of a vector can grow or shrink as needed to accommodate adding and removing items.

*Why are we using a Vector instead of an array?*

Java does not allow generic types to be used in array declarations!

### Declaration

```
Vector<E> myVector;
```

### Constructor Summary

The following constructors can be used to initialize a Vector

#### Vector()

```
myVector = new Vector<E>();
```

Constructs an empty vector so that its internal data array has size 10 and its standard capacity increment is zero.

--	--	--	--	--	--	--	--	--	--

#### Vector(int initialCapacity)

```
myVector = new Vector<E>(5);
```

Constructs an empty vector with the specified initial capacity (5) and with its capacity increment equal to zero.

#### Vector(int initialCapacity, int capacityIncrement)

```
myVector = new Vector<E>(5, 2);
```

Constructs an empty vector with the specified initial capacity (5) and capacity increment (2). As the Vector reaches its size(5), it's capacity gets incremented by 2.

1	2	3	4	5		
---	---	---	---	---	--	--

## Method Summary

These are some of the methods you can use in Assignment 6. For a complete summary of the methods, please go through the reference provided at the end of this document.

### `boolean add(E e)`

*Appends* the specified element to the end of this Vector.

```
myVector.add(10);
```

10
----

```
myVector.add(20);
```

10	20
----	----

### `boolean add(int index, E e)`

*Inserts* the specified element at a specified index.

```
myVector.add(1, 15);
```

10	15	20
----	----	----

### `E remove(int index)`

Removes the element at the specified index.

```
myVector.remove(1);
```

10	15	20
----	----	----

10	20
----	----

### `boolean set(int index, E e)`

*Replaces* the element at the specified position in this Vector with the specified element.

10	15	20
----	----	----

```
myVector.set(1, 100);
```

10	100	20
----	-----	----

### `E get(int index)`

Returns the element at the specified position in this Vector.

10	15	20
----	----	----

```
myVector.get(2);
```

➔ 20

### boolean isEmpty()

Returns true if the Vector is empty.

myVector.isEmpty();

- ➔ Note: isEmpty() returns false even if the Vector contains only null objects.
- ➔ Also, this is the vector's isEmpty() method. Do not confuse this with our heap's isEmpty() method. It is possible to call myVector.isEmpty() inside the heap's isEmpty() method.

### int size()

Returns the number of elements in the Vector.

- ➔ Again, note that this size() is different from the heap's size() method. You can *make use of* Vector's size() in heap's size() method.

10	15	20
----	----	----

myVector.size();

➔ 3

10	15	20	Null	45
----	----	----	------	----

myVector.size();

➔ 5

### References

<https://docs.oracle.com/javase/7/docs/api/java/util/Vector.html>