

**Course Experiment Report**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Course:** | Java Language | | | | | | |
|  |  | | | | | | |
| **Semester:** | 1-18th | **week** | 2nd | **year** | | 1st | **term** |
|  |  |  |  |  | |  |  |
| **Major:** | Software Engineering | | | | | **Class:** | 2019 |
|  |  | | | | |  |  |
| **Student name:** | Fu Ruoxuan | | **Student No.:** | | 222019321062060 | | |
|  |  | |  | |  | | |
| **Teacher:** | Wang Xiaomeng | | | | | | |

College of Computer and Information Science

|  |  |  |  |
| --- | --- | --- | --- |
| Project | Exp5 Exception Handling | | |
| Time | 2020.11.25 | Type | □Verification □Design ☑Synthetical |
| 1. Answer the questions  (1) Under what conditions will the finally block be executed?  The Finally keyword in Java is generally used in conjunction with try. After a program enters the try block, the contents of the finally block will be executed whether the program is terminated by an exception or otherwise returned to terminate. It mostly used under the situation of relieving memory spaces.  (2) What is the difference between the keyword throw and throws?  “throw” is specifically an outgoing exception that throws an instance of an exception or a reference of an exception. “throws” declares what type of exception it is, allowing its caller to catch it.  (3) Can the main method declare an exception?  The main method can declare an exception. The exception which is in the main function goes to the default exception handler of the thread, which exists anyway because unchecked exceptions can also happen. It's considered best to handle exception in main itself, but you can just leave it to the default exception handler  (4) Other experience.  ①printStackTrace()  In the case of throwing an exception, there are many methods, System.out.println(e).This method prints the exception and outputs where it occurred; e.printStackTrace() also prints the exception, but it will also display deeper information about the call.  2. All Codes  IntegerInput.java   1. **import** java.util.InputMismatchException; 2. **import** java.util.Scanner; 4. **public** **class** IntegerInput { 5. **public** **static** **void** main(String[]args) { 6. sumTwoIntegers(); 7. } 8. **private** **static** **void** sumTwoIntegers(){ 9. Scanner input = **null**; 10. **try**{ 11. input = **new** Scanner(System.in); 12. System.out.println("Please input the first integer"); 13. **int** first = input.nextInt(); 14. System.out.println("Please input the second integer"); 15. **int** second = input.nextInt(); 16. System.out.printf("The sum is %d",first+second); 17. }**catch**(InputMismatchException ex){ 18. System.out.println("Input does not match the integer type,please enter again!"); 19. sumTwoIntegers(); 20. }**finally** { 21. input.close(); 22. } 23. } 24. }   IllegalTriangle.java   1. **import** java.util.Scanner; 3. **public** **class** IllegalTriangle { 5. **public** **static** **void** main(String[]args){ 6. Scanner input = **new** Scanner(System.in); 7. System.out.println("Please input the first side"); 8. **double** side1 = input.nextDouble(); 9. System.out.println("Please input the second side"); 10. **double** side2 = input.nextDouble(); 11. System.out.println("Please input the third side"); 12. **double** side3 = input.nextDouble(); 13. **try**{ 14. Triangle t = **new** Triangle(side1,side2,side3); 15. } **catch** (IllegalTriangleException e){ 16. e.printStackTrace();//output the position and causes of the exception 17. } 18. } 20. } 22. **class** Triangle{ 23. **private** **double** side1; 24. **private** **double** side2; 25. **private** **double** side3; 27. Triangle(){ 28. side1 = 1.0; 29. side2 = 2.0; 30. side3 = 3.0; 31. } 32. Triangle(**double** side1,**double** side2,**double** side3) **throws** 33. IllegalTriangleException{ 34. **this**.side1=side1; 35. **this**.side2=side2; 36. **this**.side3=side3; 37. **if** (side1<=0||side2<=0||side3<=0) 38. **throw** **new** IllegalTriangleException("one side is less than 0"); 39. **else** **if** (side1+side3<=side2) 40. **throw** **new** IllegalTriangleException("side1 + side3 <= side2"); 41. **else** **if** (side1+side2<=side3) 42. **throw** **new** IllegalTriangleException("side1 + side2 <= side3"); 43. **else** **if** (side2+side3<=side1) 44. **throw** **new** IllegalTriangleException("side2 + side3 <= side1"); 45. **else** { 46. System.out.println("The input is legal!"); 47. System.out.println(**this**.toString()); 48. System.out.println("The area is " + **this**.getArea()); 49. System.out.println("The Perimeter is " + **this**.getPerimeter()); 50. } 51. } 53. **public** **double** getSide1() { 54. **return** side1; 55. } 57. **public** **double** getSide2() { 58. **return** side2; 59. } 61. **public** **double** getSide3() { 62. **return** side3; 63. } 65. **public** **double** getArea() { 66. **return** 0.25 \* Math.sqrt((side1 + side2 + side3)\*(side1 + side2 - side3)\*(side1 + side3 - side2)\*(side2 + side3 - side1)); 67. } 69. **public** **double** getPerimeter(){ 70. **return** side1 + side2 + side3; 71. } 73. **public** String toString(){ 74. **return** "Triangle: side1 = " + side1 + " side2 = " + side2 + " side3 =" + side3; 75. } 76. } 78. **class** IllegalTriangleException **extends** Exception{ 79. **public** IllegalTriangleException(){ 80. **super**(); 81. } 82. **public** IllegalTriangleException(String str){ 83. **super**(str); 84. } 85. } | | | |
|  | | | |

|  |  |  |
| --- | --- | --- |
| Evaluation | Code Correctness (60%): |  |
| Experience (40%): |  |
| Score： | |