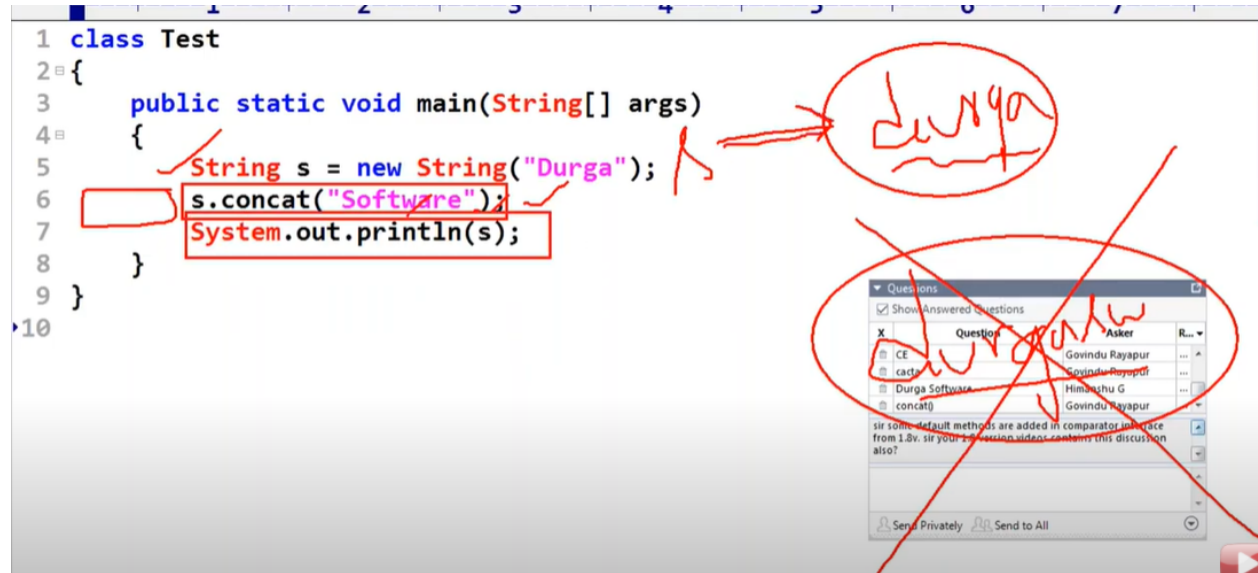


```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         String s = new String("Durga");
6         s.concat("Software");
7         System.out.println(s);
8     }
9 }
10
```



```
D:\durgaclasses>javac Test.java
```

```
D:\durgaclasses>java Test
Durga
```

```
D:\durgaclasses>
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         StringBuffer sb = new StringBuffer("Durga");
6         sb.append("Software");
7         System.out.println(sb);
8     }
9 }
10
```

It will give DurgaSoftware because it's mutable and it can edit existing data

But for string if we want concat we have to assign the reference after concat operation

Different ways to create String object

```
String s = new String();  
String s = new String(String s1);  
String s = new String("Durga");  
String s = new String(StringBuffer sb);  
  
String s = new String(char[] ch)  
String s = new String(byte[] b)
```

Example

```
String s = new String();  
String s = new String(String literal);  
String s = new String(StringBuffer sb);  
String s = new String(char[] ch);
```

```
char[] ch = {'a', 'b', 'c', 'd'};  
String s = new String(ch);  
System.out.println(s); //abcd
```

```
String s = new String(byte[] b);  
  
byte[] b = {100, 101, 102, 103};  
String s = new String(b);  
System.out.println(s); //defg
```

Q

```
public char charAt(int index);
```

```
String s = "durga";
```

```
System.out.println(s.charAt(3)); g
```

```
System.out.println(s.charAt(30));
```

```
RE: StringIndexOutOfBoundsException
```

-----

Q.

The overloaded + and += operators also meant for concatenation purpose only

```
String s = "durga";
```

```
s = s.concat("software");
```

```
//s = s+"software";
```

```
//s += "software";
```

```
System.out.println(s); //durgasoftware
```

-----

\*Equals vs IgnoreCaseEquals

**equals()=>Content comp**

```
s1="Durga";
```

```
s2="durga";
```

```
sop(s1.equals(s2))false
```

```
sop(s1.equalsIgnoreCase(s2))true
```

Q

```
public boolean equals(Object o)
```

To perform content comparison where **case** is important.  
This is overriding version of **Object class** equals() method

```
public boolean equalsIgnoreCase(String s)
```

To perform content comparison where **case** is not important.

```
String s = "java";  
System.out.println(s.equals("JAVA")); //false  
System.out.println(s.equalsIgnoreCase("JAVA")); //true
```

Q

```
public String substring(int begin);  
    return substring from begin index to end of the String
```

```
public String substring(int begin, int end);  
    return substring from begin index to end-1 index
```

```
String s = "abcdefg";  
System.out.println(s.substring(3)); //defg  
System.out.println(s.substring(2,5)); //cde
```

Q

```
=====
public int length()
    String s = "java";
    System.out.println(s.length);
        CE: cannot find symbol
            symbol: variable length
            location: java.lang.String
    System.out.println(s.length()); 4
-----
```

Q

```
public String replace(char old, char new)

String s = "ababa";
System.out.println(s.replace('a', 'b')); // bbbba
-----
```

Q

```
public int indexOf(char ch);
public int lastIndexOf(char ch);

s="ababab";
sop(s.indexOf('a'));0
sop(s.LastIndexOf('a'));4
```

I

I