

VECTOR

I. Java Implementation

ensureCapacity

```
@Assertion(@Case(
pre = "elementData !=null && 0<=elementCount && elementCount<=elementData.length",
post = "elementData !=null && 0<=elementCount && elementCount<=elementData.length &&
    minCapacity<=elementData.length"))
public void ensureCapacity(int minCapacity) {
    modCount++;
    ensureCapacityHelper(minCapacity);
}
```

indexOf

```
@Assertion(@Case(
pre = "elementData !=null && Region.arrayReg(this.elementData,0,1) && Region.setObjReg(elem,1)
    && 0<=elementCount && elementCount<=elementData.length && index >=0 &&
    elementData.length==1",
post = "$ret==-1 || ($ret>=index && elementData[$ret]==elem)"))
public int indexOf(Object elem, int index) {
    if (elem == null) {
        for (int i = index ; i < elementCount ; i++)
            if (elementData[i]==null)
                return i;
    } else {
        for (int i = index ; i < elementCount ; i++)
            if (elem==elementData[i])
                return i;
    }
    return -1;
}
```

lastIndexOf

```
@Assertion(@Case(
pre = "elementData !=null && Region.arrayReg(this.elementData,0,1) && Region.setObjReg(elem,1)
    && 0<=elementCount && elementCount<=elementData.length ",
post = "$ret==-1 || ($ret<=index && elementData[$ret]==elem)"))
public int lastIndexOf(Object elem, int index) {
    if (index >= elementCount)
        throw new IndexOutOfBoundsException(" >= ");
    if (elem == null) {
```

```

        for (int i = index; i >= 0; i--)
            if (elementData[i]==null)
                return i;
    } else {
        for (int i = index; i >= 0; i--)
            if (elem==elementData[i])
                return i;
    }
    return -1;
}

```

removeElementAt

```

@Assertion(@Case(
pre = "elementData !=null && 0<=elementCount && elementCount<=elementData.length ",
post = "true"))
public void removeElementAt(int index) {
    modCount++;
    if (index >= elementCount) {
        throw new ArrayIndexOutOfBoundsException(" >= ");
    } else if (index < 0) {
        throw new ArrayIndexOutOfBoundsException("<0");
    }
    int j = elementCount - index - 1;
    if (j > 0) {
        for(int i=0;i<j;i++) {
            elementData[index+i]=elementData[index+i+1];
        }
    }
    elementCount--;
    elementData[elementCount] = null; /* to let gc do its work */
}

```

insertElementAt

```

@Assertion(@Case(
pre = "elementData !=null && 0<=elementCount && elementCount<=elementData.length && index
    >=0 && index<=elementCount",
post = "elementData !=null && 0<=elementCount && elementCount<=elementData.length"))
public void insertElementAt(E obj, int index) {
    modCount++;
    if (index > elementCount) {

```

```

        throw new ArrayIndexOutOfBoundsException(">");
    }
    ensureCapacityHelper(elementCount + 1);
    for(int i=0;i<elementCount - index;i++) {
        elementData[index+1+i]=elementData[index+i];
    }
    elementData[index] = obj;
    elementCount++;
}

```

add

```

@Assertion( @Case(
pre = "elementData !=null && 0<=elementCount && elementCount<=elementData.length",
post = "elementData !=null && 0<=elementCount && elementCount<=elementData.length"))
public boolean add(E o) {
    modCount++;
    ensureCapacityHelper(elementCount + 1);
    elementData[elementCount++] = o;
    return true;
}

```