~~Strings~~:

array of char:

'k' "ko" 'ko' "" "k" ""

k. Nothing. n n.

123 '123' "123".

Int nothing String.

StringBuilder or StringBuffer. (we use this to create)

Strings.

String (for creating) Immutable Strings. (non-changeable)

mutable strings. (changeable)

your name.
your DoB.
your email
PAN
address

your :- age.
Pin
Pass.

(1) Strings are the series of character. enclosed with "".

(2) Two types of String in Java.

Immutable :- which coe can not change.

mutable :- which coe can change or (upd

③ Strings are treated as objects in Java

Immutable string :- after creation we cannot update / change.

These strings can be created using the in built class String.

Different way to create Immutable String

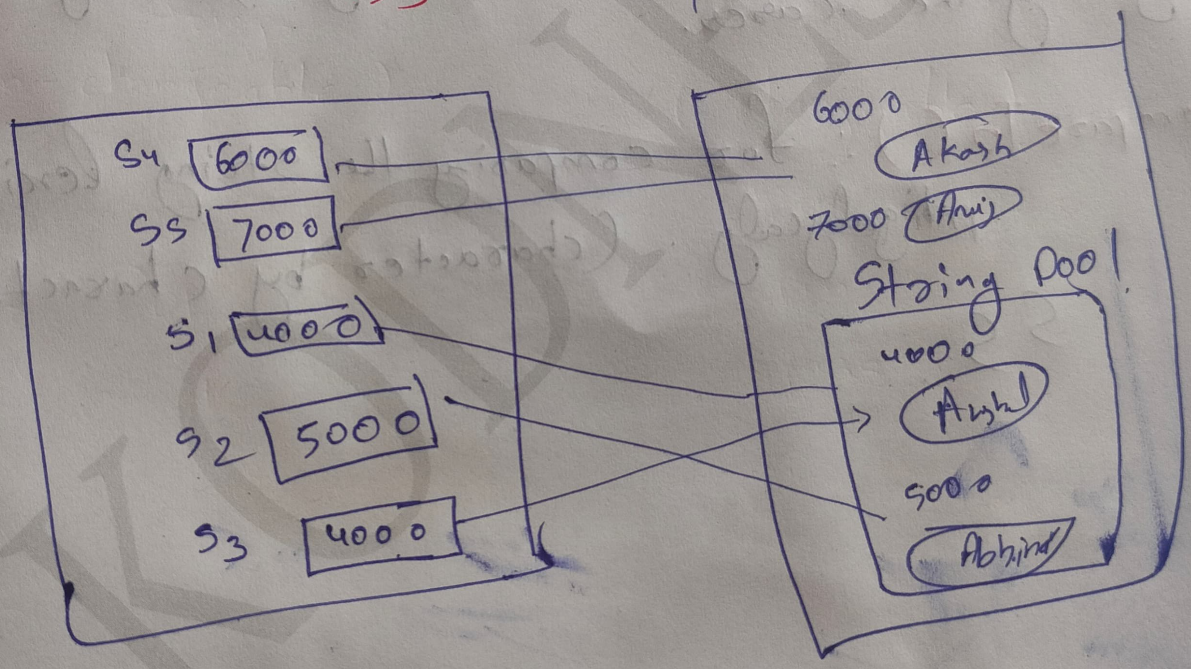
1) String s1 = "Anshul";

2) String s2 = new String ("Anshul");

③ char [] arr = {'k', 'o', 'd', 'n', 'e', 't'};
String s3 = new String (arr);

★ we write new keyword or not object is created automatically in Java. (because String is treated as object)

```
Class Program {
    Psvm (String[] args)
    {
        String s1 = "Anshul";
        SOP (s1);
        String s2 = "Abhinav";
        SOP (s2)
        String s3 = "Anuj";
        SOP (s3);
        String s4 = new String("Akash");
        SOP (s4)
        s5 = s4.s ("Anuj");
        SOP (s5)
        s6 = ns ("Anshul");
    }
```



Different ways to compare string

① == :- This operator is used for comparing the references of string object

② equals() :- This method is an inbuild method which is used for comparing the values of string object.

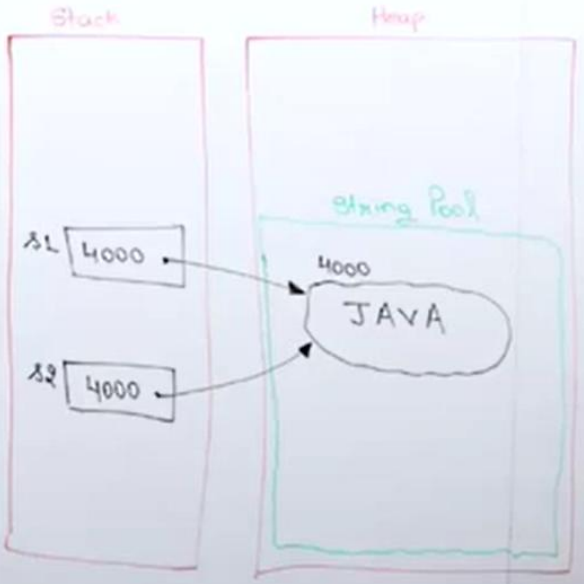
③ equalsIgnoreCase() :- This is used for comparing the values of string object by ignoring the cases.

④ compareTo() :- for compaoing the string lexicogra phically. (character by character)

SS

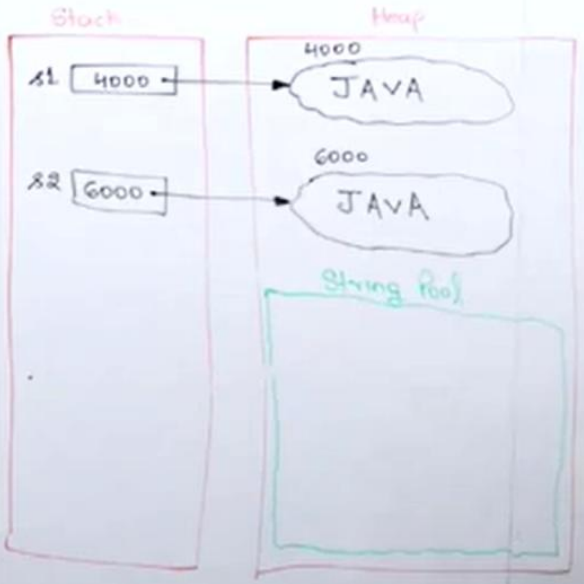
```
class Program
{
  p s v m(String [] args)
  {
    String s1 = "JAVA";
    String s2 = "JAVA";
    if (s1 == s2)
    {
      Sop("References are same");
    }
    else
    {
      S.o.p("References are not same");
    }
```

```
if(s1.equals(s2))
{
  Sop("Values are same");
}
else
{
  Sop("Values are not same");
}
}
```

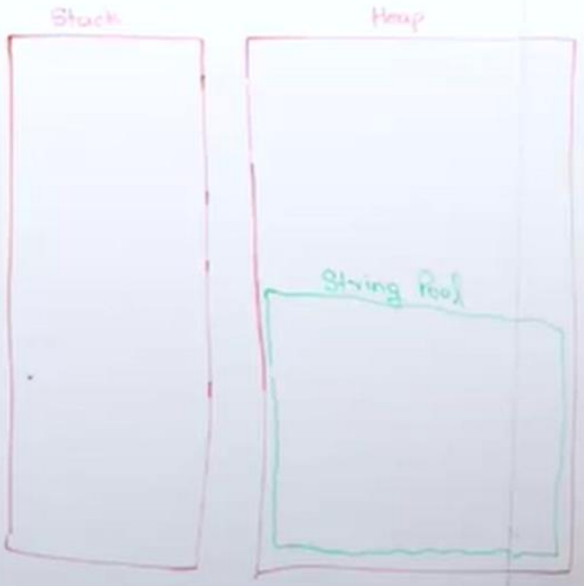


```
class Program
{
  p s v m(String [] args)
  {
    String s1 = new String("JAVA");
    String s2 = new String("JAVA");
    if (s1 == s2)        // 4000 == 6000
    {
      Sop("References are same");
    }
    else
    {
      Sop("References are not same");
    }
```

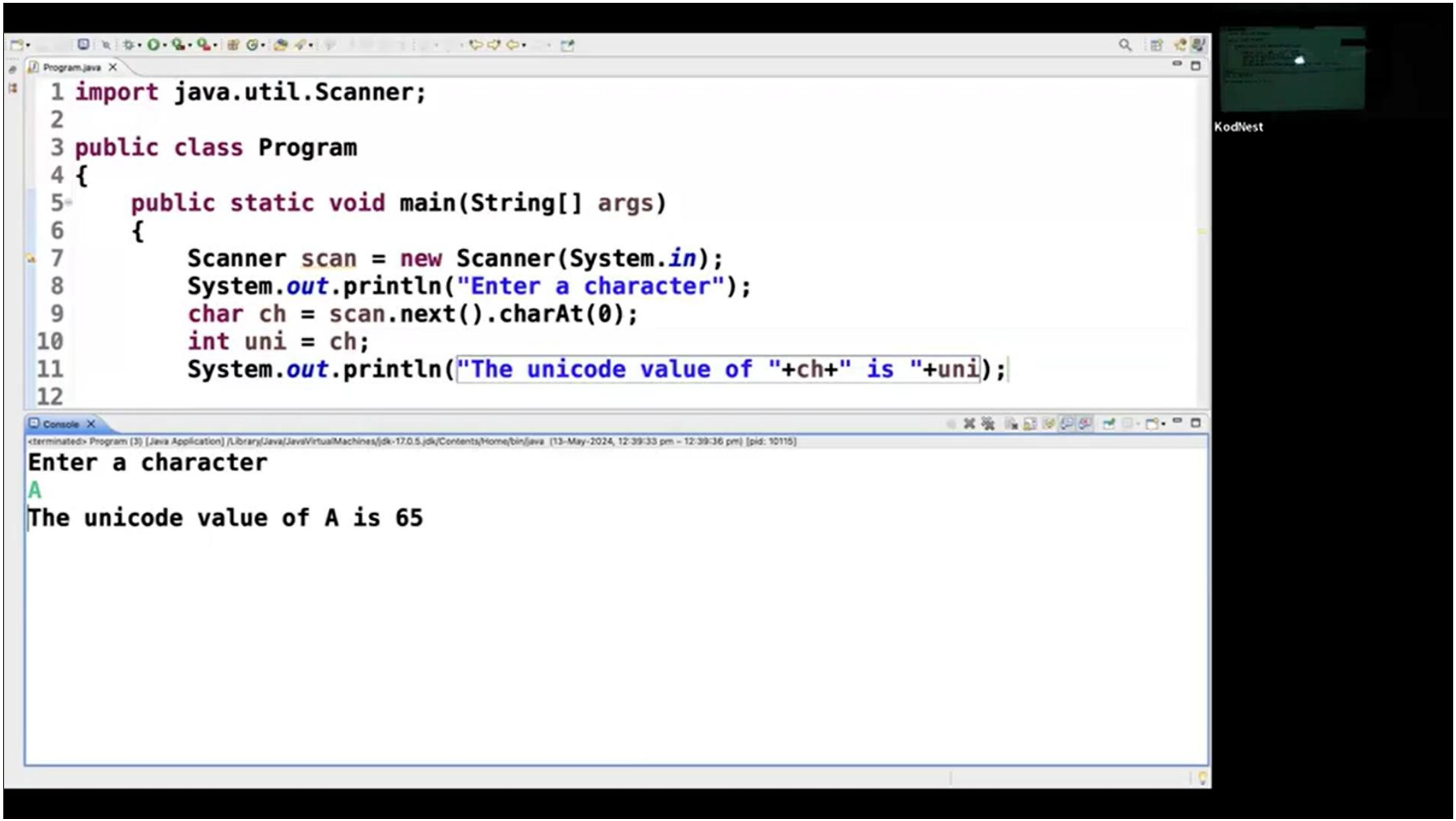
```
    if(s1.equals(s2))    // "JAVA".equals("JAVA")
    {
      Sop("Values are same");
    }
    else
    {
      Sop("Values are not same");
    }
  }
}
```

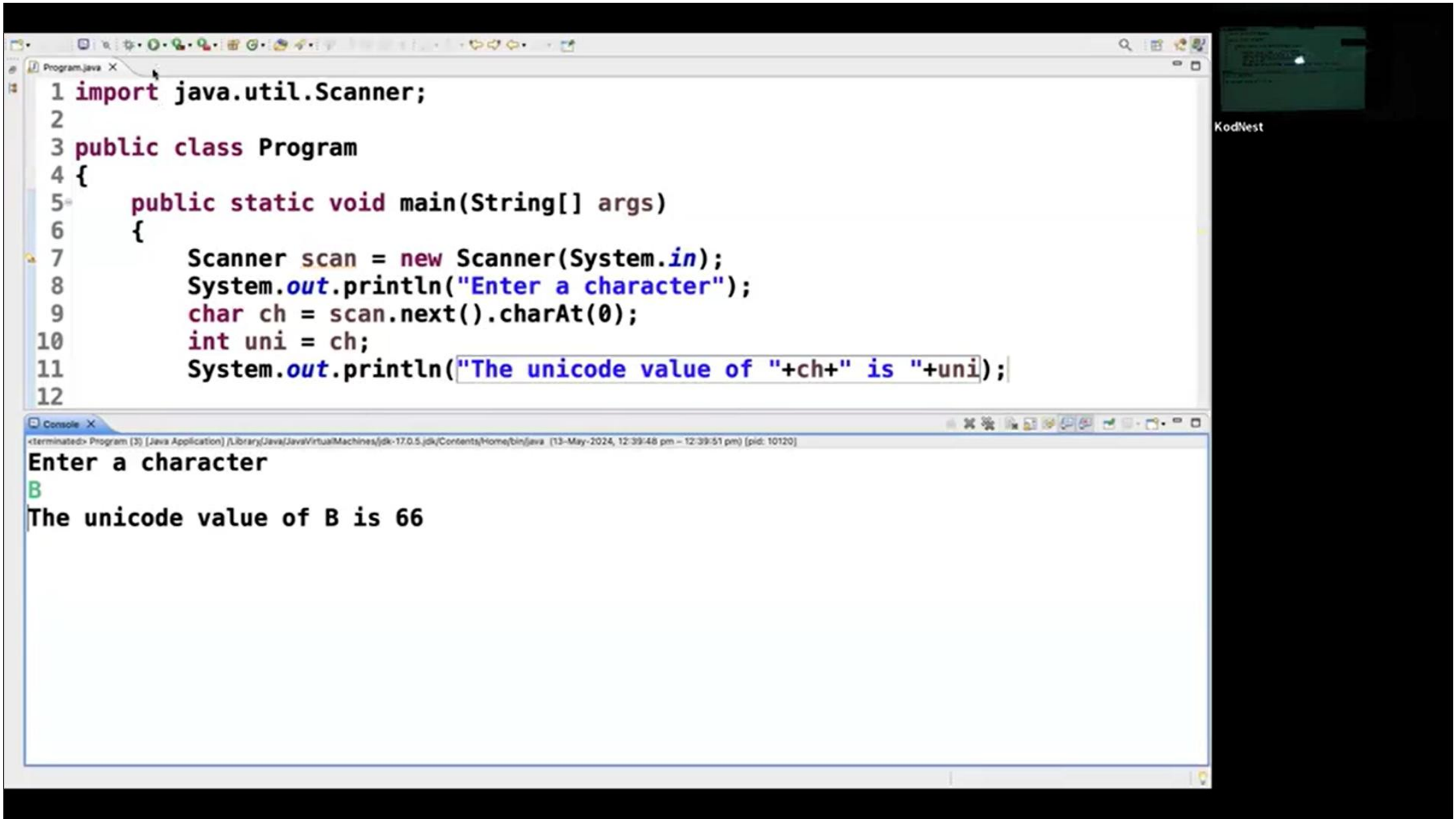


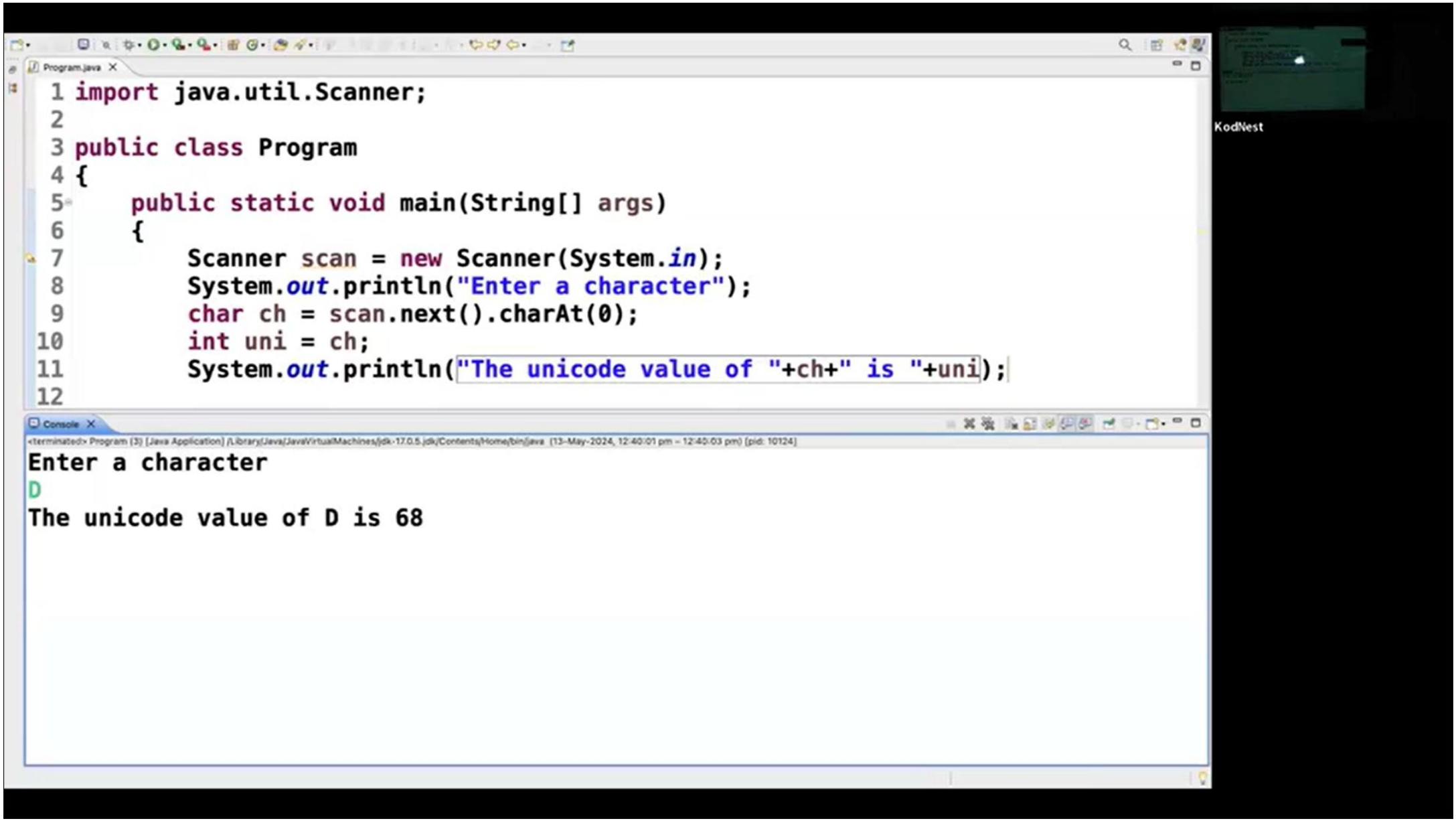
```
class Program
{
  p s v m(String [] args)
  {
    String s1 = "JAVA";
    String s2 = new String ("JAVA");
    if (s1 == s2)
    {
      Sop("References are same");
    }
    else
    {
      S.o.p("References are not same");
    }
    if(s1.equals(s2))
    {
      Sop("Values are same");
    }
    else
    {
      Sop("Values are not same");
    }
  }
}
```

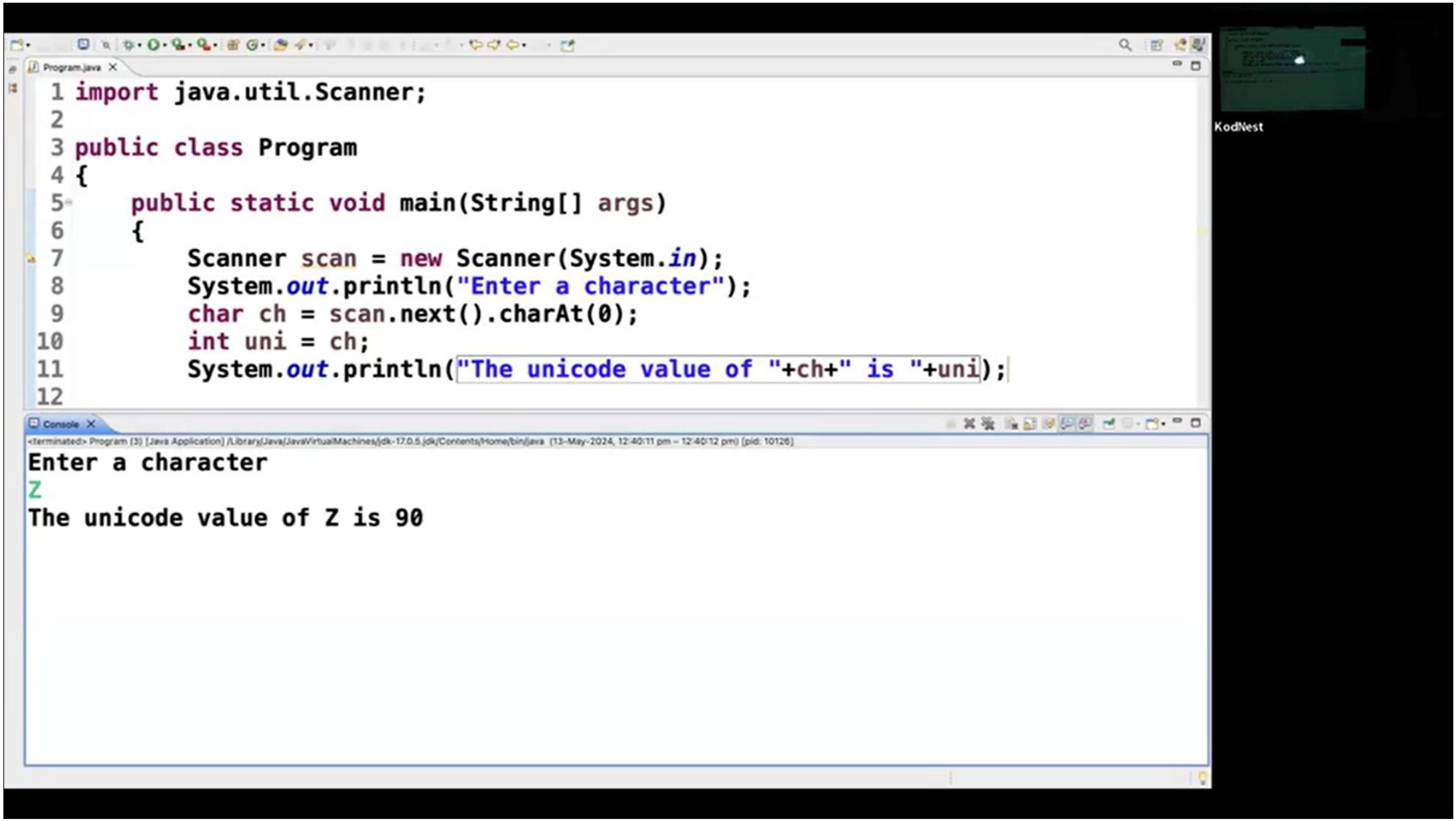


```
class Program
{
  p s v m(String [] args)
  {
    String s1 = "Java SQL Python";
    String s2 = "JAVA SQL PYTHON";
    if (s1 == s2)
    {
      S.o.p("References are same");
    }
    else
    {
      S.o.p("References are not same");
    }
    if(s1.equals(s2))
    {
      Sop("Values are same");
    }
    else
    {
      Sop("Values are not same");
    }
    if(s1.equalsIgnoreCase(s2))
    {
      S.o.p("Values are same after ignoring cases");
    }
    else
    {
      S.o.p("Values are not same after ignoring cases");
    }
  }
}
```


Program.java
1 import java.util.Scanner;
2
3 public class Program
4 {
5 public static void main(String[] args)
6 {
7 Scanner scan = new Scanner(System.in);
8 System.out.println("Enter a character");
9 char ch = scan.next().charAt(0);
10 int uni = ch;
11 System.out.println("The unicode value of "+ch+" is "+uni);
12
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (13-May-2024, 12:39:33 pm – 12:39:36 pm) [pid: 10115]
Enter a character
A
The unicode value of A is 65
KodNest

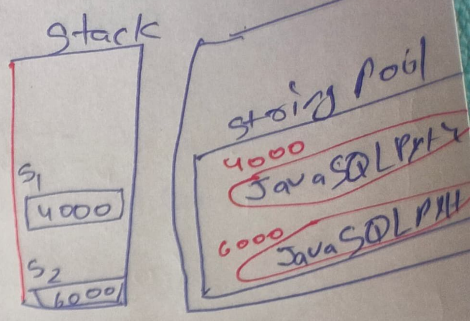

Program.java
1 import java.util.Scanner;
2
3 public class Program
4 {
5 public static void main(String[] args)
6 {
7 Scanner scan = new Scanner(System.in);
8 System.out.println("Enter a character");
9 char ch = scan.next().charAt(0);
10 int uni = ch;
11 System.out.println("The unicode value of "+ch+" is "+uni);
12
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (13-May-2024, 12:39:48 pm – 12:39:51 pm) [pid: 10120]
Enter a character
B
The unicode value of B is 66
KodNest


Program.java
1 import java.util.Scanner;
2
3 public class Program
4 {
5 public static void main(String[] args)
6 {
7 Scanner scan = new Scanner(System.in);
8 System.out.println("Enter a character");
9 char ch = scan.next().charAt(0);
10 int uni = ch;
11 System.out.println("The unicode value of "+ch+" is "+uni);
12
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (13-May-2024, 12:40:01 pm – 12:40:03 pm) (pid: 10124)
Enter a character
D
The unicode value of D is 68
KodNest


Program.java
1 import java.util.Scanner;
2
3 public class Program
4 {
5 public static void main(String[] args)
6 {
7 Scanner scan = new Scanner(System.in);
8 System.out.println("Enter a character");
9 char ch = scan.next().charAt(0);
10 int uni = ch;
11 System.out.println("The unicode value of "+ch+" is "+uni);
12
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (13-May-2024, 12:40:11 pm – 12:40:12 pm) (pid: 10126)
Enter a character
Z
The unicode value of Z is 90
KodNest

ex (4)

```
s1 = "Java SQL Python";
s2 = "JAVA SQL PYTHON";
```




```
if.(s1.equalsIgnoreCase(s2)){
    sout.("Values are same");
```

This will check only the char. is the ~~same~~ not upper and lower case.

Q. print unicode value. (A-65, Z-90) (a-97, z-122)

V.S

(4) ~~#~~

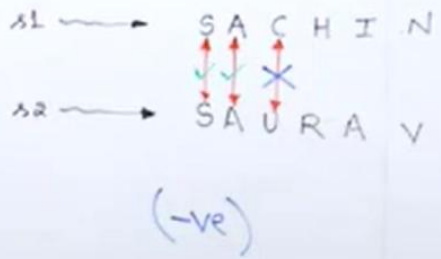
Compare To ()

(-ve) (0) (+ve)

{ s1<s2 | | both strings are equal | | s1>s2 |

SS

```
class Program
{
  psvm(String [] args)
  {
    String s1 = "SACHIN";
    String s2 = "SAURAV",
    int res = s1.compareTo(s2);
    if(res > 0)
    {
    } S.O.p("s1 is greater than s2");
    else if(res < 0)
    {
    } S.O.p("s1 is smaller than s2");
    else
    {
      S.o.p("s1 and s2 are equal");
    }
  }
}
```



```
class Program
{
  psvm(String [] args)
  {
    String s1 = "SACHIN";
    String s2 = "SAURAV",
    int res = s1.compareTo(s2);
    if(res > 0)
    {
    } S.O.p("s1 is greater than s2");
    else if(res < 0)
    {
    } S.O.p("s1 is smaller than s2");
    else
    {
      S.O.p("s1 and s2 are equal");
    }
  }
}
```

1) s1 → RAMA , s2 → RAMU

2) s1 → GEETA , s2 → GUPTA

3) s1 → RAJ , s2 → RAJENDRA

4) s1 → RAJU , s2 → RAJAN

5) s1 → JAVA , s2 → java

(-ve)

Proof Strings are immutable.

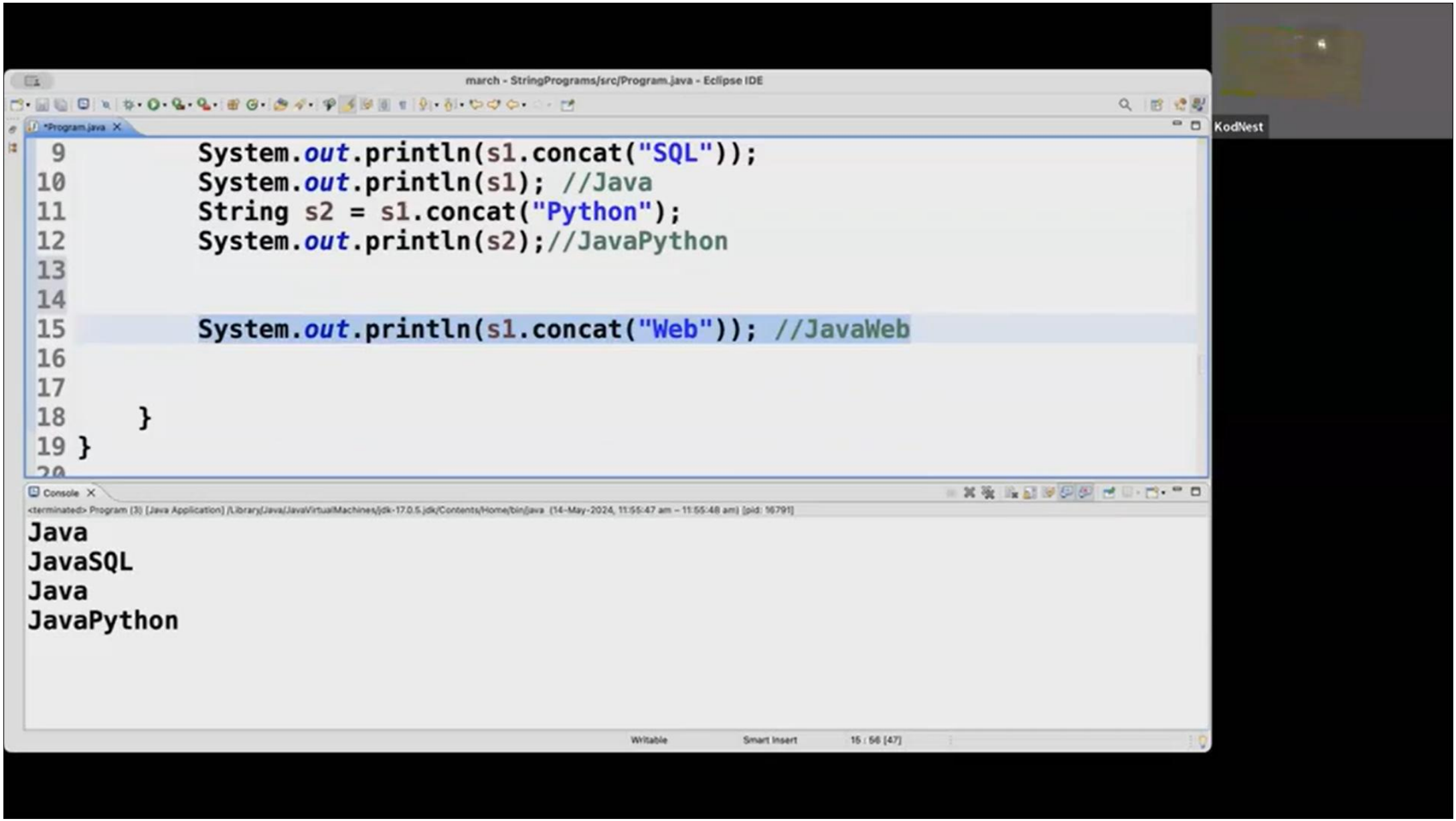
Concate() + - add in the end.

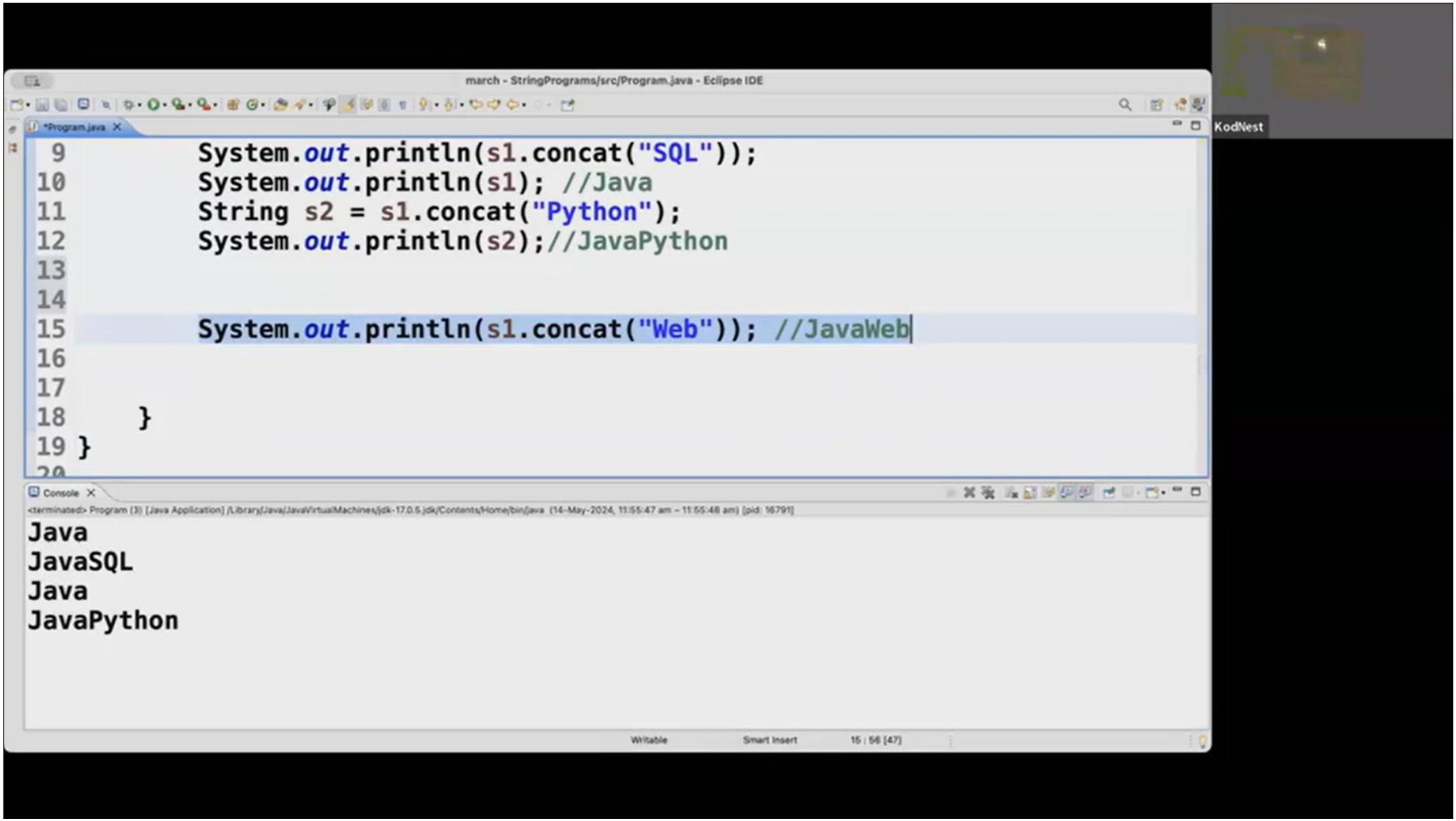
```
S1 = "Anshul";
S1.concate("Java");
So.P(S1) // Anshul ~~Java~~ ~~Bina~~. It will not update. the string.

String S2 = S1.concat("SQ")
Sop(S2); // Anshul SQ
```

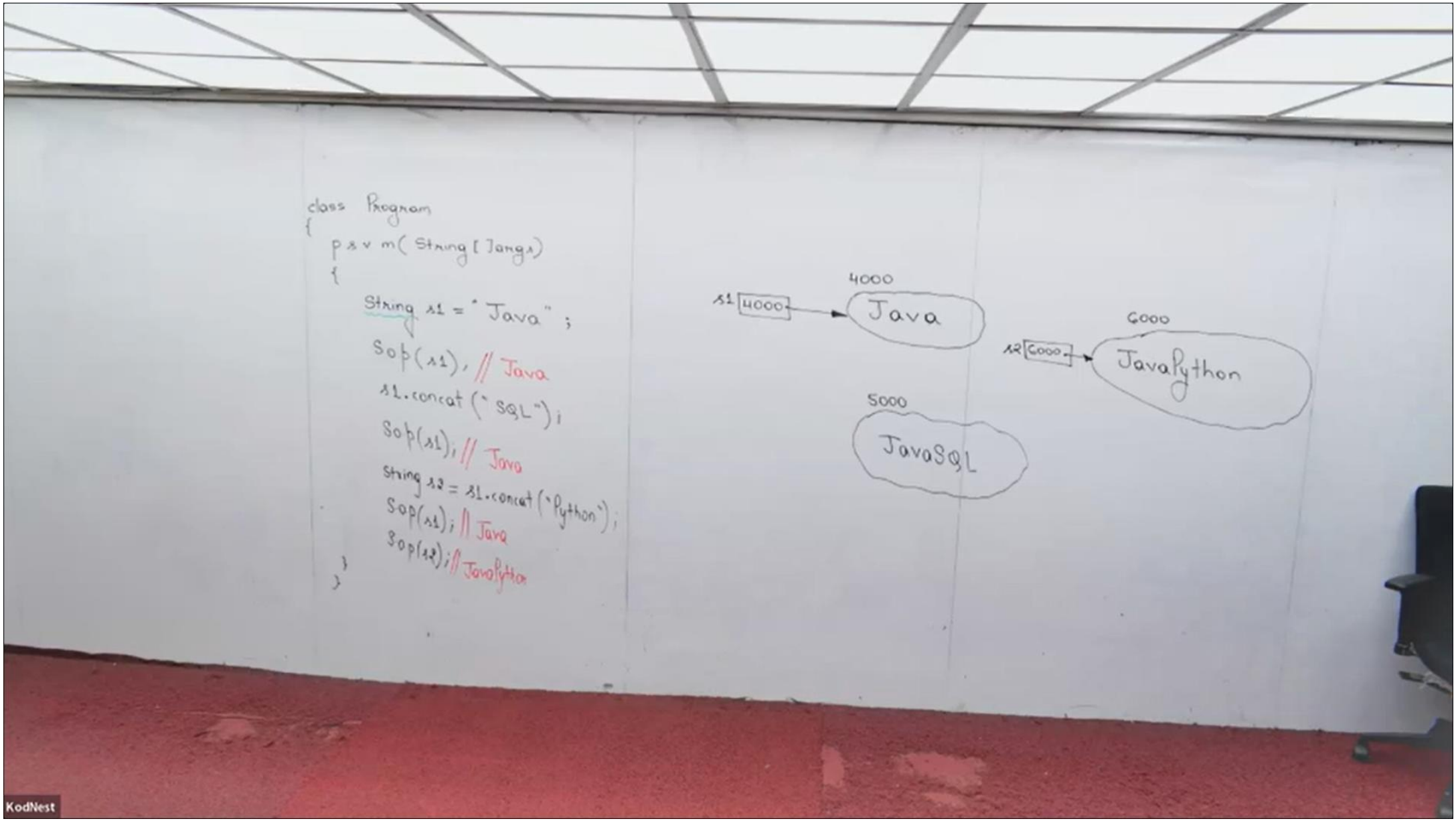
Note:- Strings created using string class are considered as immutable string which means. theirs string values cannot be changet. but if they are modified then a new object will be created and ooigiral string will remain same.

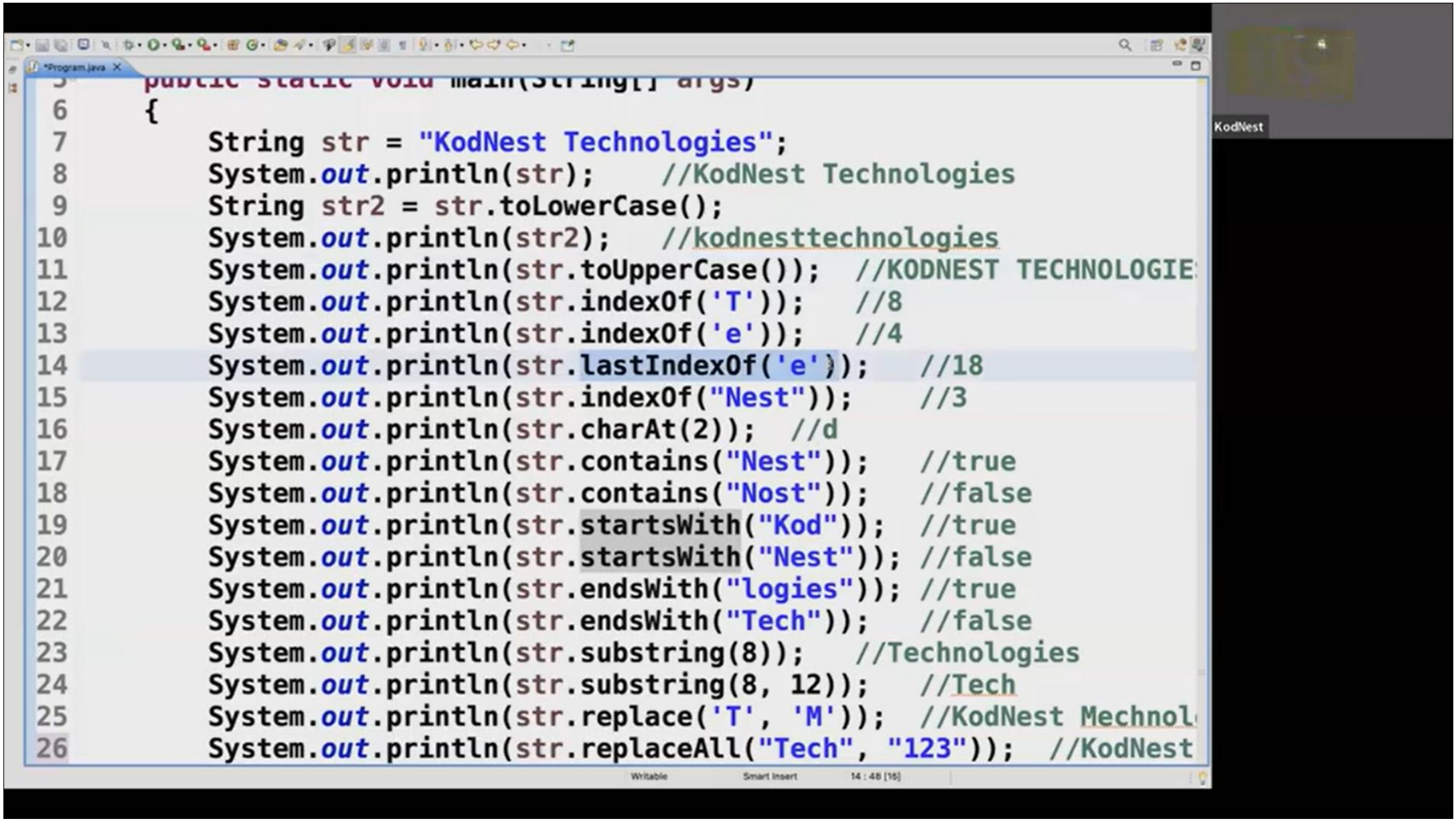
2). Concate method is used for concatinating the strings.

3) length():- S.toLowerCase();
S.toUpperCase();
S.indexOf('n');
S.lastIndexOf('4');


march - StringPrograms/src/Program.java - Eclipse IDE
*Program.java
9 System.out.println(s1.concat("SQL"));
10 System.out.println(s1); //Java
11 String s2 = s1.concat("Python");
12 System.out.println(s2);//JavaPython
13
14
15 System.out.println(s1.concat("Web")); //JavaWeb
16
17
18 }
19 }
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (14-May-2024, 11:55:47 am – 11:55:48 am) [pid: 16791]
Java
JavaSQL
Java
JavaPython
Writable
Smart Insert
15 : 56 [47]
KodNest


march - StringPrograms/src/Program.java - Eclipse IDE
*Program.java
KodNest
System.out.println(s1.concat("SQL"));
System.out.println(s1); //Java
String s2 = s1.concat("Python");
System.out.println(s2);//JavaPython
System.out.println(s1.concat("Web")); //JavaWeb
}
}
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (14-May-2024, 11:55:47 am – 11:55:48 am) [pid: 16791]
Java
JavaSQL
Java
JavaPython
Writable
Smart Insert
15 : 56 [47]


class Program
{
psvm(String [ ]args)
{
String s1 = "Java";
Sop(s1); // Java
s1.concat("SQL");
Sop(s1); // Java
String s2 = s1.concat("Python");
Sop(s1); // Java
Sop(s2); // JavaPython
}
}
4000
s1 4000
Java
5000
JavaSQL
6000
s2 6000
JavaPython
KodNest


*Program.java
public static void main(String[] args)
{
String str = "KodNest Technologies";
System.out.println(str); //KodNest Technologies
String str2 = str.toLowerCase();
System.out.println(str2); //kodnesttechnologies
System.out.println(str.toUpperCase()); //KODNEST TECHNOLOGIE
System.out.println(str.indexOf('T')); //8
System.out.println(str.indexOf('e')); //4
System.out.println(str.lastIndexOf('e')); //18
System.out.println(str.indexOf("Nest")); //3
System.out.println(str.charAt(2)); //d
System.out.println(str.contains("Nest")); //true
System.out.println(str.contains("Nost")); //false
System.out.println(str.startsWith("Kod")); //true
System.out.println(str.startsWith("Nest")); //false
System.out.println(str.endsWith("logies")); //true
System.out.println(str.endsWith("Tech")); //false
System.out.println(str.substring(8)); //Technologies
System.out.println(str.substring(8, 12)); //Tech
System.out.println(str.replace('T', 'M')); //KodNest Mechnol
System.out.println(str.replaceAll("Tech", "123")); //KodNest
Writable
Smart Insert
14 : 48 [16]
KodNest

s.substring(6); //Anshul
s.substring(0,4) //Ansh

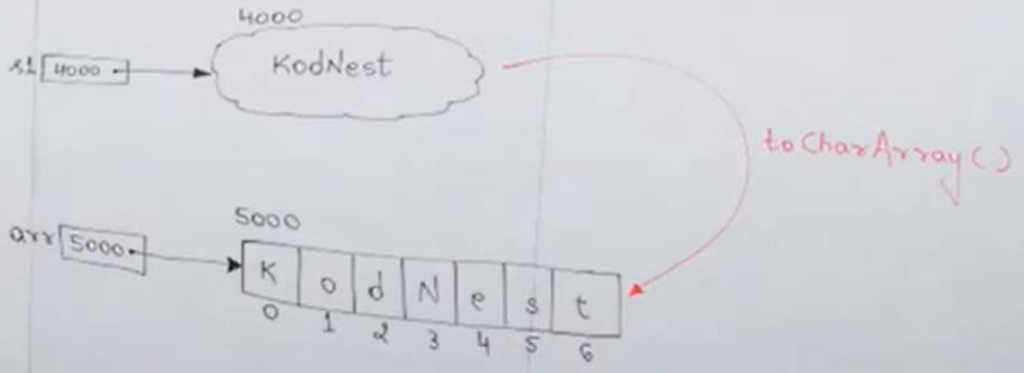
Converting a string into char type array.

char[] arr = s1.toCharArray();

intern method.

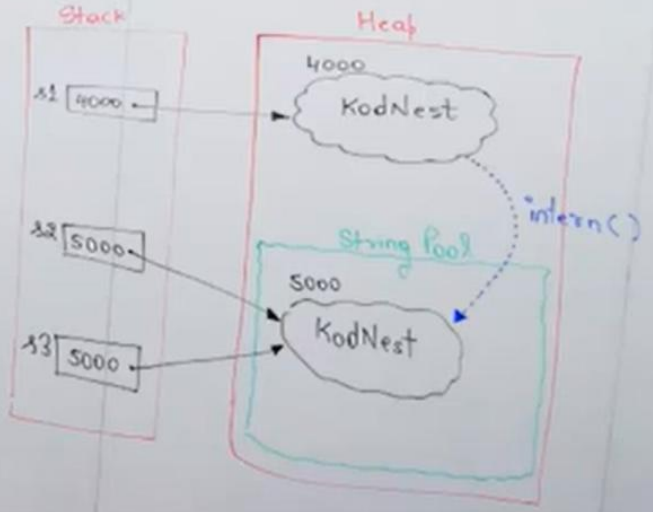
↳ ss

```
class Program
{
  p s v m (String [ ]args)
  {
    String s1 = " KodNest ";
    So p(s1);  // KodNest

    char[]arr = s1.toCharArray();

    for(char x : arr)
    {
    } So print(x+" ");
  }
}
```



```
class Program
{
  psvm(String[]args)
  {
     String s1 = new String("KodNest");
      Sop(s1);

      String s2 = s1.intern();
      Sop(s2);

      String s3 = "KodNest";
```

```
if(s2 == s3)
{
  Sop("Ref are same");
}
else
{
  Sop("Ref are not same");
}
}
```



String Builder.

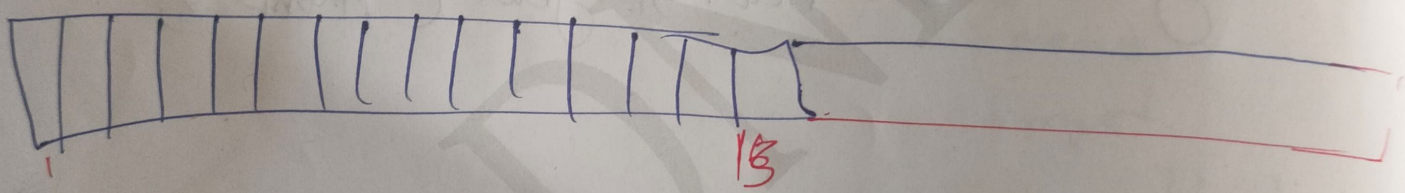
StringBuilder sbuild = new StringBuilder("Ankur");

sout(sbuild);

String Buffer

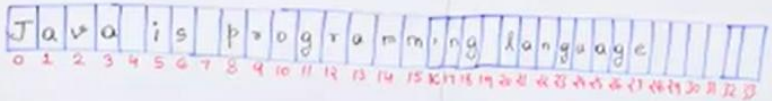
StringBuffer s = new StringBuffer("Ankur");

JVM will creat. 16 location. agr string ka size 16 se jada hai to. ((~~16+16~~)x 16x2+2).
in the end.



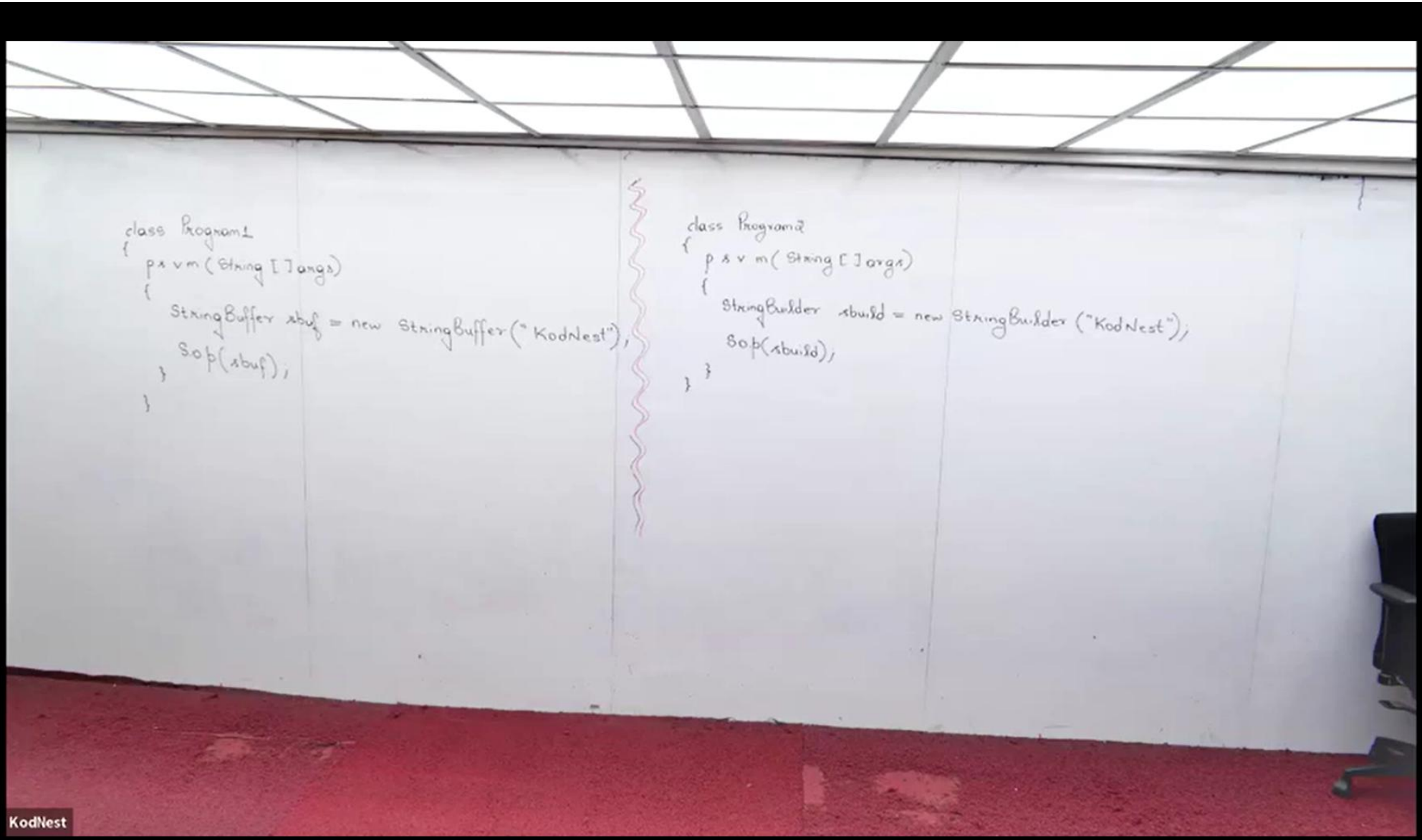
* while creating a object of SB. SB. in the constructor then its capacity will be 16. if parameter are passed then its capacity will be 16+length of string

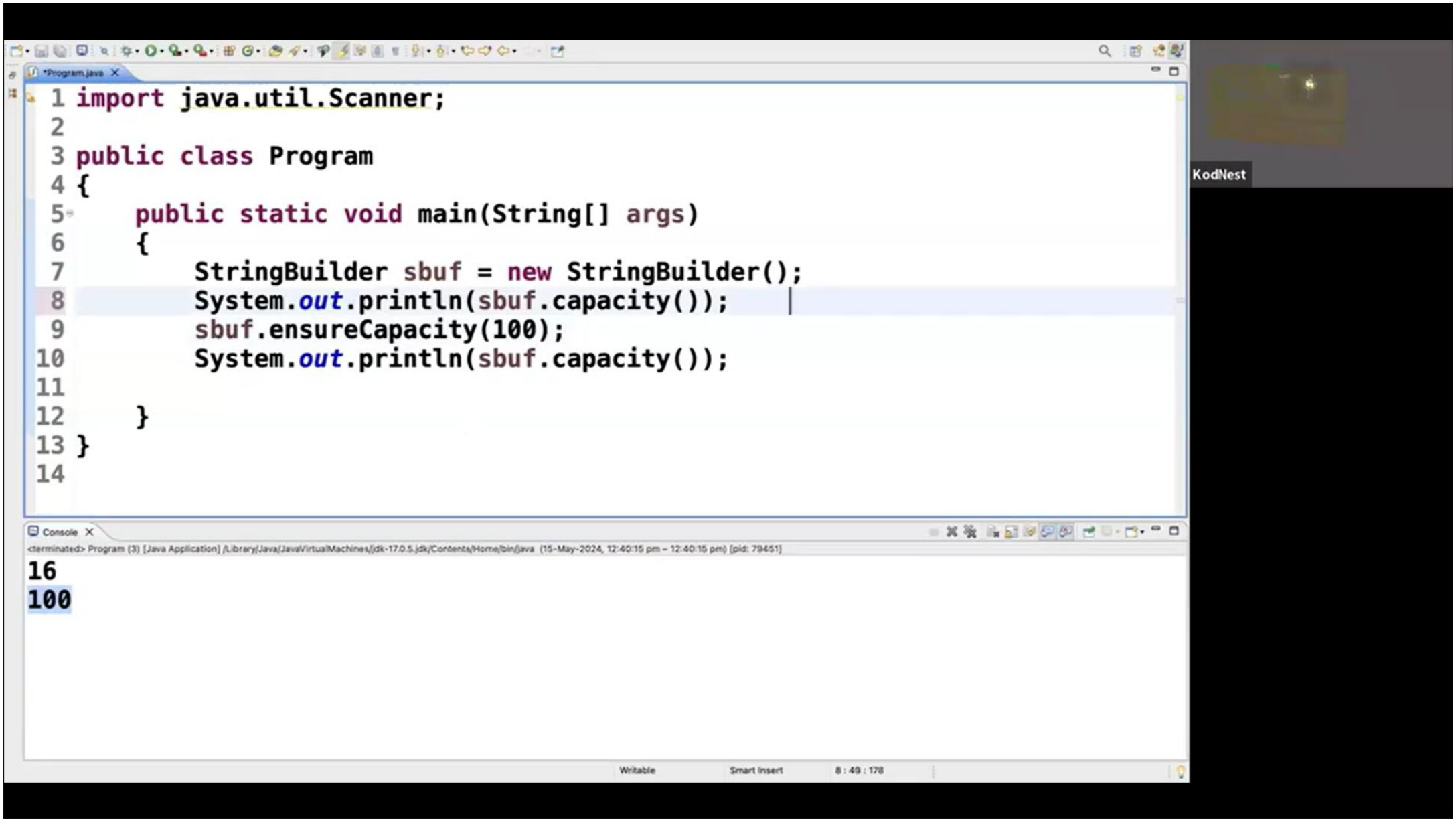
```
class Program3
{
  psvm(String[]args)
  {
    StringBuffer sbuf = new StringBuffer();
    Sop(sbuf.capacity()); // 16
    sbuf.append("Java");
    Sop(sbuf); // Java
    Sop(sbuf.capacity()); // 16
    sbuf.append(" is programming language");
    Sop(sbuf); // Java is programming language
    Sop(sbuf.capacity()); // 34
  }
}
```

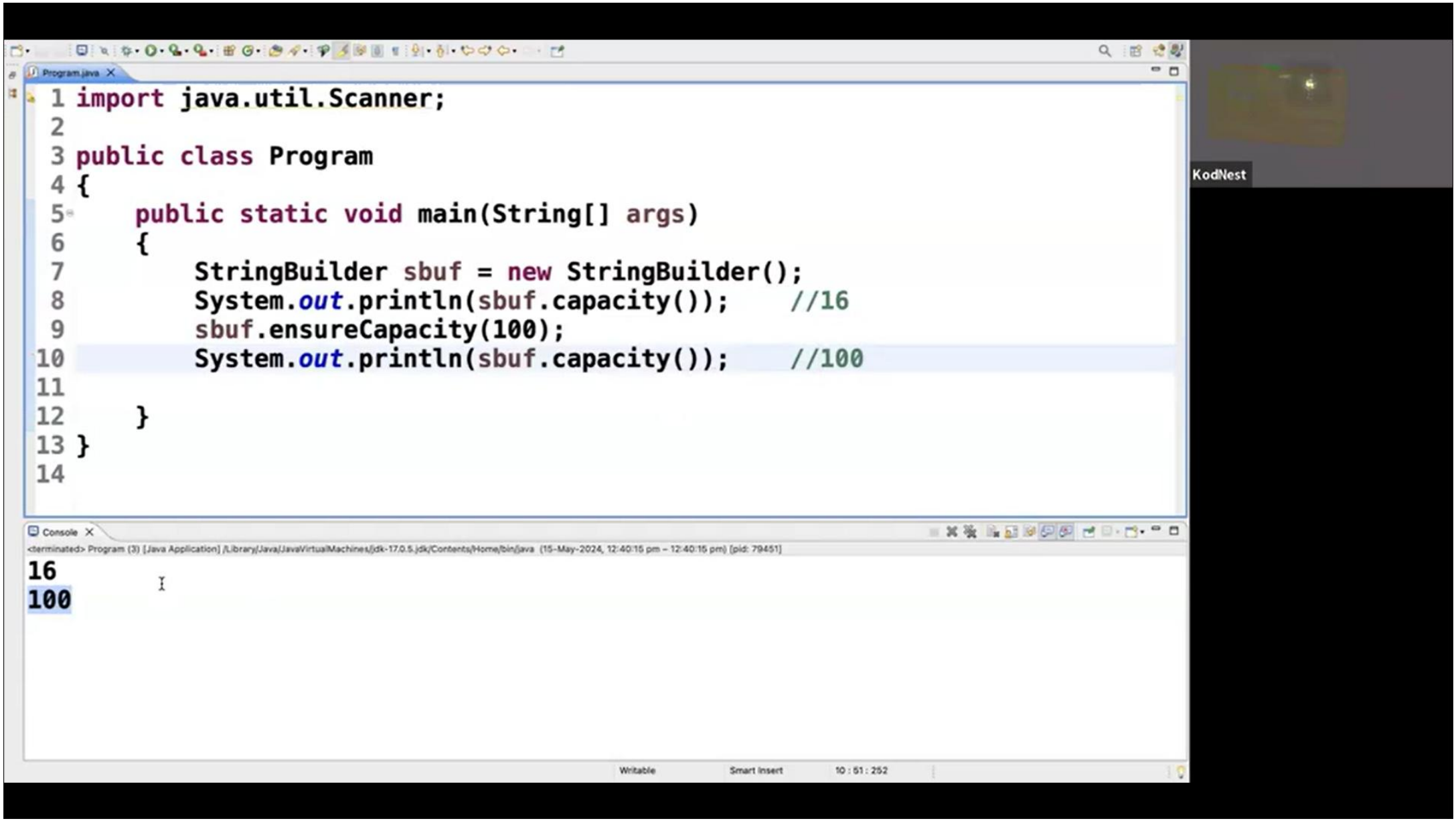


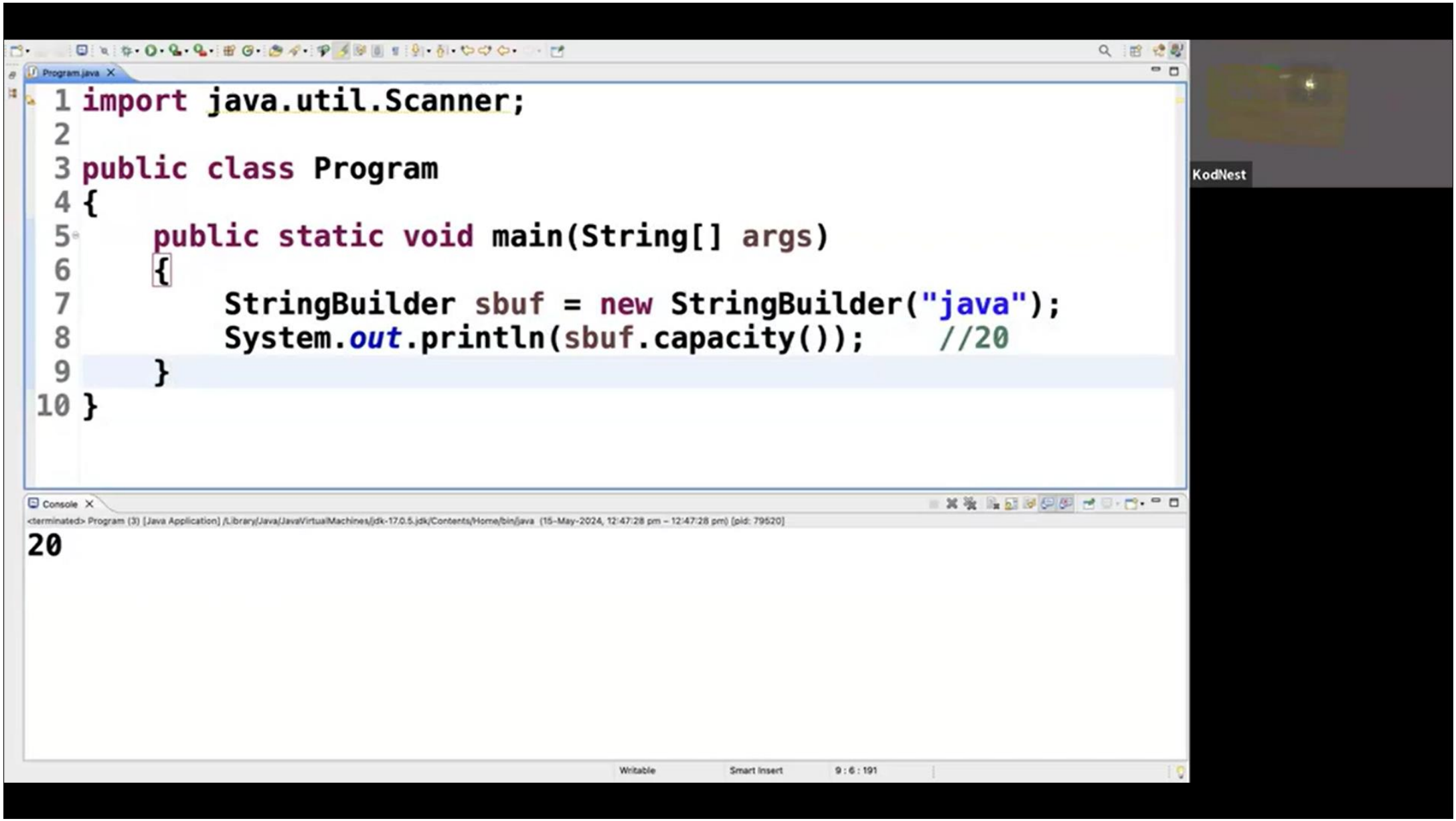
new capacity = old capacity * 2 + 2,

- 16 * 2 + 2
- = 32 + 2
- = 34


class Program1
{
ps vm (String [] args)
{
StringBuffer sbuf = new StringBuffer ("KodNest");
Sop(sbuf);
}
}
class Program2
{
ps v m(String [] args)
{
StringBuilder sbuild = new StringBuilder ("KodNest");
Sop(sbuild);
}
}
KodNest


*Program.java
1 import java.util.Scanner;
2
3 public class Program
4 {
5 public static void main(String[] args)
6 {
7 StringBuilder sbuf = new StringBuilder();
8 System.out.println(sbuf.capacity());
9 sbuf.ensureCapacity(100);
10 System.out.println(sbuf.capacity());
11
12 }
13 }
14
KodNest
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (15-May-2024, 12:40:15 pm – 12:40:15 pm) [pid: 79451]
16
100
Writable
Smart Insert
8 : 49 : 178


Program.java
1 import java.util.Scanner;
2
3 public class Program
4 {
5 public static void main(String[] args)
6 {
7 StringBuilder sbuf = new StringBuilder();
8 System.out.println(sbuf.capacity()); //16
9 sbuf.ensureCapacity(100);
10 System.out.println(sbuf.capacity()); //100
11
12 }
13 }
14
KodNest
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (15-May-2024, 12:40:15 pm – 12:40:15 pm) [pid: 79451]
16
100
Writable
Smart Insert
10 : 51 : 252


Program.java
1 import java.util.Scanner;
2
3 public class Program
4 {
5 public static void main(String[] args)
6 {
7 StringBuilder sbuf = new StringBuilder("java");
8 System.out.println(sbuf.capacity()); //20
9 }
10 }
KodNest
Console
<terminated> Program (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.5.jdk/Contents/Home/bin/java (15-May-2024, 12:47:28 pm – 12:47:28 pm) (pid: 79520)
20
Writable
Smart Insert
9 : 6 : 191

StringBuffer.	StringBuilder.
1) It is in build class use to create mutable string	1) -//-
2) Default capacity is 16 chars	2) -//-
3) This class is syncronized.	3) This class is not syncro-
4) This class is thread safe.	4) This class is not thread safe
5) it is slow in comparition with Buider.	5) It is faster.
6) It is not suitable for multi threading	6) It is suitable for multi threading
7) It is available since JDK 1.0	7) It is available since JDK 1.5

Below 10 Logical Programs Explained In This Video

1. Reverse String
2. Remove Duplicate Chars
3. Reverse Each Word in String
4. Find Char Occurrence
5. Find Non Repeated Char
6. Replace Char with Occurrence
7. Find Longest Substring
8. Sort String Characters
9. Remove Whitespaces in String
10. Remove Special Characters