- An exception can occur for many different reasons. Following are some scenarios where an exception occurs. A user has entered an invalid data. A file that needs to be opened cannot be found.

- Checked exceptions A checked exception is an exception that is checked (notified) by the compiler at compilation - time, these are also called as compile time exceptions . These exceptions cannot simply be ignored, the programmer should take care of (handle) these exceptions . Unchecked exceptions An unchecked exception is an exception that occurs at the time of execution. These are also called as Runtime Exceptions .

- These include programming bugs, such as logic errors or improper use of an API. Runtime exceptions are ignore d at the time of compilation. Errors These are not exceptions at all, but problems that arise beyond the control of the user or the programmer. For example, if a stack overflow occurs, an error will arise.

- They are also ignored at the time of compilation. The exception object contains name and description of the exception, and current state of the program where exception has occurred. Now the following procedure will happen. The block of the code is called Exception handler .

- If it finds appropriate handler then it passes the occurred exception to it. Appropriate handler means the type of the exception object thrown matches the type of the exception o bject it can handle. This handler prints the exception information in the following format and terminates program abnormally . Customized Exception Handling: Java exception handling is managed via five keywords: try , catch , th row , throws , and finally .

- Program statements that you think can raise exceptions are contained within a try block. If an exception occurs within the try block , it is thrown. Your code can catch this exception (using catch block) and handle it in some rational manner. To manually throw an exception, use the keyword throw .

- Any code that absolutely must be executed af ter a try block completes is put in a finally block. A try/catch block is placed around the code that might generate an exception. When an exception occurs, that exception occurred is handled by catch block associated with it. Every try block should be immediately followed either by a catch block or finally block.

- A catch statement involves declaring the type of exception you are trying to catch. If an exception occurs in protected code, the catch block (or blocks) that follows the try is checked. println("Something went wrong. "); } } } The output will be: Something went wrong.

- MULTIPLE CATCH BLOCKS A try block can be followed by multiple catch blocks. You can throw an exception, either a newly instantiated one or an exception that you just ca ught, by using the throw keyword. A finally block of code always executes, irrespective of occurrence of an Exception. Using a finally block allows you to run any cleanup - type statements that you want to execute, no matter what happens in the protected code.

- println ("The finally statement is executed"); } } } Output Exception thrown :java. It is not compulsory to have finally clauses when ever a try/catch block is present. The try block cannot be present without either catch clause or finally clause. Any code cannot be present in between the try, catch, finally blocks.

- USER - DEFINED CUSTOM EXCEPTION IN JAVA User Defined Exception or custom exception is creating your own exception class and throws that exception using throw keyword. This can be done by extending the class Exception. But practically, you will require some amount of customizing as per your programming needs. Output String = Custom message BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 8 Java - Files and I/O The java.

- io package contains nearly every class you might ever need to perform input and output (I/O) in Java. All these streams represent an input sour ce and an output destination. io package supports many data such as primitives, object, localized characters, etc. STREAM A stream can be defined as a sequence of data.

- There are two kinds of Streams InPutStream The InputStream is used to read data from a source. OutPutStream The OutputStream is used for writing data to a destination. In Java, 3 streams are created for us automatically. All these streams are attached with the console.

- BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 9 BYTE STREAMS Java byte streams are used to perform input and output of 8 - bit bytes. Though there are many classes related to byte streams but the most frequently used classes are, FileInputStream and FileOutputStream . Following is an example which makes use of these two classes to copy an input file into an output file Example import java . *; public class CopyFile { public static void main (String args []) throws IOException { FileInputStream in = null ; FileOutputStream out = null ; try { in = new FileInputStream ("input.

- txt"); out = new FileOutputStream ("output. txt"); int c ; while ((c = in . write (c); } } finally { if (in ! = null) { BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 10 in .

- close (); } } } } Now let"s have a file input. txt with the following con tent This is test for copy file. txt file with the same content as we have in input. So let"s put the above code in CopyFile.

- java file and do the following $javac CopyFile. java $java CopyFile CHARACTER STREAMS Java Byte streams are used to perform input and output of 8 - bit bytes, whereas Java Character streams are used to perform input and output for 16 - bit unicode. Though there are many classes related t o character streams but the most frequently used classes are, FileReader and FileWriter . We can re - write the above example, which makes the use of these two classes to copy an input file (having unicode characters) into an output file Example import java .

- t xt"); out = new FileWriter ("output. txt"); int c ; while ((c = in . write (c); } } finally { if (in ! close (); } } } } Now let"s have a file input.

- txt with the following content This is test for copy file. txt file with the same content as we have in input. So let"s put the above code in CopyFile. java fi le and do the following $javac CopyFile.

- Following is a simple program, which creates InputStreamReader to read stan dard input stream until the user types a "q" Example import java . println ("Enter characters, "q" to quit. "); char c ; do { c = (char) cin . print (c); } while (c !

- = "q"); } finally { if (cin ! close (); } } } } BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 13 Let"s keep the above code in ReadConsole. This program continues to read and output the same character until we press "q" $javac ReadConsole. java $java ReadConsole Enter characters, "q" to quit.

- 1 1 e e q q READING AND WRITING FILES As described earlier, a stream can be defined as a sequence of data. The InputStream is used to read data from a source and the OutputStream is used for writing data to a destination. Here is a hierarchy of classes to deal with Input and Output streams. The two important streams are FileInputStream and FileOutputStream , which would be discussed in this tutorial.

- BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 14 FileInputStream This stream is used for reading data from the files. Objects can be created using the keyword new and there are several types of constructors available. Returns the next byte of data and - 1 will be returned if it"s the end of the file. length bytes from the input stream into an array.

- Returns the total number of bytes read. If it is the end of the file, - 1 will be returned. 5 public int available() throws IOException{} Gives the number of bytes that c an be read from this file input stream. BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 15 FileOutputStream FileOutputStream is used to create a file and write data into it.

- The stream would create a file, if it doesn"t already exist, before opening it for output. Here are two constructors which can be used to create a FileOutputStream object. 4 public void write(byte[] w) Writes w. length bytes from the mentioned byte array to the OutputStream.

  Example Following is the example to demonstrate InputStream and OutputStr eam import java .

- When an Exception occurs the normal flow of the program is disrupted and the program/Application terminates abnormally, which is not recommend ed, therefore, these exceptions are to be handled. Errors are typically ignored in your code because you can rarely do anything about an error. Creating the Exception Object and handling it to the run - time system is called throwing an Exception. There might be the list of the methods that had been called to get to the method where exception was occurred.

- This ordered list of the methods is called Call Stack . The run - ti me system searches the call stack to find the method that contains block of code that can handle the occurred exception. The run - time system starts searching from the method in which exception occurred, pr oceeds through call stack in the reverse order in which methods were called. Briefly, here is how they work.

- System - generated exceptions are automatically thrown by the Java run - time system. Any exception that is thrown out of a method must be specified as such by a throws clause. CATCHING EXCEPTIONS A method catches an exception using a combination of the try and catch keywords. If the type of exception that occurred is listed in a catch block, the exception is passed to the catch block much as an argument is passed into a method parameter.

- println(myNumbers[10]); } catch (Exception e) { System. The throws keyword appears at the end of a method"s signature. The following method declares that it throws a RemoteException Example import java . For example, the following method declares that it throws a RemoteException and an InsufficientFundsException Example import java .

- } Example public class ExcepTest { public static void main (String args []) { int a [] = new int [2]; try { System . println ("Access element three :" + a [3]); } catch (ArrayInd exOutOfBoundsException e) { System . println ("Exception thrown :" + e); } finally { a [0] = 6 ; System . println ("First element value: " + a [0]); System .

- There is no need to override any of the above methods available in the Exception class, in your derived class . println(exp); } } } Result The abo ve code sample will produce the following result. out: standard output stream 2) System. in: standard input stream 3) System.

- err: standard error stream Java provides strong but flexible support for I/O related to files and netw orks but this tutorial covers very basic functionality related to streams and I/O. As a next step, compile the above program and execute it, which will result in creating output. As a next step, compile the above program and execute it, which will result in creating output. Standard Output This is used to output the data produced by the user"s program and usually a computer screen is used for standard output stream and represented as System.

- Standard Error This is used to output the error data produced by the user"s program and usually a computer screen is used for standard error stream and represented as System. *; public class ReadConsole { public static void main (String args []) throws IOException { InputStreamReader cin = null ; try { cin = new InputStreamReader (System . java file and try to compile and execute it as shown in the following program. Method & Description 1 public void close() throws IOException{} This method closes the file output stream.

- Releases any system resources associated with the fil e. 2 protected void finalize()throws IOException {} This method cleans up the connection to the file. Ensures that the close method of this file output stream is called when there are no more references to this stream. 3 public int read(int r)throws IOException{} This method reads the specified byte of data from the InputStream.

- 4 public int read(byte[] r) throw s IOException{} This method reads r. Method & Description 1 public void close()

throws IOException{} This method closes the file output stream. Releases any system resources associated with the file. 2 protected vo id finalize()throws IOException {} This method cleans up the connection to the file.

- Ensures that the close method of this file output stream is called when there are no more references to this stream. 3 public void write(int w)thro ws IOException{} This methods writes the specified byte to the output stream.

- BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 1 JAVA - EXCEPTIONS An exception (or exceptional event) is a problem that arises during the execution of a program. A network conn ection has been lost in the middle of communications or the JVM has run out of memory Some of these exceptions are caused by user error, others by programmer error, and others by physical resources that have failed in some manner Based on these, we have th ree categories of Exceptions. BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 2 EXCEPTION HANDLING Default Exception Handling : Whenever inside a method, if an exception has occurred, the method creates an Object known as Exception Object and hands it off to the run - time system(JVM). BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 3 If run - time system searches all the methods on call stack and couldnt have found the appropriate handler then run - time system handover the Exception Object to default exception handler , which is part of run - time system.

- Code within a try/catch block is referred to as protected code, and the syntax for using try/catch looks like the following Syntax try { // Protected code } catch (ExceptionName e1) { // Catch block } The code which is prone to exceptions is placed in the try block. Example public class MyClass { public static void main(String[ ] args) { BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 4 try { int[] myNumbers = {1, 2, 3}; System. The syntax for multiple catch blocks looks like the following Syntax try { // Protected code } catch (ExceptionType1 e1) { // Catch block } catch (ExceptionType2 e2) { // Catch block } catch (ExceptionType3 e3) { // Catch block } THE THROWS/THROW KEYWORDS I f a method does not handle a checked exception, the method must declare it using the throws keyword. BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 5 Try to understand the difference between throws and throw keywords, throws is used to postpone the handling of a checked exception and throw is used to invoke an exception explicitly.

- *; public class className { public void deposit (double amount) throws RemoteException { // Method implementation throw new RemoteException (); } // Remainder of class definition } A met hod can declare that it throws more than one exception, in which case the exceptions are declared in a list separated by commas. *; public class className { public void withdraw (double amount) throws RemoteException , InsufficientFundsException { // Method implementation } // Remainder of class definition } THE FINALLY BLOCK The finally block follows a try bl ock or a catch block. BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 6 A finally block appears at the end of the catch blocks and has the following syntax Syntax try { // Protected code } catch (ExceptionType1 e1) { // Catch block } catch (ExceptionType2 e2) { // Catch block } catch (ExceptionType3 e3) { // Catch block } finally { // The finally block always executes. ArrayIndexOutOfBoundsException: 3 First element value: 6 The finally statement is executed BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 7 Note the following A catch clause cannot exist without a try statement.

- Example: class MyException extends Exception { String s1; MyException(String s2) { s1 = s2; } public String toString() { return ("Ou tput String = "+s1); } } public class NewClass { public static void main(String args[]) { try { throw new MyException("Custom message"); } catch(MyException exp) { System. Though internally FileReader uses FileInputStream and FileWriter uses FileOutputStream but here the major difference is that FileReader reads two bytes at a time and F ileWriter writes two bytes at a time. *; public class CopyFile { public

static void main (String args []) throws IOException { FileReader in = null ; BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 11 FileWriter out = null ; try { in = new FileReader ("input. java $java CopyFile STANDARD STREAMS All the programming languages provide support for standard I/O where the user"s program can take input from a keyboard and then produce an output on the computer screen.

- If you are aware of C or BCA - JAVA PRORAMMING YUVAKSHETRA INSTITUT E OF MANAGEMENT STUD IES BCA 2017 ONWARS BATCH Page 12 C++ programming languages, then you must be aware of three standard devices STDIN, STDOUT and STDERR. Similarly, Java provides the following three standard streams Standard Input This is used to feed the data to user"s program and usu ally a keyboard is used as standard input stream and represented as System. Following constructor takes a file name as a string to create an input stream object to read the file InputStream f = new FileInputStream("C:/java/hello"); Following constructor takes a file object to c reate an input stream object to read the file. First we create a file object using File() method as follows File f = new File("C:/java/hello"); InputStream f = new FileInputStream(f); Once you have InputStream object in hand, then there is a list of help er methods which can be used to read to stream or to do other operations on the stream.

- Following constructor takes a file name as a string to create an input stream object to write the file OutputStream f = new FileOutputStream("C:/java/hello") Following constructor takes a file object to create an output stream object to write the file. First, we create a file object using File() method as follows File f = new File("C:/java/hello"); OutputStream f = new FileOutputStream(f); Once you have OutputStream object in hand, then there is a list of helper methods, which can be used to write to s tream or to do other operations on the stream.