BankTransaction - AP FRQ

Assume that a class called `BankAccount` has been implemented to represent one person's bank account. A partial declaration of the `BankAccount` class is given below.

```
public class BankAccount {

      //fields
      private int accountNum;
      private double balance;

      //constructor not shown

      //methods
      //returns the account number
      public int getAccountNum( ) { return accountNum; }

      //returns the current balance - how much money is in the
account
      public double getBalance ( ) {  return balance; }

      //adds the given amount to the current balance
      public void doDeposit (double amount ) { balance += amount; }

      //subtracts the given amount from the current balance
      public void doWithdrawal ( double amount ) { balance -= amount; }
}
```

Part A:

In order to process the various transactions performed at the bank (either by ATM or bank teller), a `Transaction` class is needed: a `Transaction` includes an account number (an integer), the transaction type (a string with a single character "d" for deposit or "w" for withdrawal), and the amount of the transaction (a double value for the amount to be deposited or withdrawn). Keep the order of the parameters for the constructor in: account number, transaction type, and then transaction amount.

In your class, these values are assigned when the `Transaction` is created and can be accessed but not modified. To access the values, write three methods: getAccountNumber, getTransactionType, and getTransactionAmount to return their corresponding values. Use these method names for testing purposes.

Write the complete class declaration for class `Transaction`. Include all necessary instance variables and implementations of its constructor and methods.

Part B:
An ATMTransaction class (a subclass of Transaction) is also needed to represent on ATM transaction. In addition to the information in a Transaction, an ATMTransaction contains a string that represents the location of the ATM that was used. Keep the order of the parameters for the constructor in: account number, transaction type, transaction amount, and then location.

The value of that location is assigned when the ATMTransaction is created and can be accessed but not modified. To access the location, write the method getATMLocation (be sure to use this method name for testing purposes). Write a complete class declaration for the ATMTransaction class. Include all necessary fields and implementations of its constructor and methods.

Part C:
A class `Bank` will be used to store information about all of the accounts in one bank and to perform transactions on those accounts. An incomplete declaration of the `Bank` class is given below.

```
public class Bank{
     private BankAccount [ ] accounts;
     /**
     * precondition: acctNum is the number of an account in the
     * accounts array.
     * postcondition: returns the index in the accounts array of the
     * given account number. */
     private int getIndex (int accountNum)
      {
          for(int i = 0; i < accounts.length; i++)
               {
                    if(accounts[i].getAccountNum()==accountNum)
                    {
                              return i;
                    }
               }
               return -1;
     }

     /**
     * precondition: trans is a transaction for an account in the
     * accounts array.
     * postcondition: the account for trans has been modified to
     * reflect the change specified
     * by the transaction. */
     public void doOneTransaction ( Transaction trans )
```

```
      {
            //code to be implemented in part C
      }
}
```
Write the `doOneTransaction` method of the `Bank` class, which has one `Transaction` parameter. Method `doOneTransaction` should find the `BankAccount` with the account number in the given transaction, and it should deposit or withdraw the amount in the given `Transaction` from that account as appropriate.