**Batch: B3 Roll No.: 121**

**Experiment / assignment / tutorial No.05**

**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](https://www.edureka.co/blog/constructor-in-java/) 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:**

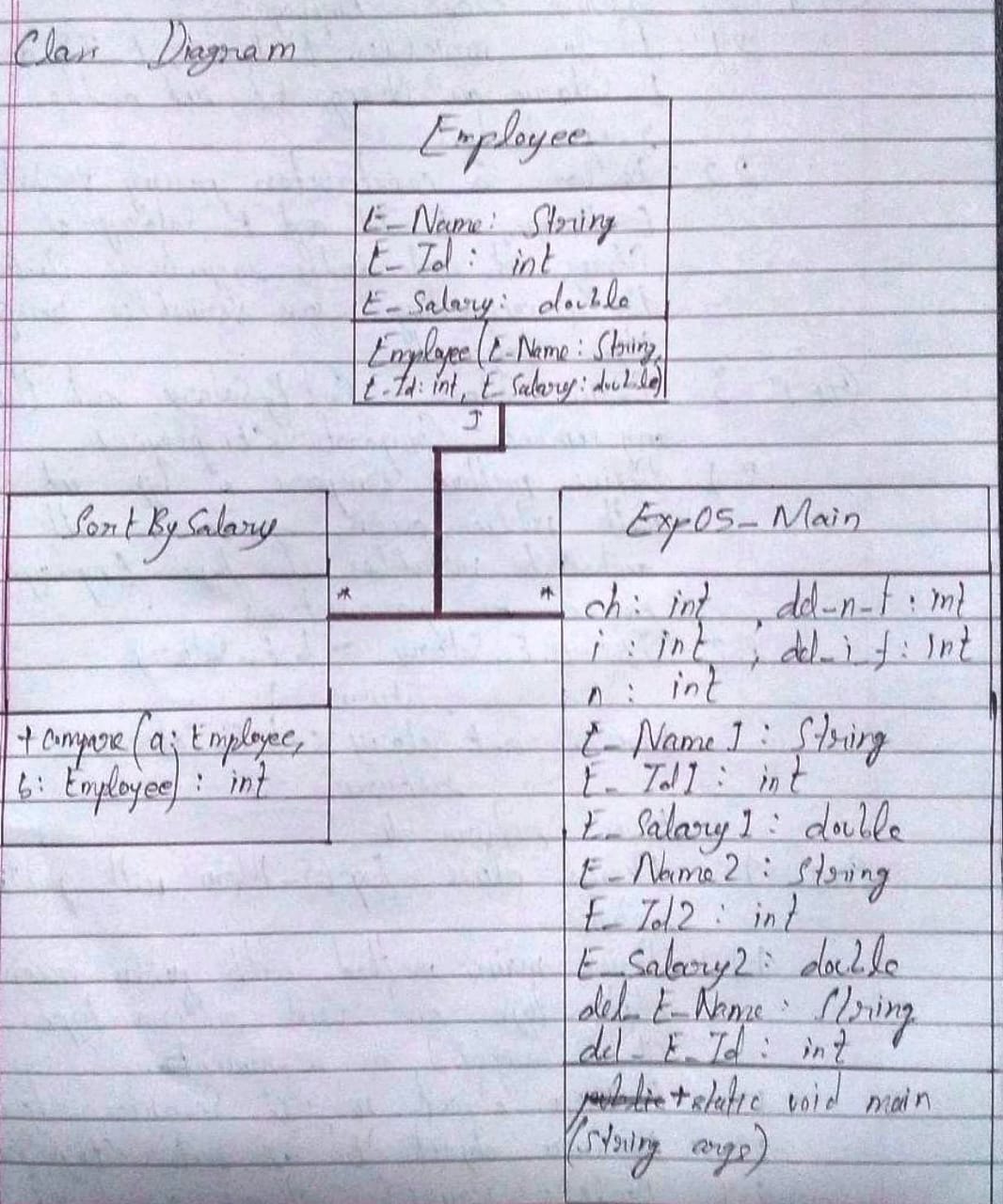
Vectors do not have a fixed size, instead, they have the ability to change their size dynamically. One might think that the vectors allocate indefinite long space to store objects. But this is not the case. Vectors can change their size based on two fields ‘capacity’ and ‘capacityIncrement’. Initially, 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.

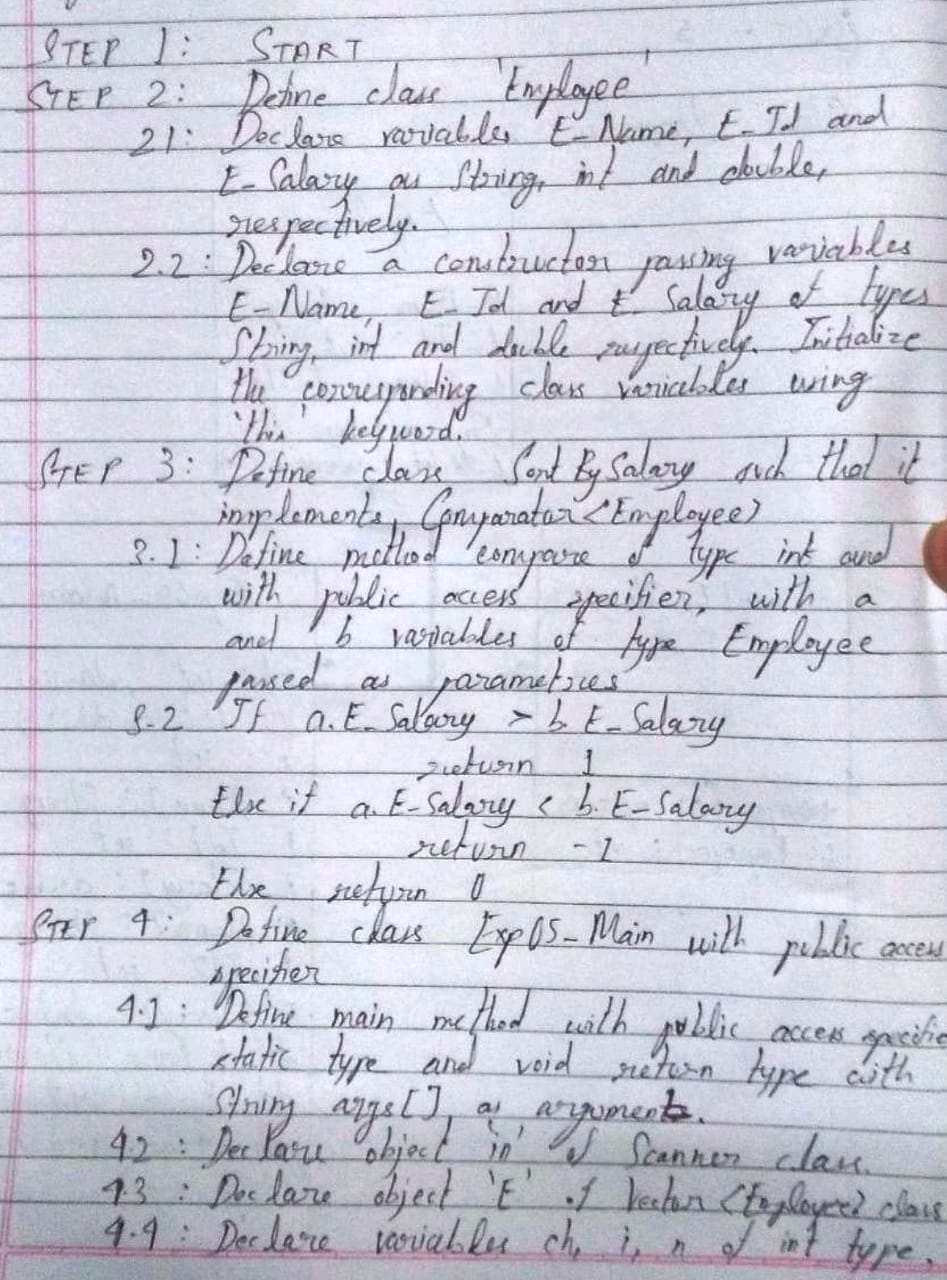
**Methods of Vectors :**

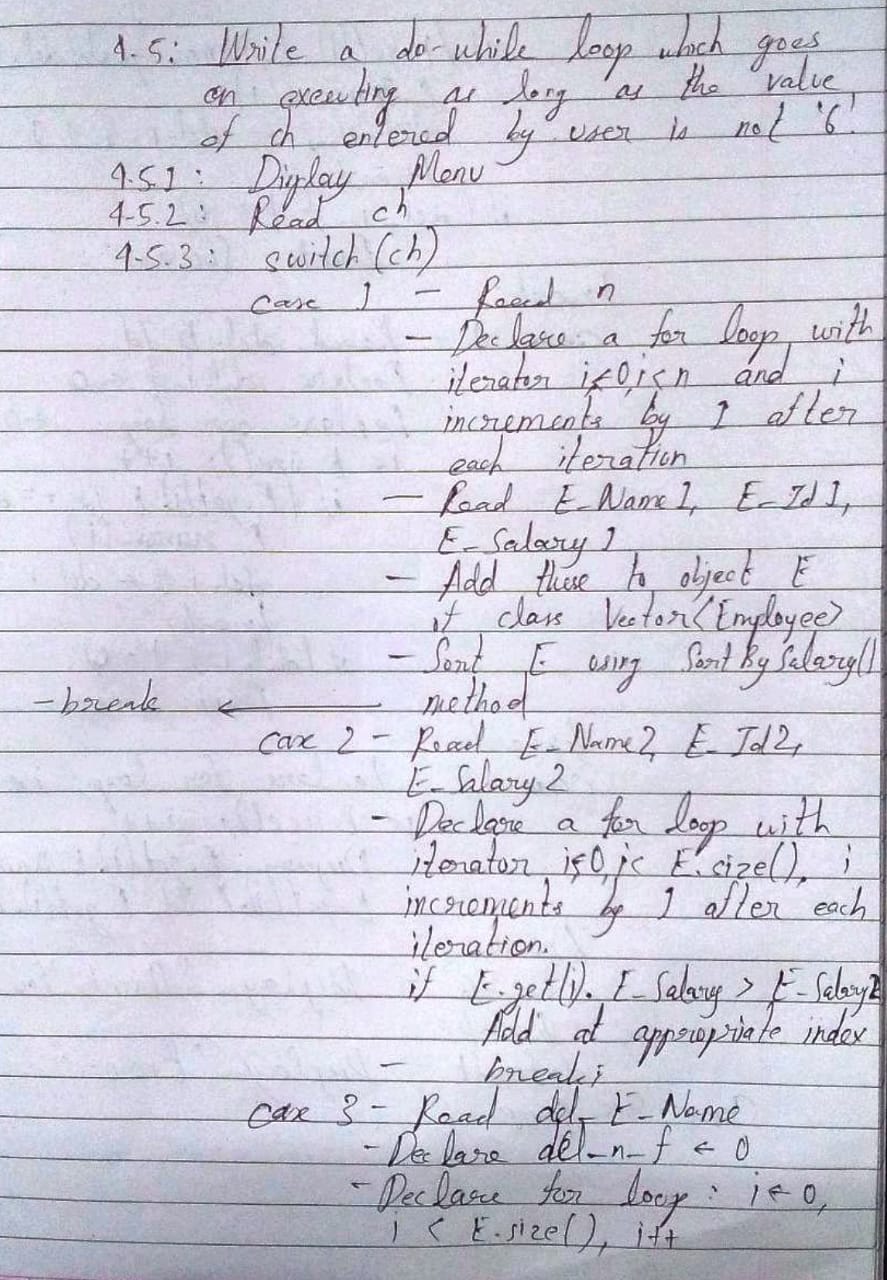
* Adding elements
* Removing elements
* Changing elements
* Iterating the vector

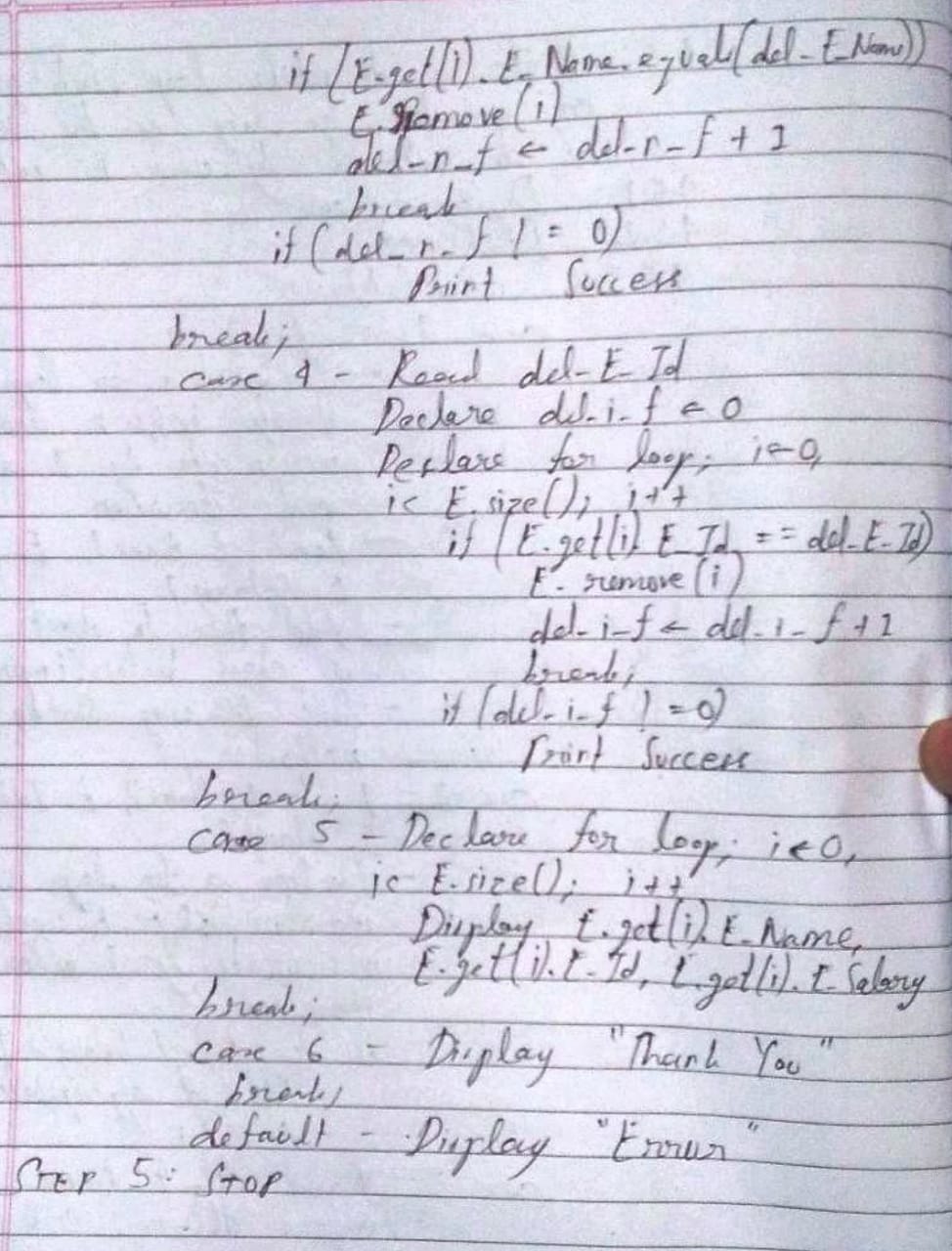
**Class Diagram:**



**Algorithm:**







**Implementation details:**

import java.util.\*;

class Employee

{

String E\_Name;

int E\_Id;

double E\_Salary;

Employee(String E\_Name, int E\_Id, double E\_Salary)

{

this.E\_Name = E\_Name;

this.E\_Id = E\_Id;

this.E\_Salary = E\_Salary;

}

}

class SortBySalary implements Comparator<Employee>

{

public int compare(Employee a, Employee b)

{

if(a.E\_Salary > b.E\_Salary)

return 1;

else if(a.E\_Salary < b.E\_Salary)

return -1;

else

return 0;

}

}

public class Exp05\_Main

{

public static void main(String args[])

{

Scanner in = new Scanner(System.in);

Vector<Employee> E = new Vector<Employee>();

int ch, i, n;

do

{

System.out.println("Enter:\n'1' to create a record of employees.\n'2' to insert more employee data.\n'3' to delete an employee's data based on name.\n'4' to delete an employee's data based on id.\n'5' to display the employee record.\n'6' to exit from the system.\nEnter your choice: ");

ch = in.nextInt();

switch(ch)

{

case 1:

System.out.println("Enter the number of employees initially: ");

n = in.nextInt();

for(i = 0; i < n; i++)

{

System.out.println("For Employee "+(i+1)+", enter Name: ");

String E\_Name1 = in.next();

System.out.println("For "+E\_Name1+", enter Employee ID: ");

int E\_Id1 = in.nextInt();

System.out.println("For "+E\_Id1+", enter Salary: ");

double E\_Salary1 = in.nextDouble();

E.addElement(new Employee(E\_Name1, E\_Id1, E\_Salary1));

}

E.sort(new SortBySalary());

break;

case 2:

System.out.println("For new Employee, enter Name: ");

String E\_Name2 = in.next();

System.out.println("For "+E\_Name2+", enter Employee ID: ");

int E\_Id2 = in.nextInt();

System.out.println("For "+E\_Id2+", enter Salary: ");

double E\_Salary2 = in.nextDouble();

for(i = 0; i < E.size(); i++)

{

if(E.get(i).E\_Salary > E\_Salary2)

{

E.add(i, new Employee(E\_Name2, E\_Id2, E\_Salary2));

break;

}

}

break;

case 3:

System.out.println("Enter the name of the employee whose data you want to delete: ");

String del\_E\_Name = in.next();

int del\_n\_f = 0;

for(i = 0; i < E.size(); i++)

{

if(E.get(i).E\_Name.equals(del\_E\_Name))

{

E.remove(i);

del\_n\_f++;

break;

}

if(del\_n\_f != 0)

System.out.println("Deletion Successful.");

else

System.out.println("Deletion Unsuccessful.");

}

break;

case 4:

System.out.println("Enter the id of the employee whose data you want to delete: ");

int del\_E\_Id = in.nextInt();

int del\_i\_f = 0;

for(i = 0; i < E.size(); i++)

{

if(E.get(i).E\_Id == del\_E\_Id)

{

E.remove(i);

del\_i\_f++;

break;

}

if(del\_i\_f != 0)

System.out.println("Deletion Successful.");

else

System.out.println("Deletion Unsuccessful.");

}

break;

case 5:

System.out.println("Sr. No.\tName\t\tId\tSalary");

for(i = 0; i < E.size(); i++)

{

System.out.println((i+1)+"\t"+E.get(i).E\_Name+"\t\t"+E.get(i).E\_Id+"\t"+E.get(i).E\_Salary);

}

break;

case 6:

System.out.println("THANK\tYOU\tVERY\tMUCH\nE\tX\tI\tT\tI\tN\tG\t.\t.\t.");

break;

default:

System.out.println("Please enter '1', '2', '3', '4', '5' or '6' only and try again!");

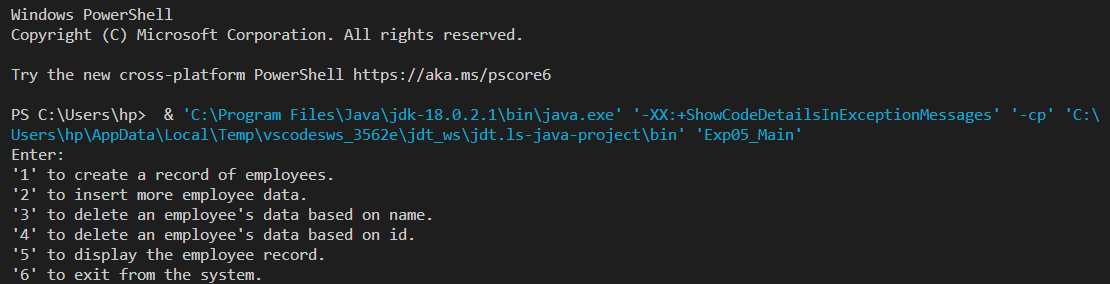
}

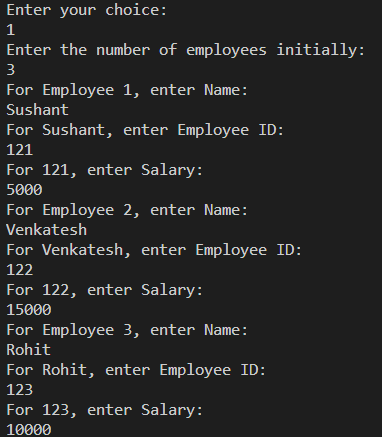
}while(ch!=6);

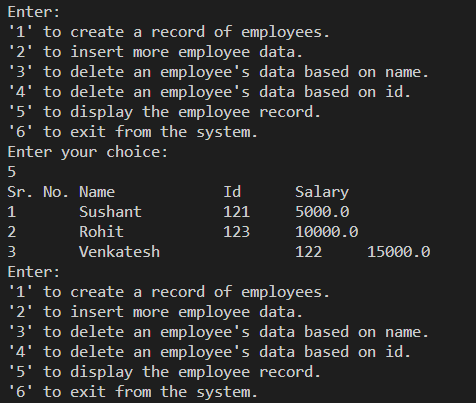
}

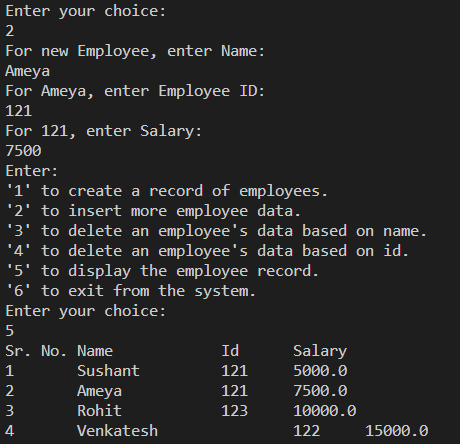
}

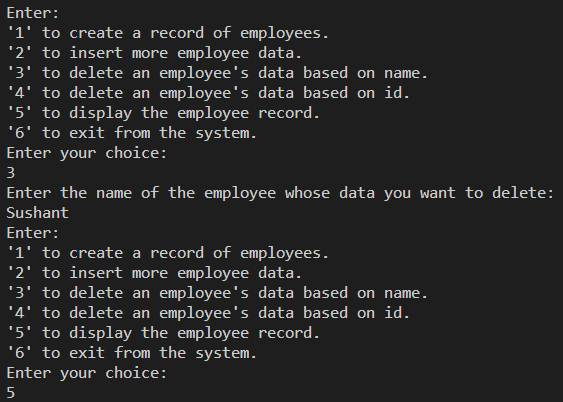
**Output:**

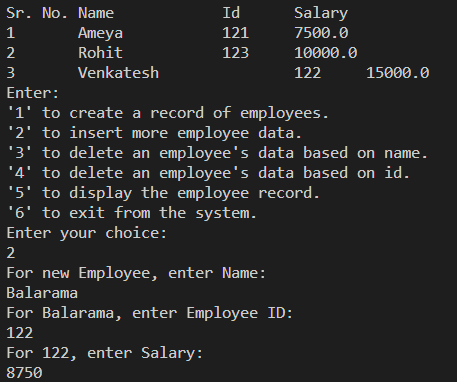


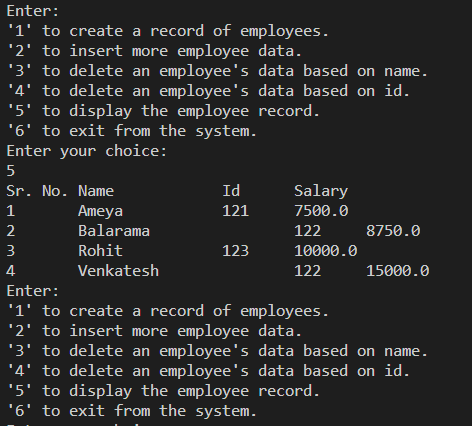


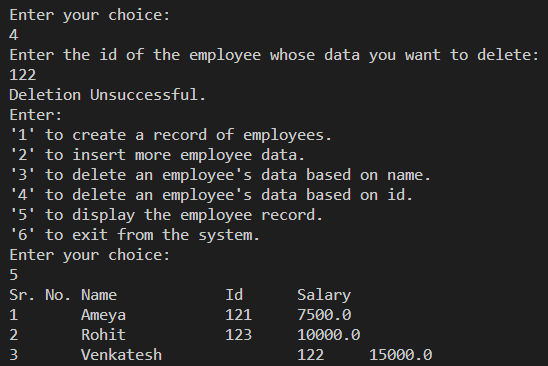


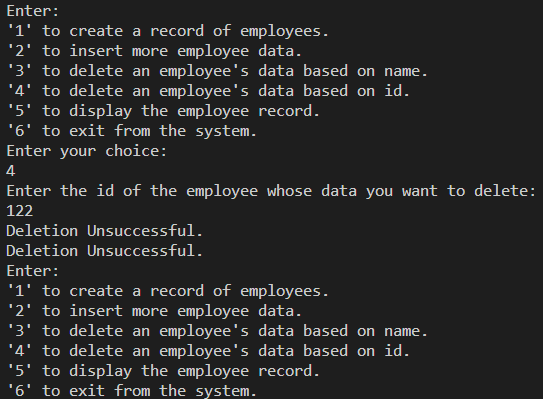


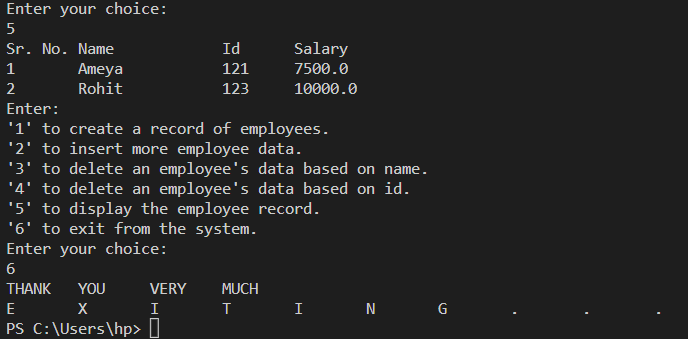












**Conclusion:**

Thus, in this experiment, the concept of Vectors in Java was learnt and a program implementing that concept was written and successfully run.

**Date: \_\_09-11-22\_\_ 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: java.lang.ArrayIndexOutOfBoundsException: 2 > 1

1. **Expain any 10 methods of Vector class in detail with the help of example**

**Ans.**

|  |  |  |
| --- | --- | --- |
| Sr. No. | Method Name | Description |
| 1. | add() | It is used to append the specified element in the vector. |
| 2. | addElement() | It is used to append the specified component to the end of this vector . It increases the vector size by one. |
| 3. | capacity() | It is used to get the current capacity of the Vector. |
| 4. | contains() | It returns true if the vector contains the specified element. |
| 5. | equals() | It is used to compare the specified object with the vector for equality. |
| 6. | firstElement() | It is used to get the first component of the Vector. |
| 7. | lastElement() | It is used to get the last component of the Vector. |
| 8. | remove() | It is used to remove the specified element from the vector. If the vector does not contain the element, it is unchanged. |
| 9. | size() | It is used to get the number of components in the vector. |
| 10. | sort() | It is used to sort the list according to the order induced by the specified Comparator. |

Example of Vector:

import java.util.\*;

public class fruits\_vector

{

    public static void main(String args[])

    {

        Vector<String> v = new Vector<String>(4);

        v.add("Banana");

        v.add("Jackfruit");

        v.add("Guava");

        v.add("Chicoo");

        System.out.println("Size of vector is: "+v.size());

        System.out.println("Default capacity is: "+v.capacity());

        System.out.println("Vector elements are: "+v);

        v.addElement("Papaya");

        v.addElement("Grape");

        v.addElement("Watermelon");

        System.out.println("Size after addition is: "+v.size());

        System.out.println("Capacity after addition is: "+v.capacity());

        System.out.println("Vector elements are: "+v);

        if(v.contains("Papaya"))

        {

            System.out.println("Papaya is present at index "+v.indexOf("Papaya")+".");

        }

        else

        {

            System.out.println("Papaya is not present in the vector.");

        }

        System.out.println("First fruit in vector is: "+v.firstElement());

        System.out.println("Last fruit in vector is: "+v.lastElement());

    }

}

