**每日作业卷答案**

**就业班JavaSE第5天**

传智播客.黑马程序员

**基础题**

### 练习一：异常的体系

问题：

1. 请描述异常的继承体系
2. 请描述你对错误(Error)的理解
3. 请描述你对异常(Expection的理解)
4. 请描述你对运行时异常(RuntimeException)的理解

答：

1. 异常继承体系为：异常的根类是 java.lang.Throwable，其下有两个子类：

java.lang.Error 与 java.util.Exception 。而Exception又分为编译时期异常：checked异常，与运行时期异常：runtime异常。

1. Error:表示不可修复的恶性的错误，只能通过修改代码规避错误的产生，通常是系统级别的，所以很严重。
2. Exception:表示可修复的良性（相对于错误）的异常，异常产生后程序员可以并且应该通过代码的方式纠正，使程序继续运行，是必须要处理的。
3. 运行时期异常:runtime异常。在运行时期,检查异常.在编译时期,运行异常不会编译器检测(不报错)。

### 练习二：throw与throws的区别

问题：

1. 请描述throw的使用位置,作用是什么?
2. 请描述throws的使用位置,作用是什么?

答：

1. throw关键字通常用在方法体中，并且抛出一个异常对象。程序在执行到throw语句时立即停止，它后面的语句都不执行。
2. throws关键字通常被应用在声明方法时，用来指定可能抛出的异常。多个异常可以使用逗号隔开。当在主函数中调用该方法时，如果发生异常，就会将异常对象抛给方法调用处。

### 练习三：异常的处理方式

问题：

1. 异常处理方式有几种,分别是什么?

2. 详细阐述每种方式对异常是如何处理的

答：

1. 异常的处理方式有两种,分别是使用throws和try...catch...finally
2. throws用在方法的声明上后接异常类名,是把异常抛给调用者进行处理
3. try...catch...finally是捕获异常,自己处理,处理完毕后面的程序可以继续运行
   1. try代码块中是可能出现异常的代码
   2. catch代码块,是遇到异常,对异常进行处理的代码
   3. finally代码块是无论是否发生异常,都必须执行的代码,用于释放资源.

### 练习四：常见异常，及产生原因

问题：请列举常见异常，并说明产生原因。

答：

**NullPointerException:空指针异常。**

当应用试图在要求使用对象的地方使用了null时，抛出该异常。譬如：调用null对象的实例方法、访问null对象的属性、计算null对象的长度等等。

**ArrayIndexOutOfBoundsException:数组索引越界异常。**

当对数组的索引值为负数或大于等于数组大小时抛出此异常。

**ArithmeticException:算术运算异常。**

程序中出现了除以零这样的运算就会出这样的异常，对这种异常，大家就要好好检查一下自己程序中涉及到数学运算的地方，公式是不是有不妥了。

**NumberFormatException:数字格式异常。**

当试图将一个String转换为指定的数字类型，而该字符串确不满足数字类型要求的格式时，抛出该异常。

**注意：答案不唯一，合理即可。**

### 练习五：看代码，分析结果

问题：根据给出的相应代码，分析可能产生的结果。

1.举例：  
**public static void** main(String[]args){  
  
 String str=**null**;System.out.println(str.length());  
  
}  
  
答：变量str的值为null，调用方法时，报空指针异常NullPointerException  
  
  
  
2.举例：  
**public static void** main(String[]args){  
**int** arr[]={1,2};System.out.println(arr[2]);  
}  
  
答：索引值2大于等于数组arr的长度时，报数组索引越界异常ArrayIndexOutOfBoundsException  
  
  
  
3.举例：  
**public static void** main(String[]args){System.out.println(1/0);  
}

答：整数0做了分母，报算术运算异常ArithmeticException:/by zero  
  
  
  
4.举例：  
**public static void** main(String[]args){System.out.println(Integer.parseInt(**"itcast"**));  
  
}  
  
答：

把字符串“itcast”转换为Integer类型时，当然会报数字格式化异常啦NumberFormatException

5.举例：  
**public static void main(String[] args) {  
 SimpleDateFormat format = new SimpleDateFormat("yyyy-mm-dd");  
   
 try {  
 Date date = format.parse("2018-04-03");  
 System.*out*.println("程序正常");  
   
 } catch (ParseException e) {  
 System.*out*.println("程序异常");  
 }  
}**

答：

打印结果“程序正常”.try代码块中并没有产生异常，catch代码块中的代码不会执行。date为2018年1月3日00点04分00秒。

### 练习六：并行、并发概念

问题：请简单描述什么是并行，什么是并发?

答：

并行：指两个或多个事件在同一时刻发生（同时发生）。

并发：指两个或多个事件在同一个时间段内发生。

通俗易懂版：

你吃饭吃到一半，电话来了，你一直到吃完了以后才去接，这就说明你不支持并发也不 支持并行。  
 你吃饭吃到一半，电话来了，你停了下来接了电话，接完后继续吃饭，这说明你支持并发。  
 你吃饭吃到一半，电话来了，你一边打电话一边吃饭，这说明你支持并行。

并发的关键是你有处理多个任务的能力，不一定要同时。  
 并行的关键是你有同时处理多个任务的能力。

所以它们最关键的点就是：是否是『同时』。

### 练习七：进程概念、线程概念、线程与进程联系

问题：请描述什么是进程，什么是线程，进程与线程之间的关系，并举例说明。

答：

进程指正在运行的程序。确切的来说，当一个程序进入内存运行，即变成一个进程，进程是处于运行过程中的程序，并且具有一定独立功能。

线程是进程中的一个执行单元，负责当前进程中程序的执行，一个进程中至少有一个线程。一个进程中是可以有多个线程的，这个应用程序也可以称之为多线程程序。

一个程序运行后至少有一个进程，一个进程中可以包含多个线程,但一个进程中至少包含一个线程。比如使用迅雷软件下载网络文件时，同时下载多个文件，就使用到了多线程下载。

### 练习八：自定义异常类

问题：

请使用代码实现

每一个学生(Student)都有学号,姓名和分数,分数永远不能为负数

如果老师给学生赋值一个负数,抛出一个自定异常

答：

*/\*  
 1.定义异常类NoScoreException,继承RuntimeException  
a)提供空参和有参构造方法  
 \*/*

|  |
| --- |
| **public class** NoScoreException **extends** RuntimeException { *// 空参构造* **public** NoScoreException() { **super**();  } *// 有参构造* **public** NoScoreException(String message) { **super**(message);  } } |

*/\*  
 2.定义学生类(Student)  
 a)属性:name,score  
 b)提供空参构造  
 c)提供有参构造;  
 i.使用setXxx方法给名称和score赋值  
 d)提供setter和getter方法  
 i.在setScore(int score)方法中  
 1.首先判断,如果score为负数,就抛出NoScoreException,异常信息为:分数不能为负数:xxx.  
 2.然后在给成员score赋值.  
 \*/*

|  |
| --- |
| **public class** Student { **private** String **name**; **private int score**; *// 空参构造* **public** Student() { **super**();  } *// c)提供有参构造; // i.使用setXxx方法给名称和score赋值* **public** Student(String name,**int** score){  setName(name);  setScore(score);  } *// d)提供setter和getter方法* **public** String getName() { **return name**;  }  **public void** setName(String name) { **this**.**name** = name;  }  **public int** getScore() { **return score**;  } *// i.在setScore(int score)方法中* **public void** setScore(**int** score) { *// 1.首先判断,如果score为负数,就抛出NoScoreException,异常信息为:分数不能为负数:xxx.* **if**(score <0){ **throw new** NoScoreException(**":分数不能为负数:"**+score);  } *// 2.然后在给成员score赋值.* **this**.**score** = score;  } } |

*/\*  
3.定义测试类Test9  
 a)提供main方法,在main方法中  
 i.使用满参构造方法创建Student对象,分数传入一个负数,运行程序  
 ii.由于一旦遇到异常,后面的代码的将不在执行,所以需要注释掉上面的代码  
 iii.使用空参构造创建Student对象  
 iv.调用setScore(int score)方法,传入一个正数,运行程序  
 v.调用setScore(int score)方法,传入一个负数,运行程序  
 \*/*

|  |
| --- |
| **public class** Test9 { **public static void** main(String[] args) { *// i.使用满参构造方法创建Student对象,分数传入一个负数,运行程序 // Student s = new Student("景甜", -10); // ii.由于一旦遇到异常,后面的代码的将不在执行,所以需要注释掉上面的代码  // iii.使用空参构造创建Student对象* Student s = **new** Student(); *// iv.调用setScore(int score)方法,传入一个正数,运行程序* s.setScore(100); *// v.调用setScore(int score)方法,传入一个负数,运行程序* s.setScore(-5);  } } |

### 练习九：多线程

问题：

创建多线程对象，开启多线程。在子线程中输出1-100之间的偶数，主线程输出1-100之间的奇数。

答：

**自定义线程类：**

|  |
| --- |
| **public class** MyThread **extends** Thread {  */\*\*  \* 重写run方法，完成该线程执行的逻辑 \*/* @Override **public void run() {  for (int i = 1; i <= 100; i++) {  if (i % 2 == 0) {  System.*out*.println("子线程打印输出偶数：" + i);  }  } }**  } |

**测试类：**

|  |
| --- |
| **public class** Test11 { **public static void** main(String[] args) { *//创建自定义线程对象* MyThread mt = **new** MyThread(); *//开启线程* mt.start(); *//在主方法中执行for循环* **for (int i = 1; i <= 100; i++) {  if (i % 2 == 1) {  System.*out*.println("主线程打印输出奇数：" + i);  } }**  } } |

## 扩展题

### 练习十：自定义异常类

问题：

请用代码描述:

在一款角色扮演游戏中,每一个人都会有名字和生命值;角色的生命值不能为负数

要求:当一个人物的生命值为负数的时候需要抛出自定的异常

答：

**操作步骤描述**

1. 创建包com.itheima.level2
2. 自定义异常类NoLifeValueExption继承RuntimeException
   1. 提供空参和有参构造
   2. 在有参构造中,需要调用父类的有参构造,把异常信息传入
3. 定义Person类
   1. 属性:名称(name)和生命值(lifeValue)
   2. 提供空参构造
   3. 提供有参构造;
      1. 使用setXxx方法给name和lifeValue赋值
   4. 提供setter和getter方法
      1. 在setLifeValue(int lifeValue)方法中
         1. 首先判断,如果 lifeValue为负数,就抛出NoLifeValueException,异常信息为:生命值不能为负数:xxx.
         2. 然后在给成员lifeValue赋值.

3. 定义测试类Test10

* 1. 提供main方法,在main方法中
     1. 使用满参构造方法创建Person对象,分数传入一个负数,运行程序
     2. 由于一旦遇到异常,后面的代码的将不在执行,所以需要注释掉上面的代码
     3. 使用空参构造创建Person对象
     4. 调用setLifeValue(int lifeValue)方法,传入一个正数,运行程序
     5. 调用setLifeValue(int lifeValue)方法,传入一个负数,运行程序

**操作步骤答案**

*/\*  
 1.自定义异常类NoLifeValueExption继承RuntimeException  
 a)提供空参和有参构造  
 b)在有参构造中,需要调用父类的有参构造,把异常信息传入  
 \*/*

|  |
| --- |
| **public class** NoLifeValueExption **extends** RuntimeException { **public** NoLifeValueExption() { **super**();  }  **public** NoLifeValueExption(String message) { **super**(message);  } } |

*/\*  
 2.定义Person类  
 a)属性:名称(name)和生命值(lifeValue)  
 b)提供空参构造  
 c)提供有参构造;  
 i.使用setXxx方法给name和lifeValue赋值  
 d)提供setter和getter方法  
 i.在setLifeValue(int lifeValue)方法中  
 1.首先判断,如果 lifeValue为负数,就抛出NoLifeValueException,异常信息为:生命值不能为负数:xxx.  
 2.然后在给成员lifeValue赋值.  
 \*/*

|  |
| --- |
| **public class** Person { *// 名称(name)* **private** String **name**; *// 生命值(lifeValue)* **private int lifeValue**;  *// 空参构造* **public** Person() { **super**();  }  *// c)提供有参构造;* **public** Person(String name, **int** lifeValue) { **super**(); *// i.使用setXxx方法给name和lifeValue赋值* setName(name);  setLifeValue(lifeValue);  }  *//d)提供setter和getter方法* **public** String getName() { **return name**;  }  **public void** setName(String name) { **this**.**name** = name;  }  **public int** getLifeValue() { **return lifeValue**;  }  *// i.在setLifeValue(int lifeValue)方法中* **public void** setLifeValue(**int** lifeValue) { *// 1.首先判断,如果 lifeValue为负数,就抛出NoLifeValueException,异常信息为:生命值不能为负数:xxx.* **if** (lifeValue <0) { **throw new** NoLifeValueExption(**"生命值不能为负数:"** + lifeValue);  } *// 2.然后在给成员lifeValue赋值.* **this**.**lifeValue** = lifeValue;  } } |

*/\*  
3.定义测试类Test10  
 a)提供main方法,在main方法中  
 i.使用满参构造方法创建Person对象,分数传入一个负数,运行程序  
 ii.由于一旦遇到异常,后面的代码的将不在执行,所以需要注释掉上面的代码  
 iii.使用空参构造创建Person对象  
 iv.调用setLifeValue(int lifeValue)方法,传入一个正数,运行程序  
 v.调用setLifeValue(int lifeValue)方法,传入一个负数,运行程序  
 \*/*

|  |
| --- |
| **public class** Test10 { **public static void** main(String[] args) { *// i.使用满参构造方法创建Person对象,分数传入一个负数,运行程序 // Person p = new Person("林思意", -100); // ii.由于一旦遇到异常,后面的代码的将不在执行,所以需要注释掉上面的代码 // iii.使用空参构造创建Person对象* Person p = **new** Person(); *// iv.调用setLifeValue(int lifeValue)方法,传入一个正数,运行程序* p.setLifeValue(1000); *// v.调用setLifeValue(int lifeValue)方法,传入一个负数,运行程序* p.setLifeValue(-100);  } } |

### 练习十一：多线程

问题：

创建三个子线程，在每个线程中开启10万次的循环，线程1循环中将循环自增变量i赋值给Integer类型变量 a，线程2循环中将字符串"黑马程序员"赋值给String类型变量b,线程3循环中将字符串"黑马程序员"和循环自增变量i拼接后赋值给String类型变量c

分别计算三个线程完成任务所用的毫秒值

答：

线程1

|  |
| --- |
| **public class Thread1 extends Thread{  @Override  public void run() {  long start = System.*currentTimeMillis*();  for (int i = 0; i < 100000; i++) {  Integer a = i;  }  long end = System.*currentTimeMillis*();  System.*out*.println("线程1执行时间："+(end-start));  } }** |

线程2

|  |
| --- |
| **public class Thread2 extends Thread{  @Override  public void run() {  long start = System.*currentTimeMillis*();  for (int i = 0; i < 100000; i++) {  String b = "黑马程序员";  }  long end = System.*currentTimeMillis*();  System.*out*.println("线程2执行时间："+(end-start));  } }** |

**线程3**

|  |
| --- |
| **public class Thread3 extends Thread{  @Override  public void run() {  long start = System.*currentTimeMillis*();  for (int i = 0; i < 100000; i++) {  String c = "黑马程序员"+i;  }  long end = System.*currentTimeMillis*();  System.*out*.println("========线程3执行时间："+(end-start));  } }** |

测试类

|  |
| --- |
| **public class TestThread {  public static void main(String[] args) {  new Thread1().start();  new Thread2().start();  new Thread3().start();  } }** |