Dt : 12/10/2023

Note:

=>In realtime LinkedList<E> is used in the applications where we have more number of add() and remove() operations,which means used in Admin-Login of an Applications

==

****imp***

(c)Vector<E>:

=>Vector<E> is an Class from java.util package and,which is Synchronized class and which organizes elements in Sequence.

=>Vector<E> is known as Legacy class in Collection Framework.

syntax:

```
Vector<Class_name> ob = new Vector<Class_name>();
```

=>The following are some important methods from Vector<E>:

```
public synchronized int capacity();
public synchronized int size();
public synchronized boolean isEmpty();

public synchronized E elementAt(int);
public synchronized E firstElement();
public synchronized E lastElement();
public synchronized void setElementAt(E, int);
public synchronized void removeElementAt(int);
public synchronized void insertElementAt(E, int);
public synchronized void addElement(E);
public synchronized boolean removeElement(java.lang.Object);
public synchronized void removeAllElements();

public java.util.Enumeration<E> elements();
```

--

faq:

define Enumeration<E>?

=>Enumeration<E> is an interface from java.util package and which is used to retrieve elements from Vector<E> objects in forward direction.

=>The following are some important methods of Enumeration<E>:

```
public abstract boolean hasMoreElements();
public abstract E nextElement();
public default java.util.Iterator<E> asIterator();
```

=>we use elements() method to create implementation object for Enumeration<E> interface.

syntax:

Enumeration<Class_name> e = ob.elements();

==
===

Ex:

program : DemoList8.java

```
package p2;
import java.util.*;
public class DemoList8 {
     @SuppressWarnings("removal")
     public static void main(String[] args) {
        Vector<Integer> v = new Vector<Integer>();
        System.out.println("default 
capacity:"+v.capacity());
        System.out.println("size:"+v.size());
        for(int i=11;i<=20;i++)
        {
         v.addElement(new ~~Integer~~(i));
        }
        System.out.println("---addElement()----");
        System.out.println(v.toString());
        System.out.println("capacity:"+v.capacity());
        System.out.println("size:"+v.size());
        System.out.println("----insertElementAt()----");
        v.insertElementAt(new ~~Integer~~(500), 5);
        System.out.println(v.toString());
        System.out.println("capacity:"+v.capacity());
        System.out.println("size:"+v.size());
        System.out.println("----elementAt(index)----");
        System.out.println("Ele at index 5 : 
"+v.elementAt(5));
        System.out.println("====Enumeration<E>====");
        Enumeration<Integer> e = v.elements();
        while(e.hasMoreElements()) {
            Integer z = e.nextElement();
            int count=0;
            for(int i=1;i<=z;i++) {
             if(z%i==0) {
                    count++;
                }
               }//end of loop
               if(count==2) {
                System.out.print(z+" ");
               }

        }//end of loop
        System.out.println("\n----asIterator()----");
        Enumeration<Integer> e1 = v.elements();
        Iterator<Integer> it = e1.asIterator();
        it.forEachRemaining((k)->
        {
        System.out.print(k.toString()+" ");
        });
    }
}
```

o/p:

```
default capacity:10

size:0

---addElement()----

[11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

capacity:10

size:10

----insertElementAt()----

[11, 12, 13, 14, 15, 500, 16, 17, 18, 19, 20]
```

capacity:20

size:11

----elementAt(index)----

Ele at index 5 : 500

====Enumeration<E>====

11 13 17 19

----asIterator()----

11 12 13 14 15 500 16 17 18 19 20

===

Note:

=>The default capacity of Vector<E> is 10 elements.

=>The default capacity of Vector<E> can be increased dynamically by doubling the capacity.

=>asIterator() method from Enumeration<E> is used to create object for Iterator<E>.

=>java.util.StringTokenizer is an implementation class of Enumeration<E>

=>In realtime Vector<E> is used in Connection Pooling concept,which means Vector<E> holding multiple database Connections.

===

****imp***

define Stack<E>?

=>Stack<E> is a ChildClass of Vector<E> and which organizes elements

based on the algorithm first-in-last-out or Last-in-first-out

synatx:

Stack<Class_name> ob = new Stack<Class_name>();

=>The following are some important methods of Stack<E>:

```
public E push(E);
public synchronized E pop();
public synchronized E peek();
public boolean empty();
public synchronized int search(java.lang.Object);
```

push(E) : method is used to add the element to Stack<E>

pop()   : method is used to delete the element from top-of-stack

peek() : method is used display the element from top-of-stack

empty() : method is used to check the Stack<E> is empty or not

search(Object) : method is used to search the element from Stack<E> and display the position of an element.
