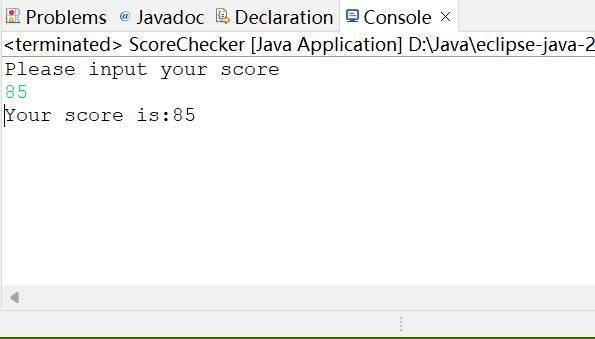
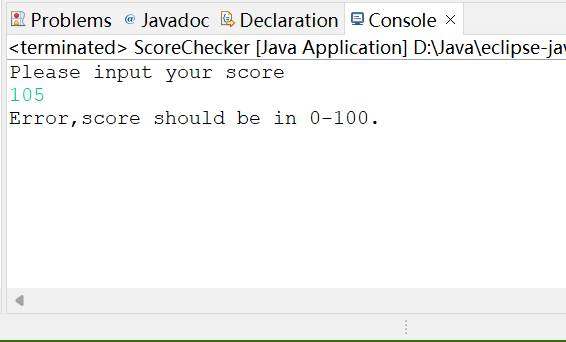
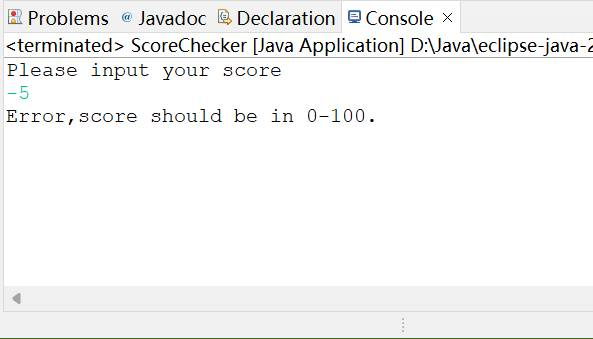
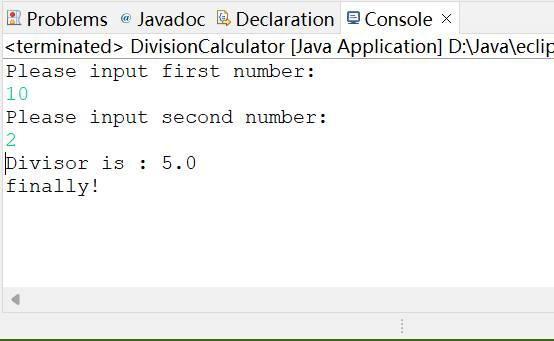
**题目①**

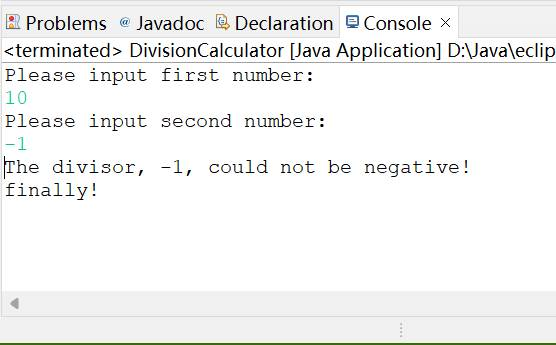
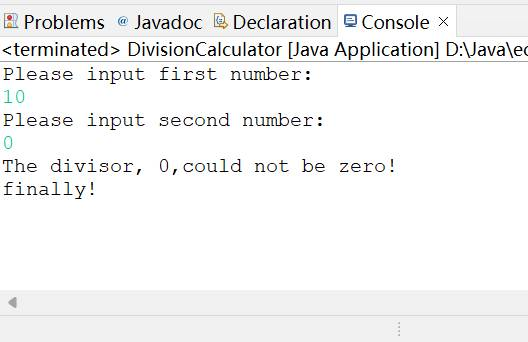
**一、源程序调试过程**  
1、初次编写时，未处理输入非整数的情况，导致 InputMismatchException。解决方法是在捕获 MyException 同时捕获 InputMismatchException。  
2、在检查分数范围时，逻辑错误导致即使分数在范围内也抛出异常。修正了 checkScore 函数中的逻辑判断，确保仅在分数超出0-100时抛出异常。  
  
**二、实验实习结果分析**  
1、输入正常分数（如85），程序正确输出：“Your score is:85”。  
  
2、输入异常分数（如-5, 105），程序正确输出错误信息：“Error, score should be in 0-100.”  


3、输入非数字字符，程序现捕获并处理异常，提示输入错误。

**三、心得体会**  
通过本次实验，我学到了异常处理的重要性和基本用法。自定义异常类可以针对特定错误情况提供更清晰的错误信息，提高程序的健壮性和用户体验。

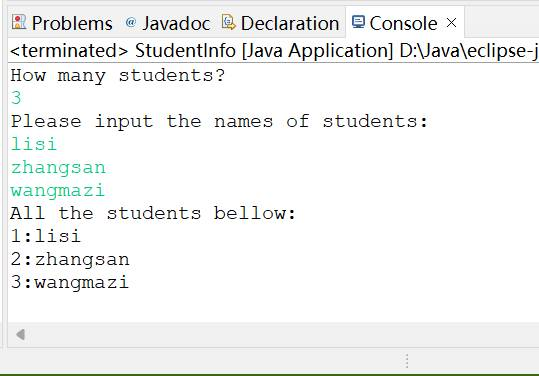
**题目②**

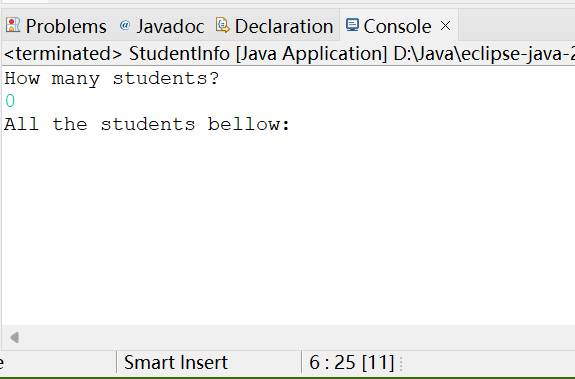
**一、源程序调试过程**  
1、最初忘记处理除数为零的情况，导致程序抛出 ArithmeticException。后增加了 ZeroException 的检测和处理。  
2、对于除数为负数的情况，原先只是简单地抛出异常，没有给出具体的错误信息。后改进为在异常信息中指明具体的错误除数值。  
  
**二、实验实习结果分析**  
1、输入正常数值进行除法操作，例如第一个数为10，第二个数为2，程序正确输出商为5.0。  
2、输入第二个数为零或负数，程序正确抛出异常并给出相应提示信息。

  
  
  
**三、心得体会**  
本实验使我深刻理解了异常处理在输入验证中的应用。自定义异常允许我们对不同的错误情况做出具体且适当的响应，增加了程序的可用性和用户的操作体验。

**题目③**

**一、源程序调试过程**  
1、初次实现时未考虑用户可能输入为0的学生数量，添加了对此情况的判断和处理。  
2、实现过程中未处理ArrayList的索引，导致错误地从0开始编号学生。后调整输出逻辑，从1开始编号。  
  
**二、实验实习结果分析**  
1、输入不同数量的学生名字，程序能正确显示学生信息，例如输入3个学生名，输出格式正确。

  
2、输入为0时，程序现在不再进行无意义的名字输入提示，而是直接显示无学生信息。

  
  
  
**三、心得体会**  
这次实验让我了解了ArrayList的使用，以及如何处理用户交互输入并与异常类处理结合。学会了如何更加灵活地处理用户输入及其对应的输出格式，提升了编程的细节处理能力。