Dt : 4/10/2023

faq:

define Generic Programming Components?

=>The programming Components which are ready to accept any types,are known as Generaic Programming Components.

=>These Generic Programming Components are categorized into the following:

1. *Generic Types*
2. *Generic Methods*
3. *Generic Classes*
4. *Generic Interfaces*

1.Generic Types:

=>The types which are ready to accept any type of data are known as Generic Types

- *T - Type*
- *E - Element*
- *K - Key*
- *V - Value*

2.Generic Methods:

=>The methods which are ready to accept type of data as parameters are known as Generic methods.

structure of Generic Method:

<T>return_type method_name(T)

{

//Method_body

}

3.Generic Classes:

=>The class Objects which are ready to hold any type of data are known as Generic Classes.

structure of Generic Class:

class Class_name<T>

{

//Class_body

}

4.Generic Interfaces:

=>The interfaces which are implemented to Generic Classes are known as Generic Interfaces.

structure of Generic Interface:

interface Interface_name<T>

{

//Interface_body

}

==========

Ex-program:(Demonstrating user defined Generic Programming Components)

p1 : User.java

```
package p1;
public class User {
    public String user;
    public User(String user) {
        this.user=user;
    }
    public String toString() {
        return user;
    }
}
```

p1 : Display.java

```
package p1;
public class Display<T>
{
     public T ob;

    public final T getOb() {
        return ob;
    }

    public final void setOb(T ob) {
        this.ob = ob;
    }
```

```
}
```

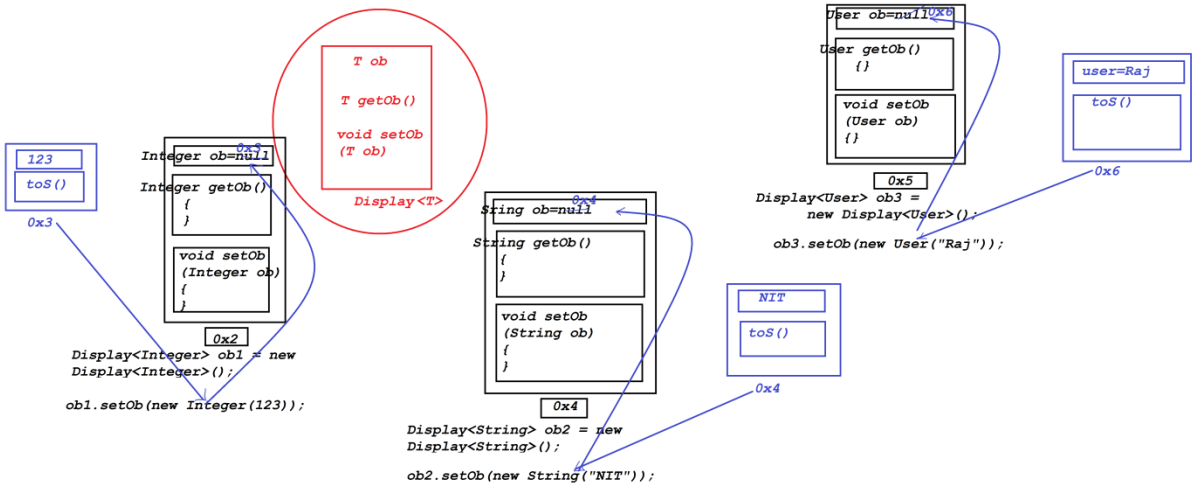
p2 : DemoGeneric.java(MainClass)

```
package p2;
import p1.Display;
import p1.User;
public class DemoGeneric {
        @SuppressWarnings("removal")
        public static void main(String[] args) {
    Display<Integer> ob1 = new Display<Integer>();
    ob1.setOb(new Integer(123));
    System.out.println("ob1 : "+ob1.getOb());
    Display<String> ob2 = new Display<String>();
    ob2.setOb(new String("NIT"));
    System.out.println("ob2 : "+ob2.getOb());
    Display<User> ob3 = new Display<User>();
    ob3.setOb(new User("Raj"));
    System.out.println("ob3 : "+ob3.getOb());
        }
}
```

o/p:

```
ob1 : 123
ob2 : NIT
ob3 : Raj
```

Diagram:



==========

1.Set<E>:

=>Set<E> is an interface from java.util package and which organizes elements without index values and cannot hold duplicate elements.

=>Set<E> means no index values and cannot hold duplicate elements.

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

public abstract int size();

public abstract boolean isEmpty();

public abstract boolean add(E);

public abstract boolean remove(java.lang.Object);

public abstract boolean containsAll(java.util.Collection<?>);

public abstract boolean addAll(java.util.Collection<? extends E>);

public abstract boolean retainAll(java.util.Collection<?>);

public abstract boolean removeAll(java.util.Collection<?>);

public abstract void clear();

public static <E> java.util.Set<E> of();

public abstract boolean contains(java.lang.Object);

public abstract java.util.Iterator<E> iterator();

public default java.util.Spliterator<E> spliterator();

public abstract java.lang.Object[] toArray();

public abstract <T> T[] toArray(T[]);

=>The following are the implementation classes of Set<E>:

(a)HashSet<E>

(b)LinkedHashSet<E>

(c)TreeSet<E>

(a)HashSet<E>:

=>HashSet<E> Organizes elements without any order.

(b)LinkedHashSet<E>:

=>LinkedHashSet<E> Organizes elements in insertion Order.

(c)TreeSet<E>:

=>TreeSet<E> Organizes elements automatically in Ascending Order.

Ex-Program:(Demonstrating Set<E>)

Program : DemoSet.java

```
package p2;
import java.util.*;
public class DemoSet
{
    @SuppressWarnings("removal")
    public static void main(String[]
args)
    {
        Scanner s = new
Scanner(System.in);
        try(s;){
          try {
              Set<Integer> ob = null;
              String nm = null;

System.out.println("****Choice****");

System.out.println("\t1.HashSet"
                      +
"\n\t2.LinkedHashSet"
                      + "\n\t3.TreeSet");
              System.out.println("Enter
the Choice:");
              int choice = s.nextInt();
              switch(choice)
              {
              case 1:
```

```
                    ob = new
HashSet<Integer>();
                    nm = "HashSet";
                    break;
                case 2:
                    ob = new
LinkedHashSet<Integer>();
                    nm = "LinkedHashSet";
                    break;
                case 3:
                    ob = new
TreeSet<Integer>();
                    nm = "TreeSet";
                    break;
                default:

System.out.println("Invalid
Choice...");
                    System.exit(0);
                }//end of switch
                System.out.println("----
Add elements to "+nm+"---");
                System.out.println("Enter
the number of eles:");
                int n = s.nextInt();
                System.out.println("Enter
"+n+" Integer eles:");
                for(int i=1;i<=n;i++)
```

```
                {
                    ob.add(new
Integer(s.nextInt()));
                }//end of loop

System.out.println("****Display
Set<E>****");

System.out.println(ob.toString());
            }catch(Exception e)
{e.printStackTrace();}
        }//end of try with resource
    }
}
```

==========

Assignment:

wap to read n Integer elements to Set<E> object and display only Prime Numbers?

==========