

Programming in Java
Prof. Debasis Samanta
Department of Computer Science Engineering
Indian Institute of Technology, Kharagpur

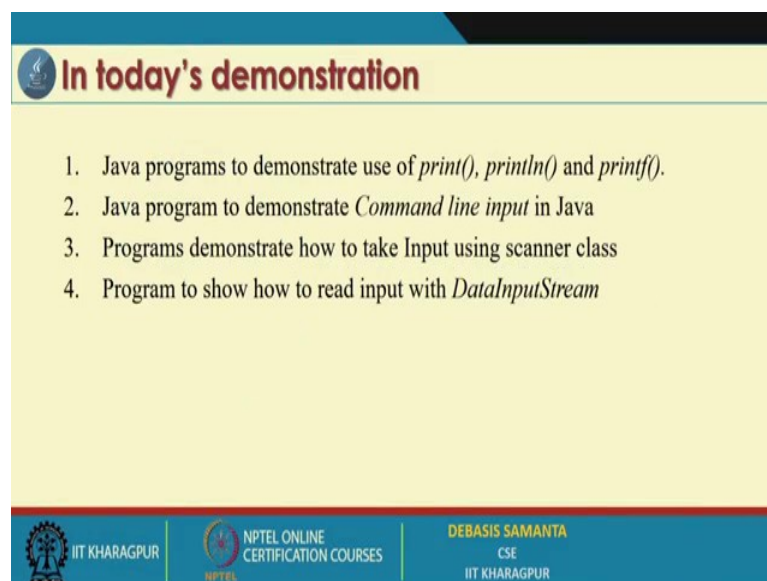
Lecture - 10
Demonstration –IV

In the last lecture we have learned about many detailed things about Java Programming the different return type and input, output and everything. So, today we will have with some we have some illustration about output and input in Java. And you know input and output are the two important activities in any Java application.

So, these two things are very important; so we will discuss about how the different way the output can be and also what are the different procedures; so that you can pass inputs to a Java program. Now, so far the output is concerned in this illustration in this demonstration we will discuss about output related to display on the screen that is called the standard output.

That mean how the different way output can be displayed on the screen. There again output can be in a different way so that output can be passed to a file, output can be passed to a network line, output can be passed to a database, which is situated in a remote location all these things will be discussed later on.

(Refer Slide Time: 01:37)



In today's demonstration

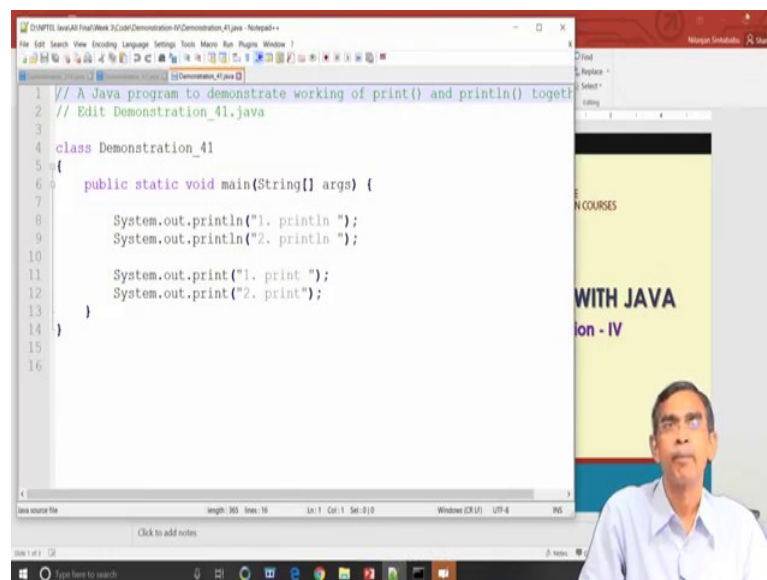
1. Java programs to demonstrate use of *print()*, *println()* and *printf()*.
2. Java program to demonstrate *Command line input* in Java
3. Programs demonstrate how to take Input using scanner class
4. Program to show how to read input with *DataInputStream*

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASIS SAMANTA CSE IIT KHARAGPUR

So, now, let us have a quick idea about how the input and output are possible in Java program. Now we have discussed the different methods which are there in the system class which defines a Java.lang package; namely the println(), and printf(). So, in our demonstration, we will discuss the uses of these three methods which are defined in system class as a System.out class actually more precisely.

And then after this discussion about output, we will have a discussion about the ways the input is possible to Java program. There are three ways by which a programmer can pass input to the Java are called the command line argument using the scanner class and using the data input stream class. So, let us have a demo about these four things now.

(Refer Slide Time: 02:32)



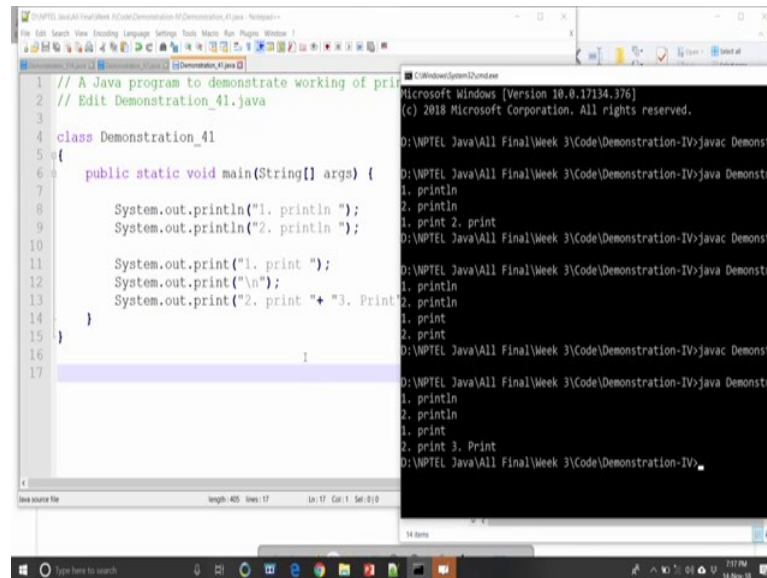
```
1 // A Java program to demonstrate working of print() and println() together
2 // Edit Demonstration_41.java
3
4 class Demonstration_41
5 {
6     public static void main(String[] args) {
7
8         System.out.println("1. println ");
9         System.out.println("2. println ");
10
11        System.out.print("1. print ");
12        System.out.print("2. print");
13    }
14 }
15
16
```

So, the first demo includes; how we can print the value on the screen. Now, this is a quick demo let us see this is a very simple program. Now we here we declare simple class the main class is here we do not have any other class in this case because it is a very simple example. So, the main class is defined here demonstration_for 41.

Now you can see the first two statement System.out.println() 1 2. Now you can guess what the print it will give. It will definitely print the same 1.println() and then 2.println(); however, it will give the print in the two different lines. So, println() means it will print the string or any value and then the cursor automatically goes to the next line.

So, for the first two print, the cursor automatically goes to the next line. Wherever for the third print, it will basically cursor goes to the next line. And for the fourth print because the next third print is only print cursor will not go to the next line. So, it will the cursor will be there in the third line itself. And similarly for this cursor will remain in the same line. Now let us have the demo.

(Refer Slide Time: 03:46)



The screenshot shows a Java IDE with two windows. The left window displays the source code for a class named `Demonstration_41`. The code includes comments and a `main` method with four print statements. The right window shows the output of the program, which is a black console with white text. The output shows the first two print statements on separate lines, followed by the third and fourth print statements on the same line, with the cursor blinking at the end of the fourth line.

```
// A Java program to demonstrate working of print
// Edit Demonstration_41.java

class Demonstration_41
{
    public static void main(String[] args) {
        System.out.println("1. println ");
        System.out.println("2. println ");

        System.out.print("1. print ");
        System.out.print("\n");
        System.out.print("2. print " + "3. Print ");
    }
}
```

Microsoft Windows [Version 10.0.17134.376]
(c) 2018 Microsoft Corporation. All rights reserved.
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_41.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_41
1. println
2. println
1. print 2. print
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_41.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_41
1. println
2. println
1. print
2. print
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_41.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_41
1. println
2. println
1. print 2. print
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>

So, that we can understand about the working of this print statement, now you see the first two print is basically in the two lines. And then second two prints are in the same line and then cursor basically in the current line actually; where the cursor is blinking here. Now so this is the idea about it, now let us see I am just adding one more statement we will see how it will work for you. After the third print writes this again system right and then enter and then `System.out.print()` and then `\n` print `\n \n`.

Now, \ it is like this not that yes. Now here if you see the third int statement that which is a fourth print statement rather which basically is a print and then it is just like in C programming language is a `\n` rather. So, `\n` is basically newline character that means it basically jumps the cursor to the next line. Now you can anticipate what output will be. So, the first two outputs will be in the two different lines, then the third output also in the third line and the `System.out.println()` cursor will automatically move to the next line. And then the last print statement will be the next line. So, now, let us see the output it is there we can see it.

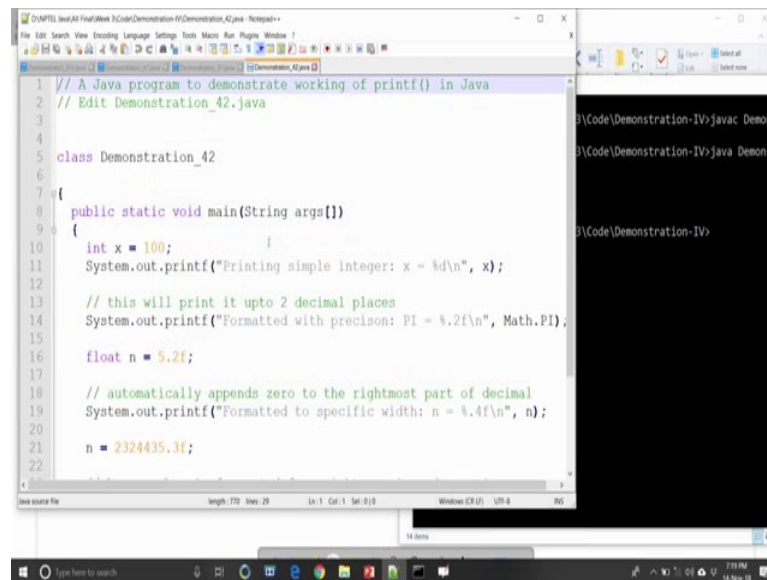
So, this is the effect of `\n` so is the new line character. So, it is the same as C programming or C++ programming concept actually. So, newline character is also Java except. So, this way you can see how using `print` and `println()` we can display the value on the screen. Now, this `println()` can also print more than two statement or two strings or values. For example, in the last statements say right print 2 plus, we can give a 3.`print`.

So, basically, we are printing 2 string and the 2 strings are basically mentioned in a separate way by means of plus this is called the concatenation of string ok. Now you can see it is like this now any value also can be printed. Now let us declare one value so `int x` at the beginning `int x` equals to 5 5 5. And the last statement let us plus `x` these.`x` plus `this.x`. Now let us see run compile this program now you can understand what you want to do is simple `x` you can get whether it will take or not. Let us see just we face one problem plus `x` fine compile. So, `this.x` is not required here right.

So, we can see it like this on right to we can print any value whatever it is there. Now, in this case, this is not resolved because here no resolution or no collision that is a this is not acceptable. So, we can just simply refer and here also we do not have to create an object because the main method here is static for any static method we in fact, access any value without any creating object. So, here the `x` value it is basically accessed here in the main method without creating an object. So, no `this.x` is required in this case. So, we can ok, now, here the little bit printing can be made more configured plus `x` we can use this space comma.

So, plus `x` before `++`. Now a little bit more please `cls`. Now we can run the same program again we have to compile it because you have to change the program. Ok run, now you can see. So, using all those things we can customize our format printing on the screen. So, it is under the control of the programmer. So, you have learn about `print` and `println()` in addition to this `print` and `println()` Java also allow to use the `printf()` function. It is same as `System.out.printf()`.

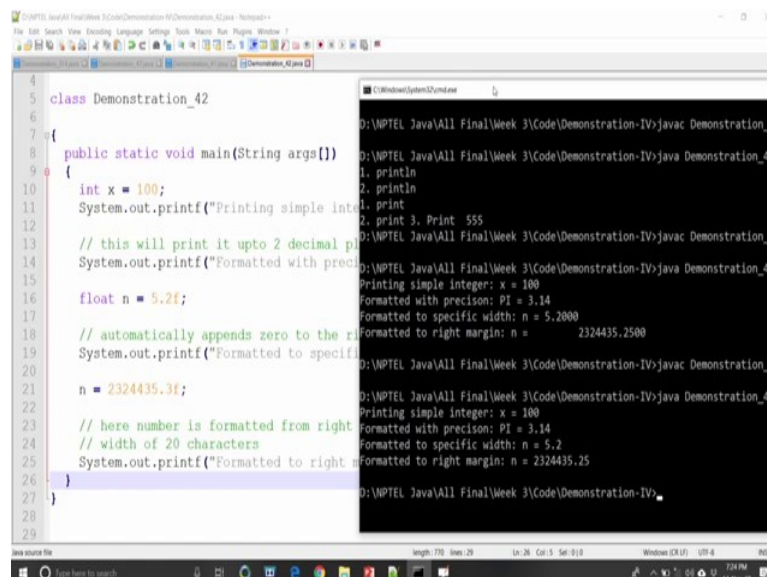
(Refer Slide Time: 08:55)



```
1 // A Java program to demonstrate working of printf() in Java
2 // Edit Demonstration_42.java
3
4
5 class Demonstration_42
6 {
7     public static void main(String args[])
8     {
9         int x = 100;
10        System.out.printf("Printing simple integer: x = %d\n", x);
11
12        // this will print it upto 2 decimal places
13        System.out.printf("Formatted with precision: PI = %.2f\n", Math.PI);
14
15        float n = 5.2f;
16
17        // automatically appends zero to the rightmost part of decimal
18        System.out.printf("Formatted to specific width: n = %.4f\n", n);
19
20        n = 2324435.3f;
21
22    }
23 }
```

And the printf() function is basically the same as the standard printf() in C programming. As the name printf() it is called the formatted print. That means, we can mention the format in printing. Again let us have the quick demo of this method this type of use of the print printf().

(Refer Slide Time: 09:12)



```
4 class Demonstration_42
5 {
6     public static void main(String args[])
7     {
8         int x = 100;
9         System.out.printf("Printing simple integer: x = %d\n", x);
10
11        // this will print it upto 2 decimal places
12        System.out.printf("Formatted with precision: PI = %.2f\n", Math.PI);
13
14        float n = 5.2f;
15
16        // automatically appends zero to the rightmost part of decimal
17        System.out.printf("Formatted to specific width: n = %.4f\n", n);
18
19        n = 2324435.3f;
20
21        // here number is formatted from right
22        // width of 20 characters
23        System.out.printf("Formatted to right margin: n = %.20f\n", n);
24    }
25 }
```

```
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_41
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_41
1. println
2. println
2. print 3. Print 555
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_42
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_42
Printing simple integer: x = 100
Formatted with precision: PI = 3.14
Formatted to specific width: n = 5.2000
Formatted to right margin: n = 2324435.2500
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_42
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_42
Printing simple integer: x = 100
Formatted with precision: PI = 3.14
Formatted to specific width: n = 5.2
Formatted to right margin: n = 2324435.25
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>
```

And in this you can see the program here demonstration_42. And we declare one method the main method in this main method we declare a variable x as 100. And now you see you use the print f method printing simple integer and to print the value we use

percentage d. It is just like a c syntax within double quote whatever you write it will print as a (Refer Time: 09:46). But the format if you specify then the value for that format needs to be placed by means of a comma. There is no plus operator like that is there in print or println(). So, here percentage d this means that x will be printed in the in decimal format %d stands for the decimal format.

Now, in the second example if we see that we print one value call the Math.P I Math.P I is defined in Java.lang package in the math class. And P I is a constant is basically pi 22 by c that is 3.1414519. So, this value will be printed. Now, here you can see the format that we have mentioned to print the value of pi is percentage.2f. This indicates that it will print after decimal only 2 digits and then before the, whatever the value it is there. So, it is basically precision up to the 2 number the decimal point.

Now, again we declare another value float and this is 5.2 f we can declare is a floating point number. Now again we can print this floating point number which has the different form of percentage.4 if it basically says you can understand what is.mean; it means that after decimal it will print up to 4 digits and then percentage whatever the value before the decimal. So, in that case, if we print n so definitely it will print 5.2000. So, there may after decimal four digits. Now if we assign this value like 0.38 and again if we call if we print this with different format 0.20.48 you can understand what will do. It will basically after decimal 4 digits and then before decimal maximum up to 20.

Now, let us run this program then will come to the discussion again ok. So, 4.2 is under compilation compiling the program is successfully compiled now executed. Now you can see what are the different output you can give it. So, hope that the output you can just see the program and little bit output that can be right. Ok, you can see the output and then format now let us come back to the program again. So, I am just going to little bit do the changes there. Now let us come to the second statement second print printf() statement second pass second.

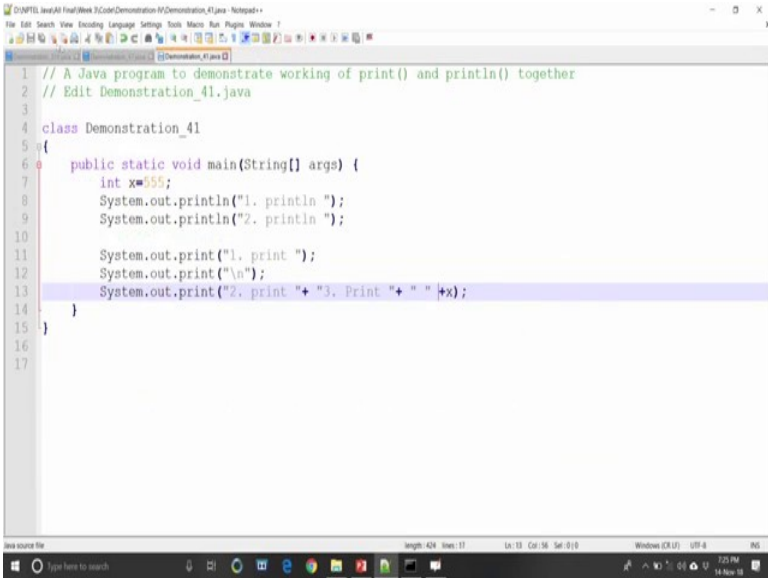
Now, come to this percentage 4.8 right we just print here say for instead of 0.4 we just point one ok. And say percentage here 2.1 2.1 percentage fine. Now next statement here is 0.2 after decimal 0.22 and here before decimal if we see the total number 7, but we want to declare 6. Now let us see what will happen. Now, here, you can see what we got

it there. So, is a truncate it now, truncation is basic not truncated. So, 0.25 it is basically what is the idea about it is basically approximate right. So, right.

So, it basically does not truncate the decimal value it basically takes the highest value whatever it is there without 3 before decimal it will not be truncated. However, after decimal, as we have mention 0.2 although there is a 0.3 it has been cast to 0.25. So, there is a little bit manipulation that the Java runtime in interpreter will do it and then it will print it. So, now, here I am just want to mention one thing that whenever you use the print f the formatting matters a lot. And you have to very careful about the formatting; sometimes it is result may be correct, but because of the wrong format that you have mentioned you may not display the right output.

So, that needs you very careful about while you are using print f. If you do not use printf() rather simply println() all those things are not applicable no formatting is possible. But println() in that case is very simple that although it will not do any format for you, but it will print the right value for you. So, sometimes it is very safe to use the print() or println() method instead of printf() in Java.

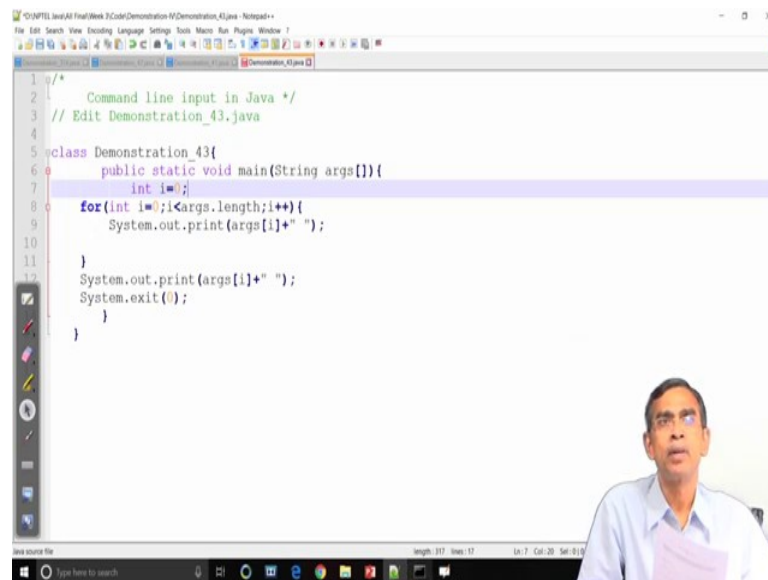
(Refer Slide Time: 14:42)



```
1 // A Java program to demonstrate working of print() and println() together
2 // Edit Demonstration_41.java
3
4 class Demonstration_41
5 {
6     public static void main(String[] args) {
7         int x=555;
8         System.out.println("1. println ");
9         System.out.println("2. println ");
10
11         System.out.print("1. print ");
12         System.out.print("\n");
13         System.out.print("2. print "+ "3. Print "+ " "+ "x");
14     }
15 }
16
17
```

Our next demonstration so we have to learn about print println() and printf() method to display the output on the standard output namely display screen.

(Refer Slide Time: 14:48)



```
1  /*
2   * Command line input in Java */
3   // Edit Demonstration_43.java
4
5  class Demonstration_43{
6      public static void main(String args[]){
7          int i=0;
8          for(int i=0;i<args.length;i++){
9              System.out.print(args[i]+" ");
10
11          }
12          System.out.print(args[i]+" ");
13          System.exit(0);
14      }
15  }
```

Now, I will discuss about how the input can be passed to users program there are three ways. One is the command line input so command line input concept is that; while you run the program; that means, you call the program. That means, your main program that time itself what are the input that is required for your program you can specify. This means that Java will take these inputs automatically and then run it Java will not take any input later on then I mean it will not ask any input.

So, if you know that these are the inputs, that is required for your program you can pass them at the time of calling this program or executing this program. Now whatever the input that will give Java interpreter will store all this input in a temporary array. So, this array is called the args. So, screen args that you can see is just highlighter you use. String args that we use so this string args basically we will use you use temporality store all the input that you passed. And as you know it is declared this string; that means, all the input will be stored as a string object and it will be processed then.

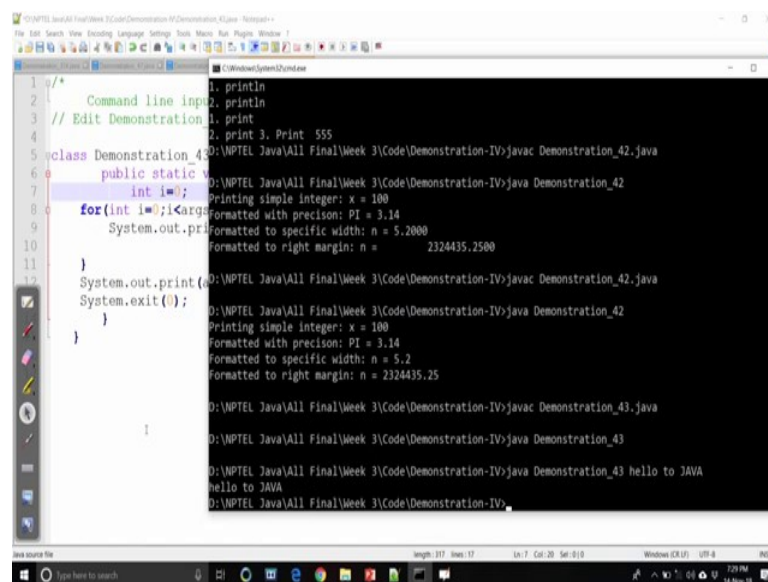
Now we have already discussed in theory class that is a string object can be configured or passed to transform to the different type using the different methods defined there in Java.lang package. Now here let us see this example it is a demonstration for 34.3 and here the public static void main (String args[]) as usual we do not have to do anything here. But now you see you have to use a for a loop. So, power in i equals to 0, i will be

less than args.length. args.length is basically the number of inputs you passed that means how many input object is stored in the args array.

So, this basically there if you pass 3 then it will be 0 to 2 actually this kind of thing. Because the first object is stored in 0 indexes is 13. So, the total number of the object is 3 and it will basically store in args 0, args 1, args 3 if it had it will be three inputs are passed. Now the next statement is basically System.out.println(args.i). It basically what about the screen that he passed it what is the input that you have passed it will be print as a string and it will just go on printing this one.

Now, instead of println() we write print() only. So, it will basically print and at the end System.out.print() \n print after the, for loop fine yes. And fine so next statement is customary written is no need to write it system.exit(0) is basically will quit this execution anyway. So, now, this is the program that we have written you can understand what the program will do for you. The program can be called with input and though those input will be passed to this object. So, let us have this execution this program there in I consider fine no problem let us see.

(Refer Slide Time: 18:16)



The screenshot shows a Java IDE with a code editor on the left and a console window on the right. The code in the editor is as follows:

```
1. println
2. Command line input2.println
3. // Edit Demonstration1.println
4. 2. print 3. Print 555
5.
6. class Demonstration_42 {
7.     public static void main(String[] args) {
8.         for(int i=0; i<args.length; i++) {
9.             System.out.print(args[i] + " ");
10.        }
11.        System.out.println();
12.        System.exit(0);
13.    }
14. }
```

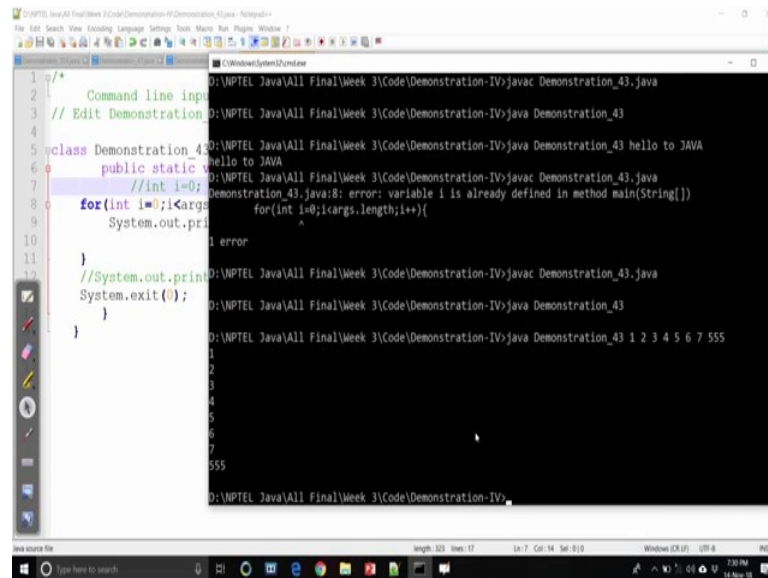
The console window shows the output of the program:

```
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_42.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_42
Printing simple Integer: x = 100
Formatted with precision: PI = 3.14
Formatted to specific width: n = 5.2000
Formatted to right margin: n = 2324435.2500
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_42.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_42
Printing simple Integer: x = 100
Formatted with precision: PI = 3.14
Formatted to specific width: n = 5.2
Formatted to right margin: n = 2324435.25
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_43.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_43
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_43 hello to JAVA
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_43 hello to JAVA
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_43 hello to JAVA
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV\java Demonstration_43 hello to JAVA
```

We have done some mistake here run hello ok. So, now, here see this fine. So, now, you see we have called this method with 3 input hello to java. Now, so, this method takes and then print it like this one. Now let us go to the program again yeah. So, System.out.println() where the first System.out.println() go there println() no issue. And

then and fine we have a little bit change it absolutely no problem there are a no major changes minor changes we have done it. Then not equal fine command fine right run this program.

(Refer Slide Time: 19:13)



The screenshot shows an IDE with a Java file named `Demonstration_43.java`. The code is as follows:

```
1 // Command line input
2 // Edit Demonstration
3
4
5 class Demonstration_43 {
6     public static void main(String[] args) {
7         //int i=0;
8         for(int i=0;i<args.length;i++){
9             System.out.println(args[i]);
10        }
11    }
12 }
13
```

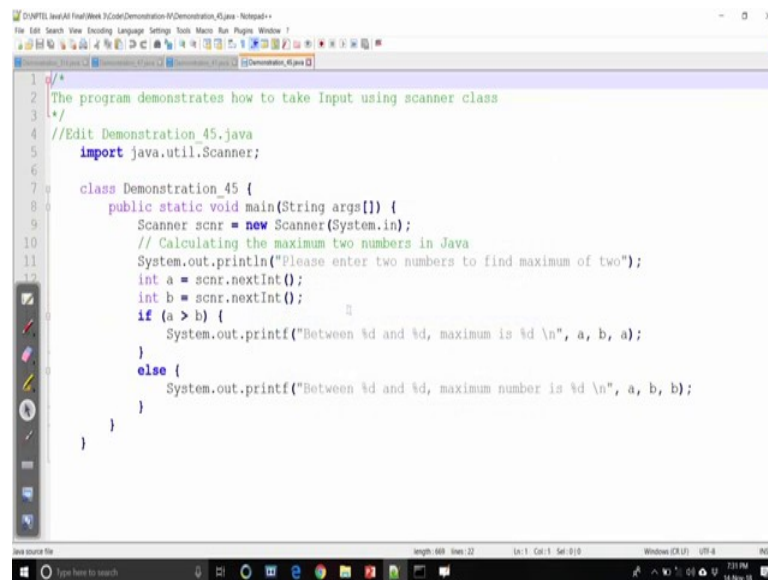
The IDE shows a compilation error: `Demonstration_43.java:8: error: variable i is already defined in method main(String[] args)`. The output window shows the following output:

```
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_43.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_43
hello to JAVA
1
2
3
4
5
6
7
555
```

Now we are calling this program passing may not 1 2 3 4 5 6 7, 1 2 3 4 5 6 7 now you see the different input that we are passing separated by a comma. Here although we have entire 1 2 3 all these are the digits or next is maybe say 5 5 5 let us see ok. So, we have the entire 8 numbers. And all these numbers entered in Java program will be these are the string and they will be printed as a string. And the 8 numbers should be printed in the next line let us run this program.

Now, you see these are the numbers it has those numbers have been printed here taking the input ok. Now go to the right. So, this is the way that by which we can pass command line input; now we have one more example so, that we can understand about. So, that command line input needs to be used very carefully because you have to specify whether you have passed input or not. So, that is why some step if you had it then it will be a more robust program.

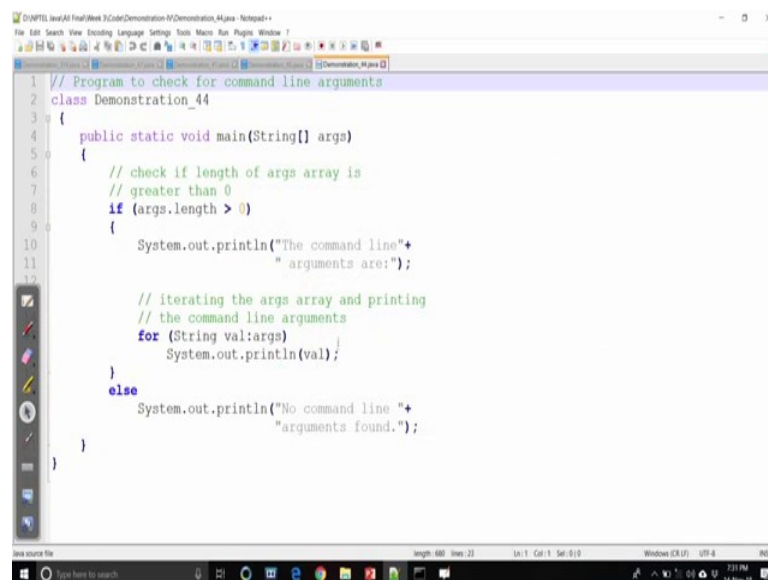
(Refer Slide Time: 20:23)



```
1  /*
2  The program demonstrates how to take Input using scanner class
3  */
4  //Edit Demonstration_45.java
5  import java.util.Scanner;
6
7  class Demonstration_45 {
8      public static void main(String args[]) {
9          Scanner scnr = new Scanner(System.in);
10         // Calculating the maximum two numbers in Java
11         System.out.println("Please enter two numbers to find maximum of two");
12         int a = scnr.nextInt();
13         int b = scnr.nextInt();
14         if (a > b) {
15             System.out.printf("Between %d and %d, maximum is %d \n", a, b, a);
16         }
17         else {
18             System.out.printf("Between %d and %d, maximum number is %d \n", a, b, b);
19         }
20     }
21 }
```

But regarding the robust program writing will discuss in details later on. So, it is a 4.4 demo 4.4.

(Refer Slide Time: 20:32)



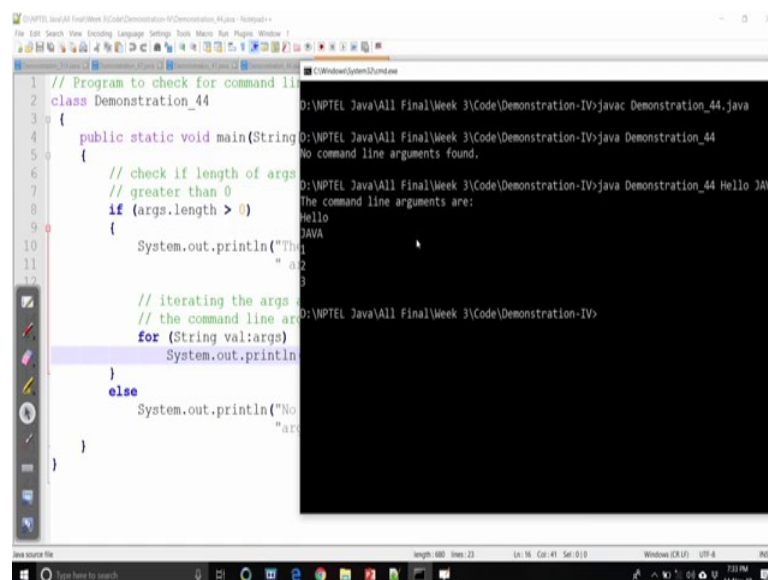
```
1  // Program to check for command line arguments
2  class Demonstration_44
3  {
4      public static void main(String[] args)
5      {
6          // check if length of args array is
7          // greater than 0
8          if (args.length > 0)
9          {
10             System.out.println("The command line"+
11                                " arguments are:");
12
13             // iterating the args array and printing
14             // the command line arguments
15             for (String val:args)
16                 System.out.println(val);
17         }
18         else
19             System.out.println("No command line "+
20                                "arguments found.");
21     }
22 }
```

Yeah, now let us see the program the same program it basically checks whether during the command line argument is passed or not if during the common line no argument is passed it will report that statement if it passed it will print this. So, here basically the pass if statement to check that if during the command during the execution of the program if it

passed any common line argument not. If it is not then it will print a message that the common line arguments are there.

If it not then it will print that the command line argument no command line argument is found. So, and if it is there then it will print all the command line arguments using is stress is expiry special for commands string well args; that means, all the value what about the true it will go on printing. So, this is basically an alternative way of printing all the input that you have passed to this program. Now let us run this program it basically reports if you do not pass any input.

(Refer Slide Time: 21:39)



The screenshot shows an IDE with a Java file named `Demonstration_44.java` and a terminal window. The Java code is as follows:

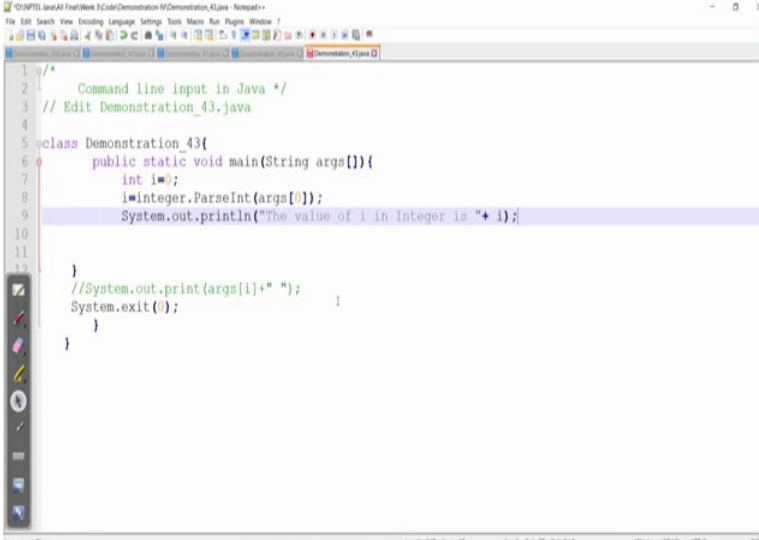
```
1 // Program to check for command line arguments
2 class Demonstration_44
3 {
4     public static void main(String[] args)
5     {
6         // check if length of args
7         // greater than 0
8         if (args.length > 0)
9         {
10            System.out.println("The command line arguments are:");
11            for (String val:args)
12            {
13                System.out.println(val);
14            }
15        }
16        else
17        {
18            System.out.println("No command line arguments found.");
19        }
20    }
21 }
```

The terminal window shows the output of running the program. The first run shows "No command line arguments found." The second run shows "The command line arguments are:" followed by "Hello", "Java", and "123".

And if in the report if you pass any input then print it. Now we are running this program without passing any input 4.4. Now so we are running this program without any input and you will see it will. So, no command line argument found. So, it gives that because we do not give input. Now again we are running the same program while passing hello Java hello Java 123.

So, these things are there ok. So, so fine no problem. Now you see it print it will the command line arguments are hello Java 1 2 3 it is according the next other than else part and then for the statement. So, so this is I mean this is a trick for writing a good program whenever you use the command line arguments. Now command line arguments also can be used to pass into the integer I can go to another program let us go to the 4.3 demos of the 4.3, 4.3 right.

(Refer Slide Time: 22:54)

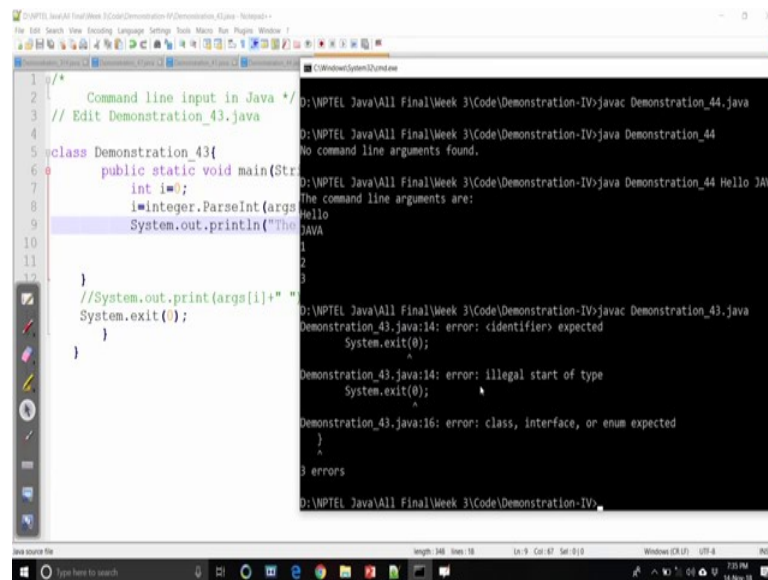


```
1  /*
2   * Command line input in Java */
3   // Edit Demonstration_43.java
4
5  class Demonstration_43{
6      public static void main(String args[]){
7          int i=0;
8          i=Integer.parseInt(args[0]);
9          System.out.println("The value of i in Integer is "+ i);
10
11      }
12      //System.out.print(args[i]+" ");
13      System.exit(0);
14  }
```

Yeah, so, here in the for loop of comment [FL] maximum for loop brackets second bracket we comment [FL] fine now we are just ok. So, comment [FL] fine in you can delete this formality clattering will be above it fine we are removing and fine. So, now we are just here integer I equal 0 declare [FL] ok.

We are declaring one variable integer and then we are I equals to integer.parseInt() integer.parse p a r s e parse capital over p a r s e parse int I n t capital I n t parse I n t parse int then args 0. Then System.out.println() int then System.out.println() in outer command System.out.print()ln() then the value within the value of I in an integer is plus I semicolon system exit a common [FL] fine.

(Refer Slide Time: 24:43)



The screenshot shows an IDE with a Java file named `Demonstration_43.java` and a terminal window. The Java code is as follows:

```
1  /*
2   * Command line input in Java */
3   // Edit Demonstration_43.java
4
5  class Demonstration_43{
6      public static void main(String[] args){
7          int i=0;
8          i=Integer.parseInt(args[0]);
9          System.out.println("Hello
10         JAVA");
11     }
12     //System.out.print(args[i]+"
13     System.exit(0);
14 }
15 }
```

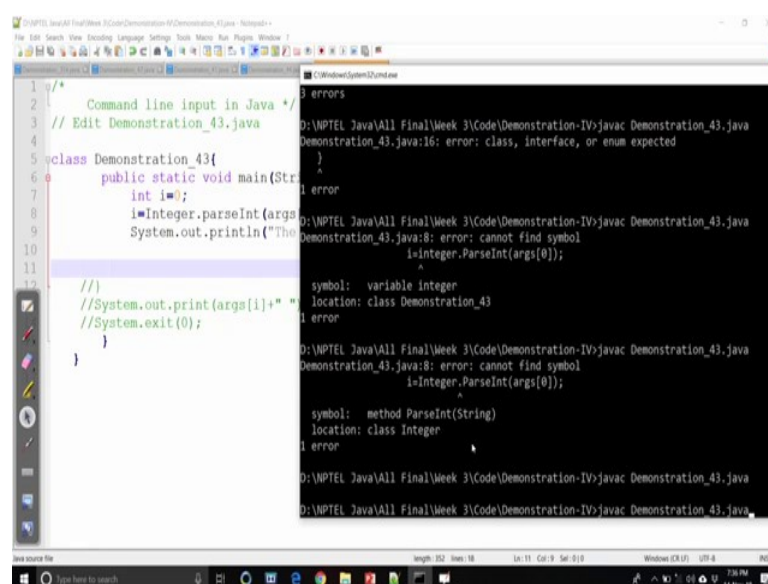
The terminal window shows the following output and errors:

```
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_44.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_44
No command line arguments found.
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_44 Hello JAVA
The command line arguments are:
Hello
JAVA
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_43.java
Demonstration_43.java:14: error: <identifier> expected
    System.exit(0);
                ^
Demonstration_43.java:14: error: illegal start of type
    System.exit(0);
                ^
Demonstration_43.java:16: error: class, interface, or enum expected
    }
    ^
3 errors
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>
```

Now you can change what we have done it nothing a different will take the passed input. And then passed input will be passed as a string, but it will be converted to an integer. So, how so common? [FL]

So, you can command Java bracket [FL] ok. So, we have fine [FL] [FL]. It is fine this is correct this is fine. Now it is there little bit clean the always thing garbage and then you can run it.

(Refer Slide Time: 25:23)



The screenshot shows the same IDE with the same Java code as before. The terminal window shows the following output and errors:

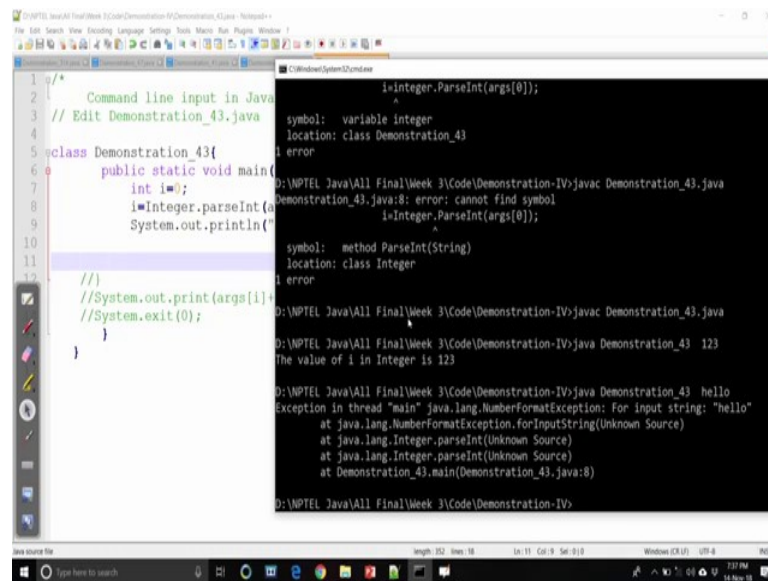
```
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_43.java
Demonstration_43.java:16: error: class, interface, or enum expected
    }
    ^
1 error
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_43.java
Demonstration_43.java:8: error: cannot find symbol
    i=Integer.parseInt(args[0]);
      ^
symbol:   variable Integer
location: class Demonstration_43
1 error
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_43.java
Demonstration_43.java:8: error: cannot find symbol
    i=Integer.parseInt(args[0]);
      ^
symbol:   method parseInt(String)
location: class Integer
1 error
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_43.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>
```

Capital I right this is correct parse capital small p [FL] it is [FL].

Student: (Refer Time: 25:52).

Now let us we have written the program we have to go fast. Now we can see the input can be passed as an integer right. 1 2 3 and this basically although we passed as a string, it will convert to an integer enter the value of I is an integer.

(Refer Slide Time: 26:13)



The screenshot shows an IDE with a Java file named `Demonstration_43.java`. The code is as follows:

```
1  /*
2  3  Command line input in Java
3  4  // Edit Demonstration_43.java
4  5  class Demonstration_43{
5  6  7  public static void main(
6  7  8  int i=0;
7  8  9  i=Integer.parseInt(a
8  9  10 System.out.println("
9  10 11
10 11 //)
11 12 //System.out.print(args[1])
12 13 //System.exit(0);
13 14 }
14 15 }
```

The output window shows the following execution results:

```
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_43.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_43 123
The value of i in Integer is 123

D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_43 hello
Exception in thread "main" java.lang.NumberFormatException: For input string: "hello"
    at java.lang.NumberFormatException.forInputString(Unknown Source)
    at java.lang.Integer.parseInt(Unknown Source)
    at java.lang.Integer.parseInt(Unknown Source)
    at Demonstration_43.main(Demonstration_43.java:8)
```

But if we pass hello then it basically gives an error. The error is basically run time error. So, this is an error because it is not able to convert this value to an integer value. So, this is the error, which is pointed out. But will learn about how this kind of error can be dealt with later on.

Now, so we have learned about command line input the value whatever it is passed as a string and that can be converted to any type as you want. So, this is the idea of the common line. Another way the scanner input. So, let us have the demo of this type of I put 4.5. So, now, let us see there is one plus which is defined in util package Java.util the name of the class is the scanner class. This scanner basically helps you to read the input from the keyboard and then that can be used for your purpose.

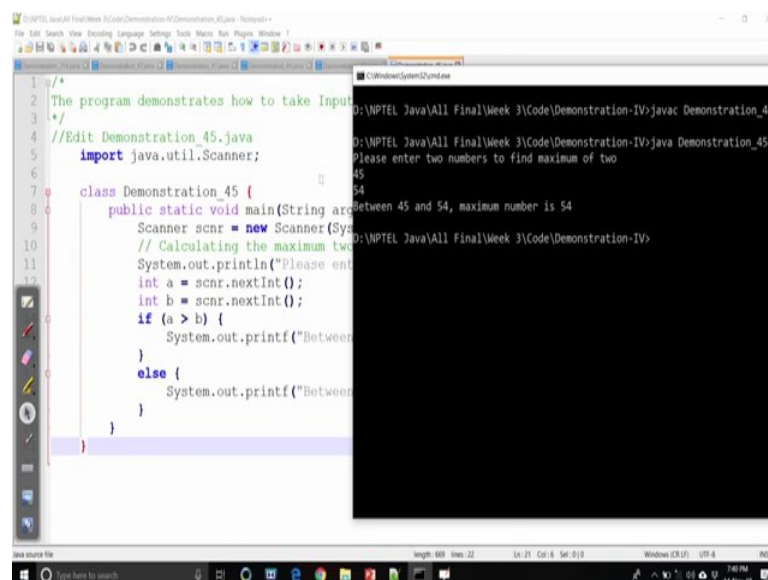
Again this scanner will read as an integer as a float as you string whatever it is there. Now here is a quick demo of this one we declare one plus the main plus is demonstration under 45_45 and here you see that the first declaration `Scanner scnr = new Scanner(System.in)` So, this is the typical syntax about new is basically the q r to create

an object the name of the object a type of the object belongs to the class scanner. And then here argument this is default we have to parse system.in that is means the standard input. So, system.in indicates the standard input means the keyboard

So, basically, the scanner will read some value from the keyboard. So, this is the way we can create an object the object name of the object is scnr. So, this is the use the defined we have defined this object name. Now we would like to read two numbers from the keyboard using the scanner class and then those actually the scnr object will be ready for you to read the input from your keyboard.

Now, this basically can be done by means of one method which is defined in the scanner class itself is called the next int. That means it will go on reading and here to success in reading will take place the value will read from the keyboard and then store into the temporary values a and b. And whatever the value it will be read as an integer because is a next int method will scan as an integer. And then the next two statements if the statement is basically processing it will print whichever the value is highest one. Now, let us quickly run this program to see how it works.

(Refer Slide Time: 28:59)



The screenshot shows an IDE with two windows. The left window displays the source code for a Java program named 'Demonstration_45.java'. The code imports 'java.util.Scanner', defines a class 'Demonstration_45', and includes a 'main' method. The 'main' method creates a 'Scanner' object 'scnr' from 'System.in', prompts the user to enter two numbers, reads them as integers 'a' and 'b', and then uses an 'if' statement to compare them and print the maximum. The right window shows the program's execution. It displays the prompt 'Please enter two numbers to find maximum of two', followed by the user input '45' and '54'. The program then outputs 'Between 45 and 54, maximum number is 54'.

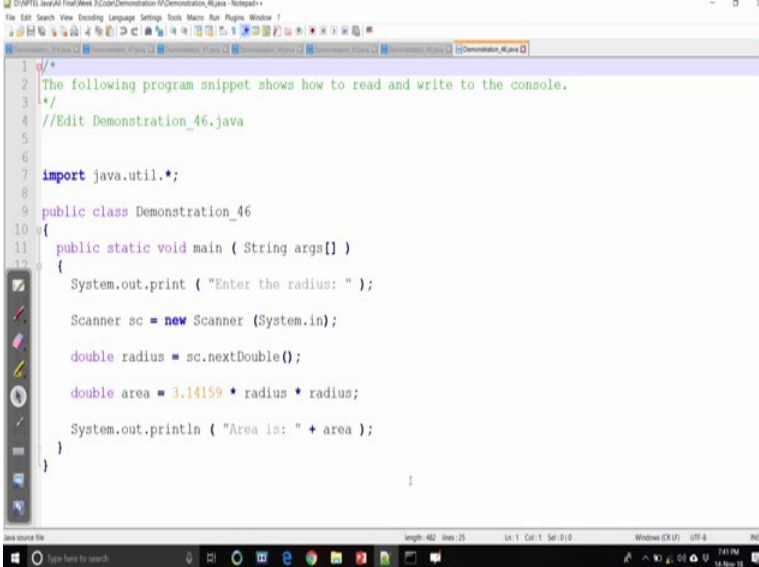
```
1  /*
2  The program demonstrates how to take Input
3  */
4  //Edit Demonstration_45.java
5  import java.util.Scanner;
6
7
8  class Demonstration_45 {
9
10     public static void main(String args[]) {
11         Scanner scnr = new Scanner(System.in);
12         // Calculating the maximum two
13         System.out.println("Please enter two numbers");
14         int a = scnr.nextInt();
15         int b = scnr.nextInt();
16         if (a > b) {
17             System.out.printf("Between %d and %d, maximum number is %d", a, b, a);
18         }
19         else {
20             System.out.printf("Between %d and %d, maximum number is %d", a, b, b);
21         }
22     }
23 }
```

D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_45.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_45
Please enter two numbers to find maximum of two
45
54
Between 45 and 54, maximum number is 54
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>

Ok, now we are giving two input so let us give 45 then 54 enter. So, here we can see the input the result that will be printed here like this one. So, that means it read two number successfully and the two numbers it basically compares and then maximum number is printed there.

So, this is the one way is an alternate way to common line input that we have discussed. And some people find it more I mean easy and then convenient for their programming so it is there. Now, let us have another demo on the same utility of scanner class again 4.6.

(Refer Slide Time: 29:57)

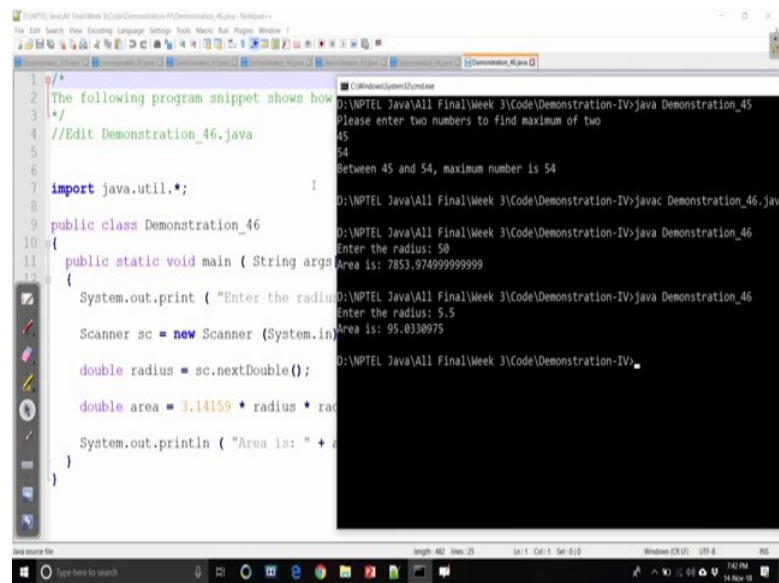
A screenshot of a Java IDE window titled "C:\NFTL\Java\All Final\Week 3\Code\Demonstration 46\Demonstration_46.java - Notepad++". The code is as follows:

```
1  /*
2  The following program snippet shows how to read and write to the console.
3  */
4  //Edit Demonstration_46.java
5
6
7  import java.util.*;
8
9  public class Demonstration_46
10 {
11     public static void main ( String args[] )
12     {
13         System.out.print ( "Enter the radius: " );
14
15         Scanner sc = new Scanner (System.in);
16
17         double radius = sc.nextDouble();
18
19         double area = 3.14159 * radius * radius;
20
21         System.out.println ( "Area is: " + area );
22     }
23 }
```

The IDE interface includes a menu bar (File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Windows), a toolbar, and a status bar at the bottom showing "length: 402, lines: 25, Ln: 1, Col: 1, Sel: 0 | 0, Windows (X) OS, UTF-8, JRS".

So, let us have the quick again we use the scanner class which is defined in a util package. So, input statement that we have used earlier like this one and here we just see enter the radius we read the radius. But in this case, we read the number not as an integer as a float. So, we should use the next double now again we create the scanner object in this case sc and then radius the value will be read from the keyboard as a double and then it will processes it and then give the result. So, let us run this program.

(Refer Slide Time: 30:31)



The screenshot shows an IDE with two windows. The left window displays a Java program named `Demonstration_46.java`. The code includes a comment, an import statement for `java.util.*`, and a `main` method that uses a `Scanner` to read a radius and calculate the area of a circle using the formula $area = 3.14159 * radius * radius$. The right window shows the program's execution output, which includes prompts for two numbers to find the maximum and for a radius to calculate the area, followed by the calculated results.

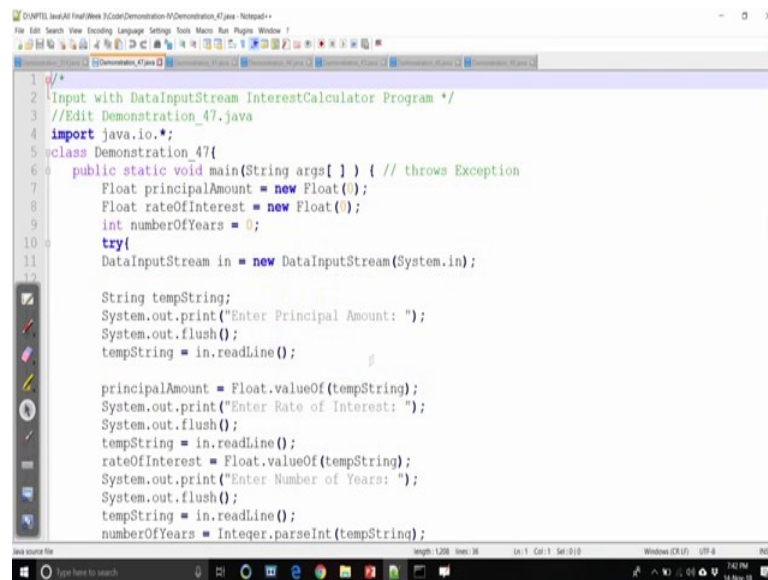
```
1  /*
2  The following program snippet shows how
3  */
4  //Edit Demonstration_46.java
5
6
7  import java.util.*;
8
9  public class Demonstration_46
10 {
11     public static void main ( String args[] )
12     {
13         System.out.print ( "Enter the radius: " );
14
15         Scanner sc = new Scanner (System.in);
16
17         double radius = sc.nextDouble();
18
19         double area = 3.14159 * radius * radius;
20
21         System.out.println ( "Area is: " + area );
22     }
23 }
```

```
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_45
Please enter two numbers to find maximum of two
45
54
Between 45 and 54, maximum number is 54
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>javac Demonstration_46.java
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_46
Enter the radius: 50
Area is: 7853.974999999999
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>
Enter the radius: 5.5
Area is: 95.0330975
D:\NPTEL Java\All Final\Week 3\Code\Demonstration-IV>
```

So, here we can give for 50.5 anyway so no problem although we enter 50, it takes as a format again run this program 5.5. So, this basically takes the value as a floating point number as a double format and then calculate the area of the value using the formula that is defined in the method.

So, this is basically the way the Scanner class can work for you can read any number from the keyboard in any format and regarding the different format that is possible that you can have in the scanner class itself. Like next string, next float next double next string so many ways the input can be read. Now, so this is the scanner class, now let us have the last method using the data input stream class.

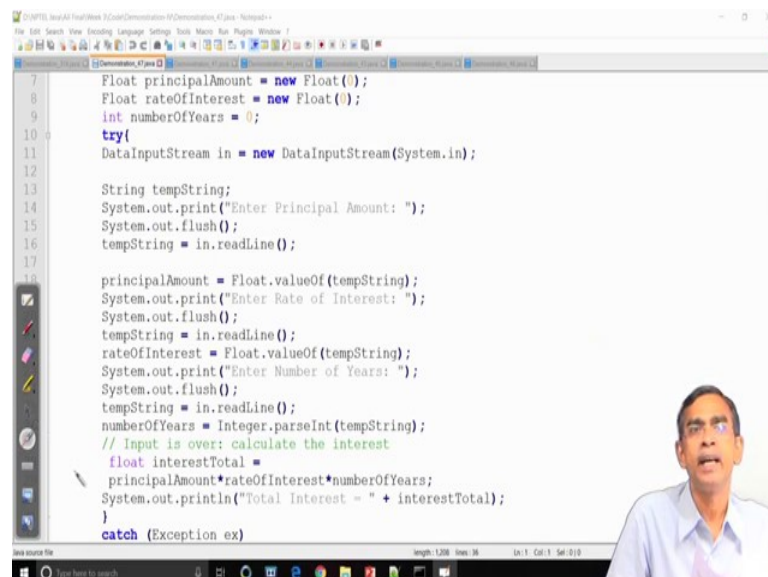
(Refer Slide Time: 31:38)



```
1  /*
2  Input with DataInputStream InterestCalculator Program */
3  //Edit Demonstration_47.java
4  import java.io.*;
5  class Demonstration_47{
6      public static void main(String args[ ] ) { // throws Exception
7          Float principalAmount = new Float(0);
8          Float rateOfInterest = new Float(0);
9          int numberOfYears = 0;
10         try{
11             DataInputStream in = new DataInputStream(System.in);
12
13             String tempString;
14             System.out.print("Enter Principal Amount: ");
15             System.out.flush();
16             tempString = in.readLine();
17
18             principalAmount = Float.valueOf(tempString);
19             System.out.print("Enter Rate of Interest: ");
20             System.out.flush();
21             tempString = in.readLine();
22             rateOfInterest = Float.valueOf(tempString);
23             System.out.print("Enter Number of Years: ");
24             System.out.flush();
25             tempString = in.readLine();
26             numberOfYears = Integer.parseInt(tempString);
```

It is a little bit difficult to understand at the moment regarding this data input stream, who will discuss in details when we will consider about input-output stream. Now let us have a very quick demo of this use usage of the data input stream class it is more or less similar to the scanner.

(Refer Slide Time: 31:56)



```
7      Float principalAmount = new Float(0);
8      Float rateOfInterest = new Float(0);
9      int numberOfYears = 0;
10     try{
11         DataInputStream in = new DataInputStream(System.in);
12
13         String tempString;
14         System.out.print("Enter Principal Amount: ");
15         System.out.flush();
16         tempString = in.readLine();
17
18         principalAmount = Float.valueOf(tempString);
19         System.out.print("Enter Rate of Interest: ");
20         System.out.flush();
21         tempString = in.readLine();
22         rateOfInterest = Float.valueOf(tempString);
23         System.out.print("Enter Number of Years: ");
24         System.out.flush();
25         tempString = in.readLine();
26         numberOfYears = Integer.parseInt(tempString);
27         // Input is over: calculate the interest
28         float interestTotal =
29             principalAmount*rateOfInterest*numberOfYears;
30         System.out.println("Total Interest = " + interestTotal);
31     }
32     catch (Exception ex)
```

But here we use data input string class which is defined in I o package. So, in this case, we should import Java.io.*. Their data input stream class is where we can I defend

Java.io.datainputstream also anyway .* means it will import the entire package io package.

Now, let us see we have declared two objects principal amount rate of interest they are declared as a floating object. So, an object of type float can be declared using this kind of statement. And also we declared another member elements number of years, which is declared as an integer. So, three variables three elements have been declared here as principal amount rate of interest and number of years. Now we just create an object the name of the object is called in basically input object we can write I np whatever you know in ok. So, in as the object is created this object is of type data input stream and data input stream and if you see the argument system.in that mean. We now decide our we now direct the data input stream to read from the keyboard.

So, here also you can use a filename so that it can read from the file regarding file reading and everything that is another discussion will be discussed later on. So, yes system.in means that we want to read some value from the keyboard for which we create an object call in. Now temp string is a temporary string and again data in data input stream class will read anything in the form of a string and it is required to convert into the desired format.

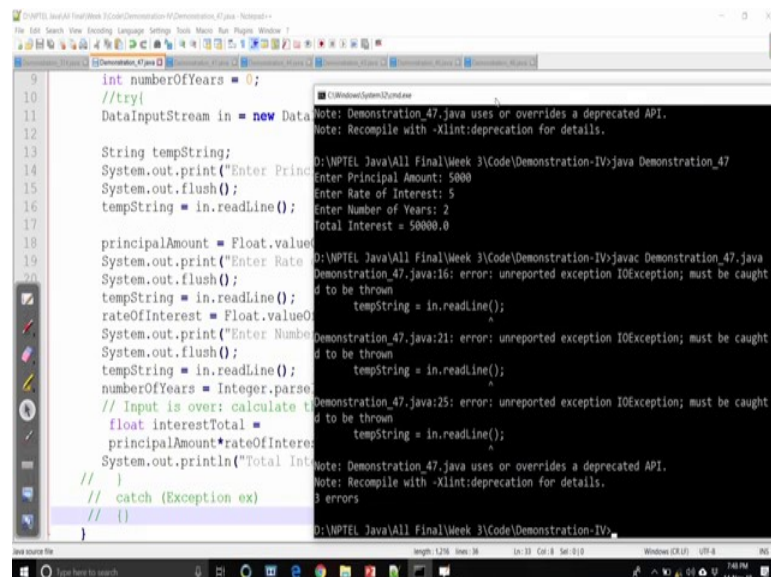
Now, let us see how we can work for that. So, System.out.print() enter principal it gives a form to the user that you should enter the principal amount. System.out class is basically clean the buffer. So, basically, every keyboard or standard input has its own buffer if you call this it will clean so, that no garbage will be entered into your in the input. Now temp string basically stores the value, which is read from the buffer keyboard and it is basically to read a buffer keyboard the statement the method that is required is read line method is declared in data input stream class. So, it basically read the entire line from the keyboard buffer and it will store as a string in a temp string with a temporary variable.

Now, we convert this value temp string into the float type and the conversion is followed by this kind of method float.value of temp string. So, this is one method by which a string can be converted into a floating point value. Now again the next we can give the form to enter the rate of interest same as system.out.class again same the read the

keyboard buffer and then we finally, read the string and then the string is converted into the float value and store in the rate of interest.

Again we give the next command next informed the entire number of years. We clean the buffer read the buffer and then finally, convert this value that we read from the keyboard using `Integer.parseInt()` and store into the variable call the number of years. And then we declare interest total and this is simple formula calculate the interest given the principal amount rate of interest number of years and then print the result. And let us see how you can run this? There are a few things I will discuss shortly.

(Refer Slide Time: 35:31)



The screenshot shows an IDE with a Java file named `Demonstration_47.java`. The code is as follows:

```
9 int numberOfYears = 0;
10 //try!
11 DataInputStream in = new DataInputStream(System.in);
12
13 String tempString;
14 System.out.print("Enter Principal Amount: ");
15 System.out.flush();
16 tempString = in.readLine();
17
18 principalAmount = Float.valueOf(tempString);
19 System.out.print("Enter Rate of Interest: ");
20 System.out.flush();
21 tempString = in.readLine();
22 rateOfInterest = Float.valueOf(tempString);
23 System.out.print("Enter Number of Years: ");
24 System.out.flush();
25 tempString = in.readLine();
26 numberOfYears = Integer.parseInt(tempString);
27 // Input is over: calculate the total interest
28 float interestTotal = principalAmount * rateOfInterest * numberOfYears;
29 System.out.println("Total Interest = " + interestTotal);
30
31 // }
32 // catch (Exception ex)
33 // {}
34 }
```

The output window shows the following execution:

```
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_47
Enter Principal Amount: 5000
Enter Rate of Interest: 5
Enter Number of Years: 2
Total Interest = 50000.0
```

The IDE also shows several warnings and errors. Warnings include "Note: Demonstration_47.java uses or overrides a deprecated API. Note: Recompile with -Xlint:deprecation for details." and errors include "Demonstration_47.java:16: error: unreported exception IOException; must be caught or declared to be thrown" and "Demonstration_47.java:21: error: unreported exception IOException; must be caught or declared to be thrown".

Let us have a quick demo we run this program ok. So, warning you can include it there is a warning because of some depreciation no issue is basic casting. So, you can ignore warning can be ignored now it is asking the principal amount right rate of interest 5 10 whatever it is here is 2 no problem. So, it is calculated the total interest like this one.

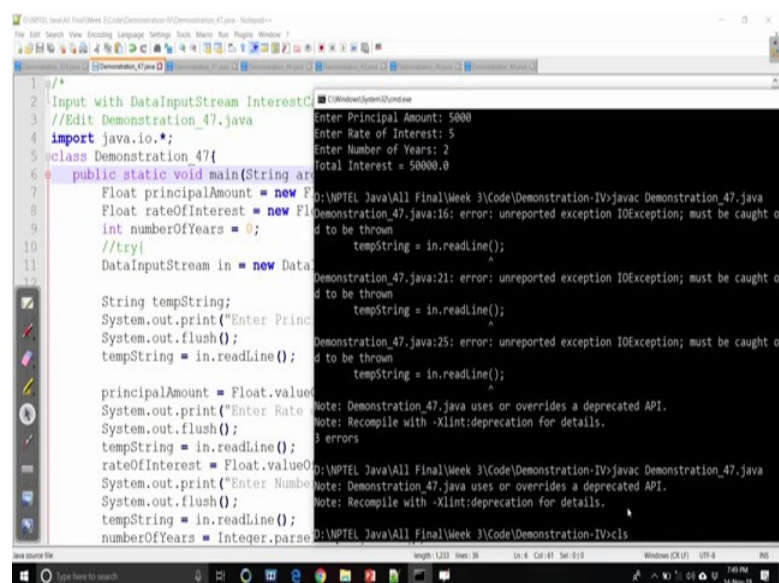
So, this is the way the method can be calculated, it is an actual rate of interest should be 0.5 whatever 0.05 like this one anyway. So, so this is a one execution now let us come back to the program again and if you see there is one statement that we have try and then within double brackets entirely brackets, we face everything and for this try going to the end we have end the curly bracket here this one and followed by a catch. Now, this is a try catch statement, which is basically for exception handling. So, if you do not put sometimes compiler will report an error and may not execute to create the bytecode. So,

that is why it is customary to use try-catch whenever the data input stream or some input stream class is used. So, regarding this try catch and everything will discuss it in details.

Now, if you do not give try catch let us see what error you can face it. So, just remove that try block command this command catch command now we have removed the try catch it is a very simple to program without any try catch as we do not know anything about it. Now let us see whether we can compile this program or not now here compilation is not successful. So; that means, it requires the try catch, which is required for dealing if there is an error in input.

So, that is why there are many `in.readLine()`. So, sometimes there may be the program can call this. So, try catch now there is an alternate way to deal with now you can do it in the main function here writing throw exception there is a one way instead of the try class also you can do that. So, here we are doing like this now we just use throws an exception this is also an alternative way in the main method if you do it then without try catch also.

(Refer Slide Time: 38:04)



The screenshot shows an IDE with a Java file named `Demonstration_47.java`. The code defines a class `Demonstration 47` with a `main` method that uses `DataInputStream` to read input. The code is as follows:

```
1 /*
2 Input with DataInputStream Interest
3 //Edit Demonstration_47.java
4 import java.io.*;
5 class Demonstration 47{
6     public static void main(String args[])
7     {
8         Float principalAmount = new Float(5000);
9         Float rateOfInterest = new Float(5);
10         int numberOfYears = 0;
11         //try{
12         DataInputStream in = new DataInputStream(System.in);
13
14         String tempString;
15         System.out.print("Enter Principal Amount: ");
16         System.out.flush();
17         tempString = in.readLine();
18
19         principalAmount = Float.valueOf(tempString);
20         System.out.print("Enter Rate of Interest: ");
21         System.out.flush();
22         tempString = in.readLine();
23         rateOfInterest = Float.valueOf(tempString);
24         System.out.print("Enter Number of Years: ");
25         System.out.flush();
26         tempString = in.readLine();
27         numberOfYears = Integer.parseInt(tempString);
28     }
29 }
```

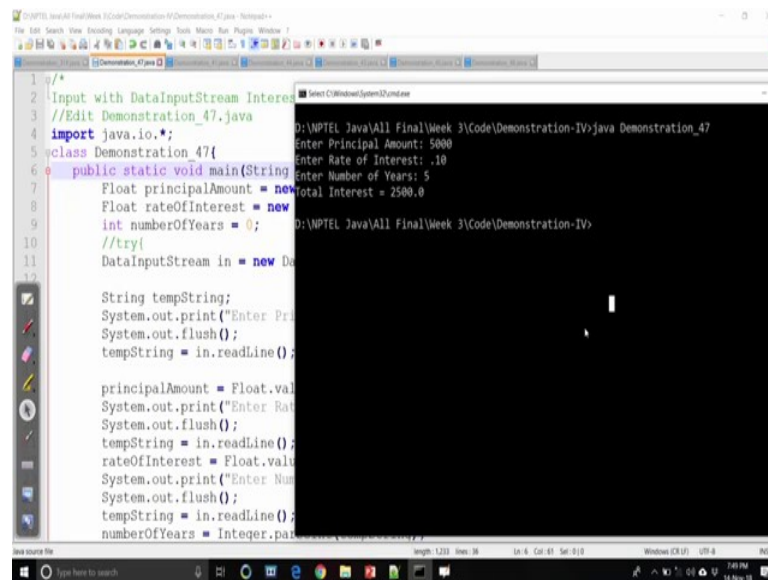
The output window on the right shows the program's execution and compilation errors:

```
Enter Principal Amount: 5000
Enter Rate of Interest: 5
Enter Number of Years: 2
Total Interest = 50000.0

D:\MPTEL Java\All Final\Week 3\Code\Demonstration-IV\javac Demonstration_47.java
Demonstration_47.java:16: error: unreported exception IOException; must be caught or
thrown
    tempString = in.readLine();
                        ^
D:\MPTEL Java\All Final\Week 3\Code\Demonstration-IV\javac Demonstration_47.java
Demonstration_47.java:21: error: unreported exception IOException; must be caught or
thrown
    tempString = in.readLine();
                        ^
D:\MPTEL Java\All Final\Week 3\Code\Demonstration-IV\javac Demonstration_47.java
Demonstration_47.java:25: error: unreported exception IOException; must be caught or
thrown
    tempString = in.readLine();
                        ^
Note: Demonstration_47.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
3 errors
```

Your program may run. So, warning you can ignore. So, this program is running now let us go clear right c l s.

(Refer Slide Time: 38:12)



```
1 /*
2 Input with DataInputStream Interest
3 //Edit Demonstration_47.java
4 import java.io.*;
5 class Demonstration_47{
6     public static void main(String args[])
7     {
8         Float principalAmount = new Float(5000);
9         Float rateOfInterest = new Float(.10);
10        int numberOfYears = 5;
11        //try
12        DataInputStream in = new DataInputStream(System.in);
13
14        String tempString;
15        System.out.print("Enter Principal Amount: ");
16        System.out.flush();
17        tempString = in.readLine();
18
19        principalAmount = Float.valueOf(tempString);
20        System.out.print("Enter Rate of Interest: ");
21        System.out.flush();
22        tempString = in.readLine();
23        rateOfInterest = Float.valueOf(tempString);
24        System.out.print("Enter Number of Years: ");
25        System.out.flush();
26        tempString = in.readLine();
27        numberOfYears = Integer.parseInt(tempString);
28
29        double totalInterest = principalAmount * rateOfInterest * numberOfYears;
30        System.out.println("Total Interest = " + totalInterest);
31    }
32 }
```

Output:

```
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>java Demonstration_47
Enter Principal Amount: 5000
Enter Rate of Interest: .10
Enter Number of Years: 5
Total Interest = 2500.0
D:\WPTEL Java\All Final\Week 3\Code\Demonstration-IV>
```

Running principal amount 5000 at 0.1 0 0.100 fine and the number of years 5. So, this basically gives you the calculation of interest how you can do this.

Now, you have learned about three ways of giving input and regarding input and output will discuss many more thing on the running discussion. So, it is a till time is a today up to this demo we have planned about print how to output on the screen and then input to the program ok.

Thank you very much.