ACPCE
Where knowledge is second nature

Jawahar Education Society's
A. C. Patil College of Engineering, Kharghar
Navi Mumbai 410210

Student Name: Chetan Ingale

Course Name: C.S.E. (IoT CS BC)

Year: S.E.

Roll No.: 17

PRN No.: 221111030

Course code: CSL304

Semester: 3

Experiment Evaluation Sheet

Experiment No.: 7

Experiment Name:
Program on String Buffer and Vectors

Sr No.	Evaluation Criteria	Marks (Out of 9)	Performance Date	Correction Date and Signature of Instructor
1	Experiment Performance			
2	Journal Performance			
3	Punctuality			
Total				

Aim : Program on String Buffer and Vectors

Software required : Java, Javac.

Theory :

Class StringBuffer :-
A thread-safe, mutable sequence of characters. A string buffer is like a String, but can be modified. At any point in time it contains some particular sequence of characters, but the length and content of the sequence can be changed through certain method calls.
String buffers are safe for use by multiple threads. The methods are synchronized where necessary so that all the operations on any particular instance behave as if they occur in some serial order that is consistent with the order of the method calls made by each of the individual threads involved.

The principal operations on a StringBuffer are the append and insert methods, which are overloaded so as to accept data of any type. Each effectively converts a given datum to a string and then appends or inserts the characters of that string to the string buffer. The append method always adds these characters at the end of the buffer; the insert method adds the characters at a specified point.

In general, if sb refers to an instance of a StringBuffer, then **sb.append(x)** has the same effect as **sb.insert(sb.length(), x)**.

Every string buffer has a capacity. As long as the length of the character sequence contained in the string buffer does not exceed the capacity, it is not necessary to allocate a new internal buffer array. If the internal buffer overflows, it is automatically made larger.

Code 7.a :

```
public class StringBuff {
  public static void main(String[] args) {
    StringBuffer sb = new StringBuffer("Hello");
    StringBuffer k = new StringBuffer("");
    System.out.println("Capasity before append:"+k.capacity());
    System.out.println();
    k.append("Java");
    k.append(" I love java.");
    System.out.println("Capasity after appends:"+k.capacity());
    System.out.println();

    sb.append(" Java");
    System.out.println("Append:"+sb);
    System.out.println();

    sb.replace(0,5,"Program");
    System.out.println("Replace :"+sb);
    System.out.println();

    sb.insert(0,"Fun ");
    System.out.println("Insert :"+sb);
    System.out.println();

    sb.delete(0,5);
```

```
        System.out.println("Delete :"+sb);
        System.out.println();

        sb.reverse();
        System.out.println("Reverse:"+sb);
        System.out.println();

        k.reverse();
        System.out.println("Reverse:"+k);
    }
}
```

Output 7.a :

```
student@csiot-ThinkCentre-M70s:~/CHETAN_I_007/OOPs/Exp07$ java StringBuff
Capasity before append:16

Capasity after appends:34

Append:Hello Java

Replace :Program Java

Insert :Fun Program Java

Delete :rogram Java

Reverse:avaJ margor

Reverse:.avaj evol I avaJ
```

Theory :

Java Vector :
Vector is like the dynamic array which can grow or shrink its size. Unlike array, we can store n-number of elements in it as there is no size limit. It is a part of Java Collection framework since Java 1.2. It is found in the java.util package and implements the List interface, so we can use all the methods of List interface here.

It is recommended to use the Vector class in the thread-safe implementation only. If you don't need to use the thread-safe implementation, you should use the ArrayList, the ArrayList will perform better in such case.

The Iterators returned by the Vector class are fail-fast. In case of concurrent modification, it fails and throws the ConcurrentModificationException.

It is similar to the ArrayList, but with two differences-

Vector is synchronized.
Java Vector contains many legacy methods that are not the part of a collections framework.

Code 7.b :

```
import java.util.*;

public class Vectors {
   public static void main(String[] args) {
      Vector<String> mammals = new Vector<>();

      mammals.add("Dog");
      mammals.add(1,"Horse ");

      System.out.println("Mammals:" + mammals);
      System.out.println();

      Vector<String> animals= new Vector <>();

      animals.add("Crocodile");
      animals.add("Whale");
      animals.add("Duck");
;

      System.out.println("Animals: "+ animals);
      System.out.println();

      animals.addAll(mammals);
      System.out.println("Extended Mammals:"+animals);
      System.out.println();

      String element= animals.get(2);
      System.out.println("Element at index 2:"+element);
   }
}
```

Output 7.b :

```
student@csiot-ThinkCentre-M70s:~/CHETAN_I_007/OOPs/Exp07$ java Vectors
Mammals:[Dog, Horse ]

Animals: [Crocodile, Whale, Duck]

Extended Mammals:[Crocodile, Whale, Duck, Dog, Horse ]

Element at index 2:Duck
```

Conclusion :

With this experiments we learn how to implement StringBuffer and Vectors in java programming language.