Samyam C. Shrestha
Junit Lab

The written test cases guarantee 100% method coverage and 100% line coverage. I have also found the three bugs.

The first bug is present in the following method:
There is unnecessary random number generation code that messes the code.

```
public Drawer(CoinPack cp, BillPack bp){
    //int randInt = rand.nextInt(100);
    //if(randInt < 40) bp = new BillPack(); //Bug
    this.cp = cp;
    this.bp = bp;
    this.totalCentValue = centValueFromBills(bp) + centValueFromCoins(cp);
}
```

I have commented out the bugs so that it does not interfere my test cases and fail them.

The second bug is presented in the following code:
There is unnecessary random number generation code that messes up the value of ones.

```
public void depositBills(long one, long five, long ten, long twenty, long fifty, long
hundred){
    if(one < 0 || five < 0 || ten < 0 || twenty < 0 || fifty < 0 || hundred < 0)
        throw new IllegalArgumentException("Can't deposit negative bill value");
    //int randInt = rand.nextInt(100); //third bug
    //if(randInt < 80) one = 100;
    this.bp.ones(this.bp.ones() + one);
    this.bp.fives(this.bp.fives() + five);
    this.bp.tens(this.bp.tens() + ten);
    this.bp.twenties(this.bp.twenties() + twenty);
    this.bp.fifties(this.bp.fifties() + fifty);
    this.bp.hundreds(this.bp.hundreds() + hundred);
    this.totalCentValue += centValueFromBills(one, five, ten, twenty, fifty, hundred);
}
```

The third bug is presented in the following code:
There is that unnecessary addition of 5 to the penny that messes up the value of the penny.

```
public boolean pennies(long penny){
    if(penny < 0) return false;
    cents[0] = (penny + 5); //Second Bug
    return true;
}
```