**第4课 Java 的异常处理**

|  |  |  |
| --- | --- | --- |
| **课 题** | Java 的异常处理 | |
| **课 时** | 5课时（225 min）。 | |
| **教学目标** | **知识技能目标：**  1．理解异常处理的概念和意义。  2．掌握异常类的层次结构以及常见的异常类。  **思政育人目标：**  让学生通过学习Java 的异常处理，了解异常处理机制对于编写安全、健壮、简洁的 Java 程序，具有非常重要的作用。 | |
| **教学重难点** | **教学重点：**异常的处理  **教学难点：**自定义异常 | |
| **教学方法** | 讲授法、问答法、讨论法 | |
| **教学用具** | 电脑、投影仪、多媒体课件、教材 | |
| **教学设计** | 第1节课：考勤（2min）--知识讲解（40min）--作业布置（3min）  第2节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第3节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第4节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第5节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min） | |
| **教学过程** | **主 要 教 学 内 容 及 步 骤** | **设计意图** |
| **考勤**  **（2min）** | ■【教师】清点上课人数，记录好考勤  ■【学生】班干部报请假人员及原因 | 培养学生的组织纪律性,掌握学生的出勤情况 |
| **知识讲解**  （40min） | **【教师】**展示异常的处理（一）  **任务描述**  编写一个会引发异常的程序，并对其进行处理。  **任务目标**  1. 能够理解异常、异常处理的概念和意义。  2. 能够掌握异常类的层次结构以及常见的异常类。  3. 能够掌握在 Java 中处理异常的方法。  4. 懂得“逢山开道，遇水搭桥”的道理，做一个能“克服任何困难，勇往直前”的人。  **预备知识**  在程序执行中，我们无可避免地会遇到中断程序正常运行的情况，那么我们应该能对这些情况做些预判并对它做出处理，以增加程序的健壮性。  **一、异常和异常处理**  在 Java 编程语言中，程序可能遇到的轻微的错误条件，即未能在编译时检查出来，在运行时会引起程序中止的一种可由 Java 来解决的错误。这时可以写代码来处理异常并继续执行程序，而不是让程序中断。  在程序执行中，任何中断正常程序流程的异常条件就是错误或异常。它可能由 Java 运行时的系统产生，也有可能是编写的程序自己产生。例如，发生下列情况时，会出现异常：  z 做除法时，除数为零；  z 需要打开的文件不存在；  z 网络连接中断；  z 受控操作数超出预定范围；  例如：  public class Ex0401  {  public static void main (String args[])  {  int d = 0;  int a = 100/d;  System.out.println("a = "+a);  }  }  在本例中，将一个数除以零，这种计算显然是不可能的，所以程序在执行过程中出现了一个 ArithmeticException（算术异常），并导致程序终止。这是 Java 常见异常的一种。  注意：上述所出现的异常情况都能由 Java 来处理的，有些其他错误可严重到不能被解决，这种情况下的错误被称为致命错误，建议中断程序。  Java 处理异常是通过 5 个关键字控制：try、catch、finally、throw 和 throws。  **二、异常类的层次结构**  （1）在 Java 编程语言中，所有的异常都是从 Throwable 继承而来的，它是所有异常的共同祖先。  （2）Throwable 有 两 个 子 类，Error 和 Exception，如图 4−1 所 示。 其 中，Error 是错误。对于所有的编译时期的错误以及系统错误都是通过 Error 抛出的。这些错误表示故障发生于虚拟机自身或者发生在虚拟机试图执行应用时，如 Java 虚拟机运行错误（Virtual MachineError）、类定义错误（NoClassDefFoundError）、堆栈溢出等。这些错误是不可查的，因为它们在应用程序的控制和处理能力之外，而且绝大多数是程序运行时不允许出现的状况。对于设计合理的应用程序来说，即使确实发生了错误，本质上也不应该试图去处理它所引起的异常状况。在 Java 中，错误通过 Error 的子类描述。    （3）Exception 是另外一个非常重要的异常子类。它规定的异常是程序本身可以处理的异常。异常和错误的区别是：异常是可以被处理的，而错误是没法处理的。  （4）运行时异常和非运行时异常。  ①运行时异常：程序中可以不做处理，直接由运行时系统来处理。  ②非运行时异常：在程序中必须对其进行处理，否则编辑器会指出错误。  ③ RuntimeException 表示一种设计或实现问题。也就是说，它表示如果程序运行正常，从不会发生的情况。例如：如果数组索引扩展不超出数组界限，那么，ArrayIndexOutOfBoundsException 异常就不可能被抛出。这也适用于取消引用一个空值对象变量。  因为一个正确设计和实现的程序从不出现这种异常，通常对它不做处理。这会导致一个运行时信息，应确保能采取措施更正问题，而不是将它藏到谁也不注意的地方。  ④其他异常表示一种运行时的困难，它通常由环境效果引起，可以进行处理。例如：包括文件未找到或无效 URL 异常（用户打了一个错误的 URL），如果用户错误地打开了什么东西，两者都容易出现。这两者都可能因为用户的错误而出现，这就需要程序员去处理它们。  **【学生】**思考、讨论。 | **展示异常的处理（一），让学生更加仔细的阅读，从而激发学生的学习欲望。** |
| **作业布置**（3min） | **【教师】**布置课后作业  简述异常类的层次结构。 | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**  （40min） | **【教师】**展示异常的处理（二）  **三、常见的异常类**  在java.lang包中定义了多个异常类，一般是 RuntimeException的子类，表4−1中描述了一些常见异常类。    **四、捕获异常**  Java 编程语言设计的两种对付可控制异常的办法，其中一种就是捕获异常。这种方法由三个程序块组成：try、catch、finally，其中前两个程序块是必须的，第三个是可选项。  **（一）try——发现异常**  我们需要将可能引起异常发生的语句放在 try 语句块中，这是种监控的机制，其格式为：  try  {  //code that might generate exception  }  如果产生异常，它将会被 try 代码块捕获，并送到 catch 代码块处理。  **（二）catch——处理异常**  由 try 代码块捕获的异常必须被处理，捕获异常的程序块叫作 catch 程序块，至少有一个 catch 程序块与 try 程序块成对出现，其格式为：  try  {  //code that might generate exceptions  }  catch(ExceptionType1 ex1)  {  //code to deal with this type of exception  }  当异常被捕获后，程序就会中断执行 try 代码块中的内容，转而执行 catch 代码块中的内容。执行永远不会从 catch 块“返回”到 try 块。若 try 程序块中的程序代码可能引发多个不同种类的异常，针对捕获异常的不同种类就必须使用不同的 catch 代码块来处理，其格式为：  try  {  //code that might generate exceptions  }  catch(ExceptionType1 ex1)  {  //code to deal with this type of exception  }  catch(ExceptionType1 ex2)  {  //code to deal with this type of exception  }  catch(ExceptionType1 ex3)  {  //code to deal with this type of exception  }  当异常被捕获时，异常处理机制将负责搜索参数与异常类型相匹配的第一个处理程序，然后进入 catch 块执行。  注意：当使用多重 catch 代码块时，必须将异常的子类排列在其父类的异常类前面进行捕获，因为使用父类异常类来捕获异常的 catch 代码块将捕获该类型及其所有子类类型的异常。这样的话，如果父类排在前面，则使用子类异常类来捕获异常的 catch 代码块中的内容将永远不会到达，而在 Java 的编译系统中，存在不可能被执行的语句是一种错误。  **（三）finally——清理异常**  当异常被引发后，程序会转入处理异常的代码块进行异常处理，但有时我们还需要给用户一些有用信息或者不希望异常过早返回，这时 Java 提供给我们另一种代码块finally，它出现在 try−catch 块之后。finally 代码块不管异常有没有被引发都会被执行，即使你在 catch 块中，使程序退出，但是在其退出之前，finally 块的代码都会被执行，所以这是相当特殊的代码块。finally 是可选项。  **五、抛出异常**  **（一）throw——产生异常**  在程序的编写中，除了会遇到由 Java 运行时系统引发的异常外，我们也可以使用throw 语句来明确引发指定的异常。在上例中，我们使用了  ArithmeticException a = new ArithmeticException();  throw a;  这条语句明确地引发了一个 ArithmeticException 异常，注意此时需要使用 new 关键字创建一个异常类的对象，其后还需要明确地使用 catch 语句来捕获异常以便对其进行处理。  **（二）throws——方法抛出异常**  我们现在知道引发异常后，必须有相应的处理语句，即使用 catch 语句块，Java语言系统也允许程序员以关键字 throws 来将方法可能会引起的异常抛出，这时在方法中就不必再处理被抛出的异常，但必须由调用这个方法的方法来处理。在“任务实施”EX0402. Java 中，在 main 方法中，调用了 throwsTest 方法，由于这个方法会引发异常，所以在调用时必须将其放在 try 语句块中：  try  {  throwsTest();  }  同时由于这个方法会引发 ArithmeticException 异常 , 所以在其后编写语句：  catch(ArithmeticException e)  {  System.out.println("information:"+e);  }  由其对这个异常进行处理。  现在我们来思考一个问题：如果在 main 方法不处理这个异常应该怎么办？对，也可以将异常抛出，具体语句请同学们自己完成。  **任务实施**  /\* 源程序文件名：Ex0402.java\*/  import java.lang.Math;  public class Ex0402  {  public static void throwTest() throws ArithmeticException  {  ArithmeticException a = new ArithmeticException();  throw a;  }  public static void main (String args[])  {  int x = (int)(Math.random()\*5);  int y = (int)(Math.random()\*10);  int [] z = new int[5];  try  {  System.out.println("y/x gives"+(y/x));  System.out.println("y is"+y+"z[y] is"+z[y]);  throwTest();  }  catch(ArithmeticException e)  {  System.out.println("Division by zero");  }  catch(IndexOutOfBoundsException e)  {  System.out.println("Warning:Out of range");  }  finally  {  System.out.println(" 这条语句始终会被执行 ");  }  System.out.println("after the catch");  }  }  **【学生】**思考、讨论。 | **通过教师讲解，了解异常的处理（二）的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了异常的处理（二），让学生注意此时需要使用 new 关键字创建一个异常类的对象，其后还需要明确地使用 catch 语句来捕获异常以便对其进行处理。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | **【教师】**布置课后作业  下面程序有问题，怎么修改？  public class ThrowsDemo  {  static void throwOne()  {  System.out.println("Inside throwOne.");  throw new IllegalAccessException("demo");  }  public static void main(String args[])  {  throwOne();  }  } | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示自定义异常  **任务描述**  编写程序，处理数值越界异常。  **任务目标**  1. 能够理解自定义异常的概念与意义。  2. 能够掌握在 Java 编程语言中自定义异常的方法以及其在程序中的应用。  3. 能够做一个遇到问题不回避，积极主动处理问题的人，在今后的学习和工作中“踔毅奋发，勇毅前行”。  **预备知识**  在 Java 语言系统中，虽然其内置的异常能处理大多数的错误情况，但有时我们必须自定义一些异常来处理编程中的特殊情况。同时，异常处理机制不仅能处理错误，有时也能利用它来完成自己的一些编程目的。  **一、自定义异常的定义**  自定义异常类并不复杂，首先它是一个类，这就如定义普通类一样，不同之处是这个类是一个异常类，所以它必须继承自类 Exception，也就是说自定义异常类是类Exception 的子类。类 Exception 继承了类 Throwable 中定义的一些方法，因此编程人员可根据需要覆盖这些方法，相关方法请参阅 Java 的 API 说明书。下面定义 Exception 的一个子类 myExcetpoin：  public class myException extends Exception{}  同时我们定义了它的构造函数，也覆盖了 toString( ) 方法，以处理这个异常。  **二、处理自定义异常**  由于自定义异常不是系统定义的，所以肯定不能由系统引发，因此编程人员必须自行编写判断条件，并在可能引发时将异常抛出，以便让 catch 语句块处理。  **任务实施**  /\* 源程序文件名：Ex0403.java\*/  class MyException extends Exception  {  private int i;  MyException(int i)  {  this.i = i;  }  public String toString()  {  return " 数值越界异常 ("+i+")";  }  }  public class Ex0403  {  public static void test(int i) throws MyException  {  if (i>100)  {  throw new MyException(i);  }  }  public static void main(String args[])  {  try  {  test(10);  test(200);  }  catch(MyException e)  {  System.out.println(e.toString());  }  }  }  程序运行结果如图 4−3 所示。    这个例子中，定义了一个名叫 MyException 的自定义异常，在类 Ex0403 中又定义了一个方法 test，该方法会对数值大小进行判断，若超出 100 则会抛出一个 MyException异常，在 main 中调用 test 方法，并传入大于 100 的参数以引发异常，进入异常处理的语句块，完成任务。  **【学生】**思考、讨论。 | **教师通过自定义异常展示，让学生了解自定义异常的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了自定义异常，异常处理机制不仅能处理错误，有时也能利用它来完成自己的一些编程目的。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  如何处理自定义异常？ | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示思考与练习  **一、简答题**  （1）Java 中处理异常机制的结构是怎样的？各语句块的功能是什么？  （2）抛出异常时如果不在当前的调用该方法的方法中处理，我们还能怎么办？  （3）Java 中异常的层次结构是怎样的（画出层次图）？  （4）阅读下面的程序，并回答执行的结果是什么？  class SubCatch  {  public static void main(String args[])  {  try  {  int a = 0;  int b = 100/a;  }  catch(Exception e)  {  System.out.printly(" 父类捕获异常 ");  }  catch(ArithmeticException e)  {  System.out.printly(" 子类捕获异常 ");  }  }  }  **二、选择题**  （1）给定下列 Java 代码，运行时会产生（　　）类型异常。  int a = 0,d;  d = 10/a;  System.out.println(d);  A. NullPointerException B. IOException  C. ArithmeticException D. Error  （2）在 Java 的一个异常处理中，语句块可以有多个的是（　　）。  A. catch B. finally  C. try D. throws  （3）在程序运行过程中抛出异常但不处理的关键字是（　　）。  A. try B. catch  C. throw D. throws  （4）在程序运行过程中抛出异常的语句是（　　）。  A. try B. catch  C. throw D. throws  （5）对于 catch 子句的排列，下列说法正确的是（　　）。  A. 父类在前，子类在后  B. 子类在前，父类在后  C. 随便排列  D. 有继承关系的异常不能在同一个 try 程序段内  （6）关于 try 语句，下列说法正确的是（　　）。  A. try 语句中，必须得有 catch  B. try 语句中必须得有 finaly  C. try 语句中的 catch 在程序的最后会被执行  D. 一个 try 只能有一个 catch  （7）关于 try，catch，finally，下列说法正确的是（　　）。  A. catch 在程序的最后会被执行  B. 执行永远不会从 catch 块“返回”到 try 块  C. 有 try 语句，必须得有 catch  D. 如果程序中有多个 catch 块，要求父异常排在前，子异常排在后  **【学生】**思考、讨论。 | **教师通过思考与练习展示，让学生了解思考与练习的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了思考与练习，让学生可以巩固自己所学知识。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  下列选项中能正确地抛出异常的语句是（　　）。  A. Exception e=new IOException（“File not found”）; if（!f.exists( )）{ throw e; }  B. if（!f.exists( )）{ throw new IOException（“File”+f.getName( )+“notfound”）; }  C. if（!f.exists( )）{ throw“file not found”; }  D. if（!f.exists( )）{ throw new IOException( ); } | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**展示实训任务  （1）编写程序：定义类 myException，为这个类编写一个接受字符串参数的构造函数，把此参数保存在对象内部的字符串中。编写一个方法用于输出此字符串。编写一个try−catch 语句块，对这个新异常进行测试。  （2）分别编写能产生非法使用空引用，算术错误，数组越界错误，强制转换类型异常的代码，并将其捕获。  （3）定义异常 A、B、C，其中 B 继承于 A，C 继承于 B。写一个方法，该方法抛出这三种异常，并在 main 方法中，使用三个 catch 语句来捕获这三种异常，注意语句间顺序。你能用一个语句来捕获它们吗？  **【学生】**思考、讨论。 | **教师通过实训任务展示，让学生了解实训任务的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了实训任务，可以更好的增强学生的实际操作能力。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **编写程序：为数组赋值，并处理数组越界时可能引发异常。** | 通过课后练习，使学生巩固所学新知识 |
| **教学反思** | 健康的信息文化素养的形成、网络道德等网络文化规范的潜移默化，应该在不断的信息实践中产生，而不是在简单的说教中产生。 | |