/\*\*

\* Models a battery with a specific capacity, drain and charge

\* **@author** Hunter Damron

\* Honor code: I created this masterpiece on my own

\* Purpose: Emulates a battery which can be drained and recharged to its initial state

\*/

**public** **class** **Battery** {

/\*\*

\* instance variables

\*/

**private** **double** capacity;

**private** **double** initialCapacity;

/\*\*

\* Constructor(s)

\* **@capacity**

\*/

**public** **Battery**(**double** capacity){

**this**.capacity = capacity;

initialCapacity = capacity;

}

/\*\*

\* drains the battery by the given amount

\* **@param** amount

\*/

**public** **void** **drain**(**double** amount){

capacity = capacity - amount;

}

/\*\*

\* charges the battery to its original capacity

\*/

**public** **void** **charge**(){

capacity = initialCapacity;

}

/\*\*

\* returns the remaining capacity of the battery

\* **@return** remaining capacity

\*/

**public** **double** **getRemainingCapacity**(){

**return** capacity;

}

}

/\*

\* BatteryTester.java

\* Hunter Damron

\* Honor Code: I copied this tester from Mrs. Bunn's board with integrity

\* Purpose: Tests Battery class by initializing with a charge, draining, and recharging

\*/

**public** **class** **BatteryTester** {

**public** **static** **void** **main**(**String**[] args) {

**Battery** **AA** = **new** Battery(2400);

**double** **drainAmount** = 50;

AA.drain(drainAmount);

**System**.***out***.println ("Remianing capacity: " + AA.getRemainingCapacity());

AA.charge();

**System**.***out***.println("This battery capacity is " + AA.getRemainingCapacity());

}

}