Part 1

In this lab, you will implement a `VendingMachine` class that holds cans of soda. To buy a can of soda, the customer needs to insert a token into the machine. When the token is inserted, a can drops from the can reservoir into the product delivery slot. The vending machine can be filled with more cans. The goal is to determine how many cans and tokens are in the machine at any given time.

The instance variables could be

```
private int numCans;
private int tokens;
```

The methods you would supply for a `VendingMachine` class will be but not limited to

```
public void fillUp(int cans){. . .}
public void insertToken( ){. . .}
public void insertToken(int k){. . .}

public int getCanCount( ){. . .}
public int getTokenCount( ){. . .}
```

The method `fillUp(int cans)` is to add more cans to the machine. Simply add the number of new cans to the can count.

Do not worry about the case where there are no more cans in the vending machine. For now, assume that the `insertToken( )` and `insertToken(int k)` methods will not be called if the vending machine is empty or there is not enough cans left in the vending machine. This method should decrement the number of cans in the vending machine and increment the number of tokens after one token is inserted.

Part 2

You need to implement a testing class to test the `VendingMachine` class

```
// The following is a template of a testing program.  Do not copy!!!
public class VendingMachineTester {
   public static void main(String[] args) {
      VendingMachine machine = new VendingMachine();
      machine.fillUp(10); // Fill up with ten cans
      machine.insertToken();
      machine.insertToken();
      machine.insertToken(5);

      . . . . . .
      System.out.print("Token count: ");
      System.out.println(machine.getTokenCount());
      System.out.println("Expected: . . .");
      System.out.print("Can count: ");
      System.out.println(machine.getCanCount());
      System.out.println("Expected: . . .");
   }
}
```