1. **(Save as printSquareMain.java)**
   Write a method called printSquare that accepts a minimum and maximum integer and prints a square of lines increasing numbers. The first line should start with the minimum, and each line that follows should start with the next-higher number. The sequence of numbers on a line wraps back to the minimum after it hits the maximum. For example, the call

   printSquare(3, 7)

   should produce the following output:

   34567
   45673
   56734
   67345
   73456

2. **(Save as largestAbsValMain.java)**
   Write a method called largestAbsVal that takes three integers as parameters and returns the largest of their three absolute values. For example, a call of largestAbsVal(7, -2, -11) would return 11, and a call of largestAbsVal(-4, -5, 2) would return 5.

3. **(Save as verticalMain.java)**
   Write a method called vertical that accepts a string as its parameter and prints each letter of the string on separate lines. For example, a call of vertical("object oriented") should produce the following output:

   o
   b
   j
   e
   c
   t

   o
   r
   i
   e
   n
   t
   e
   d

4. **(Save as printTriangleTypeMain.java)**
Write a method called printTriangleType that accepts three integer arguments representing the lengths of the sides of a triangle and prints the type of triangle that these sides form. The three types are equilateral (three sides of the same length), isosceles (two sides that are the same length) and scalene (three sides of different lengths).

Here are the sample calls to printTriangleType:

printTriangleType(5, 7, 7)
printTriangleType(6, 6, 6)
printTriangleType(5, 7, 8)
printTriangleType(2, 18, 2)

The output produced by these calls should be

Isosceles
Equilateral
Scalene
Isosceles

Your method should throw an IllegalArgumentException if passed invalid values, such as ones where one side's length is longer than the sum of the other two, which is impossible in a triangle. For example, the call of printTriangleType(2, 18, 2) should throw an exception.

5. **(Save as swapPairsMain.java)**
Write a method called swapPairs that accepts a String as a parameter and returns that String with each pair of adjacent letters reversed. If the String has an odd number of letters, the last letter is unchanged. For example, the call swapPairs("example") should return "xemalpe" and the call swapPairs("hello there") should return "ehll ohtree".