Exception Handling

Please create your own Java Exception, InvalidDivideException. This exception will be used to signal your test class that an invalid entry has been received by the Divide class.

Complete the division method in Divider.java. This method should check to see if the divisor is 0. If it is, throw an InvalidDivideException which must be handled in your DivideTester. You are not allowed to change the signature of division.

DivideTester should call the division method in the Divide class. You should print the numbers being passed to the division method and the return result. When you pass a 0 value as the divisor, your exception must be thrown by the division method and caught in DivideTester. You should build the appropriate message in the division method when instantiating the exception. In DivideTester, handle the exception and print the message.

Sample run:

```
The result of 5 divided by 2 is 2.50
The result of 4 divided by 3 is 1.33
The result of 2 divided by 8 is 0.25
Attempted to divide by 0.  Problem was 4/0
The result of 0 divided by 5 is 0.00
```

Submit DivideTester.java, Divider.java, InvalidDivideException.java.

File I/O

Create a program that reads in the text file, input.txt. Your program must reverse the letters on each line as well as reversing the order of the lines in the file. For example, if input.txt contains:

```
esruoc siht ssap ot tnaw I
!!!A na htiw
```

Your file output file should contain

```
I want to pass this course
with an A!!!
```

Submit FileReverse.java.

Inheritance/Polymorphism

1. You have been given the Pet, Dog, and Cat classes. Pet is the superclass for Dog and Cat.
2. Create a subclass of Pet named Rodent. **Do not change Pet, Dog, or Cat.** The rodent will add an attribute, hasWheel that holds true or false.
3. You must supply a no argument constructor and a constructor for all attributes of rodent.
4. Create the correct getters and setters in the Rodent subclass.
5. Create a toString() in Rodent. It must use the parent toString() to display data.
6. Override the eats() method in Pet. A rodent "eats rodent food". The method declaration for eats is:
   public String eats()
7. Create a PetTester class. Add 2 cats, 2 dogs and 2 rodents to an ArrayList. Print your pets. You must use a single ArrayList to hold your pets.
8. Test your new eats() method by printing the results of calling this method.

Sample Output:
```
Cat  [name = Mitten, breed = Siamese, color = gray, number of legs = 4, weight = 15.
Mitten likes mice. ]
 eats cat food
Cat  [name = Tim, breed = Maine Coon Cat, color = tabby, number of legs = 4, weight
= 25.  Tim hates mice.]
 eats cat food
Dog  [name = Dusty, breed = Golden Retriever, color = gold, number of legs = 4,
weight = 95 trained = true, size = large]
 eats dog food
Dog  [name = Kenai, breed = Australian Shepherd, color = Red Merl, number of legs =
4, weight = 75 trained = false, size = Medium]
 eats dog food
Rodent [name = Squeaky, breed = hamster, color = brown, number of legs = 4,
weight = 1 Has Wheel = true]
eats rodent food
Rodent [name = Oinker, breed = pig, color = pink, number of legs = 4, weight = 100
Has Wheel = false]
eats rodent food
```

Submit Pet.java, Cat.java, Dog.java and Rodent.java.