K. J. Somaiya College of Engineering, Mumbai-77

(A Constituent College of Somaiya Vidyavihar University)

Department of Computer Engineering

Batch No. B2	**Roll No. 16010121110**
Experiment / Assignment/ Tutorial No. 5	
Grade : AA/AB/BB/BC/CC/CD/DD	
Signature of the Staff In-charge with date	

TITLE : Vector

AIM: Create a class Employee which stores E-Name, E-Id and E-Salary of an Employee. Use class Vector to maintain an array of Employee with respect to the E-Salary. Provide the following functions

1) Create (): this function will accept the n Employee records in any order and will arrange them in the sorted order.

2) Insert (): to insert the given Employee record at appropriate index in the vector depending upon the E-Salary.

3) delete ByE-name( ): to accept the name of the Employee and delete the record having given name

4) deleteByE-Id ( ): to accept the Id of the Employee and delete the record having given E-Id.

Provide the following functions

1. boolean add(E e) : This method appends the specified element to the end of this Vector.

2. void addElement(E obj) This method adds the specified component to the end of this vector, increasing its size by one.

3. int lastIndexOf(Object o, int index) This method returns the index of the last occurrence of the specified element in this vector, searching backwards from index, or returns -1 if the element is not found.

4. void removeElementAt(int index)This method deletes the component at the specified index.

Expected OUTCOME of Experiment:

CO2: Explore arrays, vectors, classes and objects in C++ and Java.

Books/ Journals/ Websites referred:

1. Ralph Bravaco , Shai Simoson , “Java Programing From the Group Up” Tata McGraw-Hill.

2.Grady Booch, Object Oriented Analysis and Design .

Pre Lab/ Prior Concepts:

Vectors in Java are one of the most commonly used data structures. Similar to Arrays data structures which hold the data in a linear fashion. Vectors also store the data in a linear fashion, but unlike Arrays, they do not have a fixed size. Instead, their size can be increased on demand.
Vector class is a child class of AbstractList class and implements on List interface. To use Vectors, we first have to import Vector class from java.util package:

```
import java.util.Vector;
```

Access Elements in Vector:

We can access the data members simply by using the index of the element, just like we access the elements in Arrays.

Example- If we want to access the third element in a vector v, we simply refer to it as v[3].

Vectors Constructors

Listed below are the multiple variations of vector constructors available to use:

1. **Vector(int initialCapacity, int Increment)** – Constructs a vector with given initialCapacity and its Increment in size.
2. **Vector(int initialCapacity) –** Constructs an empty vector with given initialCapacity. In this case, Increment is zero.
3. **Vector()** – Constructs a default vector of capacity 10.
4. **Vector(Collection c) –** Constructs a vector with a given collection, the order of the elements is same as returned by the collection's iterator.

There are also three protected parameters in vectors

- **Int capacityIncrement()-** It automatically increases the capacity of the vector when the size becomes greater than capacity.
- **Int elementCount()** – tell number of elements in the vector
- **Object[] elementData()** – array in which elements of vector are stored

Memory allocation of vectors:

Vectors do not have a fixed size, instead, they have the ability to change their size dynamically. One might think that the vectors allocate indefinite long space to store objects. But this is not the case. Vectors can change their size based on two fields 'capacity' and 'capacityIncrement'. Initially, a size equal to 'capacity' field is allocated when a vector is declared. We can insert the elements equal to the capacity. But as soon as the next element is inserted, it increases the size of the array by size 'capacityIncrement'. Hence, it is able to change its size dynamically.
For a default constructor, the capacity is doubled whenever the capacity is full and a new element is to be inserted.

Methods of Vectors :

- Adding elements
- Removing elements
- Changing elements
- Iterating the vector

Department of Computer Engineering

Implementation details:

/**

AIM: Create a class Employee which stores E-Name, E-Id and E-Salary of an Employee. Use class Vector to maintain an array of Employee with respect to the E-Salary. Provide the following functions

1) Create (): this function will accept the n Employee records in any order and will arrange them in the sorted order.

2) Insert (): to insert the given Employee record at appropriate index in the vector depending upon the E-Salary.

3) delete ByE-name( ): to accept the name of the Employee and delete the record having given name

4) deleteByE-Id ( ): to accept the Id of the Employee and delete the record having given E-Id.

Provide the following functions

1. boolean add(E e) : This method appends the specified element to the end of this Vector.

2. void addElement(E obj) This method adds the specified component to the end of this vector, increasing its size by one.

3. int lastIndexOf(Object o, int index) This method returns the index of the last occurrence of the specified element in this vector, searching backwards from index, or returns -1 if the element is not found.

4. void removeElementAt(int index)This method deletes the component at the specified index.

**********/

```
public class Main
{
        public static void main(String[] args) {
        Employee a = new Employee("Suleman",1,20);
        Employee b = new Employee("Shambho",2,11);
        Employee c = new Employee("Pratik",13,33);
        EmployeeVector vect = new EmployeeVector();
        vect.create(a,b,c);
        vect.display();
        System.out.println();
        vect.delete(2);
        vect.display();

        }
}
```

```
import java.util.*;

class Employee implements Comparable<Employee>{
   String name;
   int ID;
   int salary;
   Employee(String name,int ID,int salary){
      this.name=name;
      this.ID=ID;
      this.salary=salary;
   }
   public String toString(){
      return "Employee "+name+" id "+ID+" earns "+salary+" $";
   }
   public int compareTo(Employee b){


     if (salary>b.salary){
        return 1;
     }
     else if(salary<b.salary){
        return -1;
     }
```

```
    return 0;
  }
}

import java.util.*;
public class EmployeeVector{
  Vector<Employee> EmployeeList = new Vector<Employee>();
  void create(Employee ... a){
    for (Employee i:a){
      EmployeeList.add(i);
    }
    sort();
  }
  void sort(){
    Collections.sort(EmployeeList);
  }
  void insert(Employee a){
    EmployeeList.add(a);
    sort();
  }
  void delete(String name){
```

```
    //ignore if element not found
     for (int i = 0; i < EmployeeList.size(); i++) {
       if(EmployeeList.get(i).name==name){
          EmployeeList.remove(i);
       }
    }
  }
  void delete(int ID){
    for (int i = 0; i < EmployeeList.size(); i++) {
       if(EmployeeList.get(i).ID==ID){
          EmployeeList.remove(i);
       }
    }
  }
  void display(){
    for (Employee i:EmployeeList){

      System.out.println(i);
    }
  }
```

}

Output:

Employee Shambho id 2 earns 11 $

Employee Suleman id 1 earns 20 $

Employee Pratik id 13 earns 33 $

Employee Suleman id 1 earns 20 $

Employee Pratik id 13 earns 33 $

Conclusion:

Thus we have understood what vectors are in Java, how they are implemented and their methods. Vectors are defined in collections class in java and are inherited from lists.

Date: 18 oct 22 **Signature of faculty in-charge**

Post Lab Descriptive Questions

1. **What is the output of the following Program**

```
import java.util.*;
class demo2 {
   public static void main(String[] args)
   {
      Vector v = new Vector(20);
      v.addElement("Geeksforgeeks");
      v.insertElementAt("Java", 2);
      System.out.println(v.firstElement());
   }
}
```

Output:

Geeksforgeeks

2. **Explain any 10 methods of Vector class in detail with the help of example**

add()	It is used to append the specified element in the given vector.
addAll()	It is used to append all of the elements in the specified collection to the end of this Vector.
addElement()	It is used to append the specified component to the end of this vector. It increases the vector size by one.
capacity()	It is used to get the current capacity of this vector.

clear()	It is used to delete all of the elements from this vector.
clone()	It returns a clone of this vector.
contains()	It returns true if the vector contains the specified element.
containsAll()	It returns true if the vector contains all of the elements in the specified collection.
copyInto()	It is used to copy the components of the vector into the specified array.
elementAt()	It is used to get the component at the specified index.
elements()	It returns an enumeration of the components of a vector.
ensureCapacity()	It is used to increase the capacity of the vector which is in use, if necessary. It ensures that the vector can hold at least the number of components specified by the minimum capacity argument.

Reference https://www.javatpoint.com/java-vector