```java
/*
 * File: Bank.java
 * Author: David Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample
 * Vers: 1.0.0 09/04/2018 dgg - initial coding
 */

/**
 * Model a simple bank
 * @author David Green DGreen@uab.edu
 */
public class Bank {


    private static final int MAX_ACCOUNTS = 100;
    private String          name;
    private BankAccount[]   accounts;
    private int             numAccounts;
    private String          newLine;

    /**
     * Make a named bank using supplied name if available
     * @param name desired name for bank, if null one will be created
     */
    public Bank(String name) {
        if (name == null) {
            this.name = "Unnamed bank";
        } else {
            this.name   = name;
        }
        accounts    = new BankAccount[MAX_ACCOUNTS];
        numAccounts = 0;
    }

    // queries

    /**
     * get the name of the bank
     * @return name of bank as a String
     */
    public String getName() {
        return name;
    }

    /**
     * get the number of accounts in the bank
     * @return the number of accounts as an integer
     */
    public int getNumAccounts() {
        return numAccounts;
    }

    /**
     * Make statements for all accounts in bank
     * @return String representation of a (simple) report
     */
    public String getStatements() {

        String report          = "";
        String reportSeparator = System.getProperty("line.separator");

        for (int i = 0; i < numAccounts; i++ ) {
            report += getStatement(i) + reportSeparator;
        }
        return report;
    }

    /**
     * Make a statement for the specified account
     * @param accountNumber of the account to supply statement for
     * @return string representing account statement
     */
    public String getStatement(int accountNumber) {
        String statement;

        if (accountNumber < 0 || accountNumber >= numAccounts) {
            statement = "";
        } else {
            statement = accounts[accountNumber].toString();
        }
        return statement;
    }

    // commands

    /**
     * addAccount to bank
     *
     * TBD:  Handle error (presently silently ignores request)
     * TBD:  Avoid duplicate names  (should be done on making account)
     *
     * @param account to be added
     */
    public void addAccount(BankAccount account) {
        if ( numAccounts < MAX_ACCOUNTS ) {
            accounts[numAccounts++] = account;
        } else {
            // handle error
        }
    }

    /**
     * Pay interest on appropriate accounts
     */
    public void payInterest() {
        // TBD
    }

}
```

```java
/*
 * File: BankAccount.java
 * Author: David Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample
 * Vers: 1.1.0 01/24/2019 dgg - clean up
 * Vers: 1.0.0 09/04/2018 dgg - initial coding
*/

/**
 * Generic BankAccount
 * @author David Green DGreen@uab.edu
 */
public class BankAccount {

    // instance variables

    /**
     * Balance of account in cents
     */
    protected int balance;

    /**
     * Name of Account (as text)
     */
    protected String name;

    /**
     * Constructor for objects of class BankAccount
     */
    private BankAccount()
    {
        // Don't Allow
    }

    /**
     * Constructor with name, initial balance
     *
     * @param  aname     name of account
     * @param  abalance  initial amount in cents
     */
    public BankAccount(String aname, int abalance)
    {
        name = aname;
        balance = abalance;
    }

    /**
     * getBalance
     *
     * @return present balance in cents
     */
    public int getBalance()
    {
        return balance;
    }

    /**
     * deposit money into account
     *
     * @param cents   amount to add to present balance
     */
    public void deposit( int cents )
    {
        balance += cents;
    }

    /**
     * withdraw   remove money from account
     *
     * @param cents   amount to remove from account (if possible)
     * @return boolean  true if withdrawal is successful
     */
    public boolean withdraw( int cents )
    {
        if ( balance >= cents )
        {
            balance -= cents;
            return true;
        }
        else
            return false;
    }

    /**
     * toString return information about account
     *
     * @return string with type, name of account and balance
     */
    @Override
    public String toString()
    {
        return "Banking: " + name + ", " + balance ;
    }

    /**
     * getClassAuthor return name of author
     *
     * @return string containing name of author
     */
    public static String getClassAuthor()
    {
        return "David G. Green";
    }
}
```

```java
/*
 * File: BankingDemo.java
 * Author: David Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample
 * Vers: 1.1.0 01/24/2019 dgg - clean up
 * Vers: 1.0.0 09/04/2018 dgg - initial coding

/**
 * Explore banking Objects
 * @author David Green DGreen@uab.edu
 */

/**
 * Demo program to explore Banking Model
 * @author dgreen
 */

public class BankingDemo {

    /**
     * Program starts here
     * @param args unused
     */
    public static void main(String[] args) {

        Bank            bank  = new Bank("Birmingham");
        CheckingAccount ca    = new CheckingAccount("Joe Checking", 10000);
        SavingsAccount  sa    = new SavingsAccount( "Jill Savings", 20000);

        bank.addAccount(ca);
        bank.addAccount(sa);

        System.out.println( bank.getStatements() );
    }

}
```

```java
/*
 * File: CheckingAccount.java
 * Author: David Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample
 * Vers: 1.0.0 09/04/2018 dgg - initial coding
 */

/**
 * Simple checking account
 * @author David Green DGreen@uab.edu
 */
public class CheckingAccount extends BankAccount {
    /**
     * constructor to build checking account
     *
     * @param   name   name of account
     * @param   cents  initial balance
     */
    public CheckingAccount( String name, int cents )
    {
        super( name, cents );
    }

    /**
     * toString - convert information to string representation
     *
     * @return String information about account
     */
    @Override
    public String toString()
    {
        return "Checking: " + name + ", " + balance;
    }

    /**
     * clearCheck process a check with overdraft penalty
     *
     * @param cents - amount of check
     * @return boolean - true if check clears
     */
    public boolean clearCheck( int cents )
    {
        if ( withdraw( cents ) )
            return true;
        // overdraw penalty
        balance -= 1500;
        return false;
    }
}
```

```java
/*
 * File: SavingsAccount.java
 * Author: David Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample
 * Vers: 1.0.0 09/04/2018 dgg - initial coding
 */

/**
 * Model a Savings Account
 * @author David Green DGreen@uab.edu
 */
public class SavingsAccount extends BankAccount {

    /**
     * Create a savings account with a given name and balance
     * @param name text name for account
     * @param cents opening balance
     */
    public SavingsAccount( String name, int cents )
    {
        super( name, cents );
    }

    @Override
    public String toString()
    {
        return "Savings: " + name + ", " + balance;
    }

    /**
     * Change savings account balance by paying interest at furnished rate for period
     * of time since last call to this method.
     * Rounding goes to the bank (not paid) to account
     * @param rate interest rate for period of time since last call in percent
     */
    public void payInterest( float rate )
    {
        int interest = (int)( (float) balance * rate ) / 100;
        balance += interest;
    }
}
```

```java
/*
 * File: BankAccountNGTest.java
 * Author: David G. Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample - EE333 Spring 2019
 * Vers: 1.0.0 01/24/2019 dgg - initial coding
 */

import static org.testng.Assert.*;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

/**
 *
 * @author David G. Green DGreen@uab.edu
 */
public class BankAccountNGTest {

    private BankAccount ba;

    public BankAccountNGTest() {
    }

    @BeforeMethod
    public void setUpMethod() throws Exception {

        ba = new BankAccount("Joe", 10000);
    }

    @AfterMethod
    public void tearDownMethod() throws Exception {
    }

    /**
     * Test of getBalance method, of class BankAccount.
     */
    @Test
    public void testGetBalance() {
        assertEquals(ba.getBalance(), 10000);
    }


}
```

```java
/*
 * File: BankNGTest.java
 * Author: David G. Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample - EE333 Spring 2019
 * Vers: 1.0.0 01/24/2019 dgg - initial coding
 *
 * Credits:  (if any for sections of code)
 */

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

import static org.testng.Assert.*;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

/**
 *
 * @author David G. Green DGreen@uab.edu
 */
public class BankNGTest {

    public BankNGTest() {
    }

    @BeforeMethod
    public void setUpMethod() throws Exception {
    }

    @AfterMethod
    public void tearDownMethod() throws Exception {
    }

    /**
     * Test of getName method, of class Bank.
     */
    @Test
    public void testGetName() {
        System.out.println("getName");
        Bank instance = null;
        String expResult = "";
        String result = instance.getName();
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

    /**
     * Test of getNumAccounts method, of class Bank.
     */
    @Test
    public void testGetNumAccounts() {
        System.out.println("getNumAccounts");
        Bank instance = null;
        int expResult = 0;
        int result = instance.getNumAccounts();
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

    /**
     * Test of getStatements method, of class Bank.
     */
    @Test
    public void testGetStatements() {
        System.out.println("getStatements");
        Bank instance = null;
        String expResult = "";
        String result = instance.getStatements();
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

    /**
     * Test of getStatement method, of class Bank.
     */
    @Test
    public void testGetStatement() {
        System.out.println("getStatement");
        int account = 0;
        Bank instance = null;
        String expResult = "";
        String result = instance.getStatement(account);
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

    /**
     * Test of addAccount method, of class Bank.
     */
    @Test
    public void testAddAccount() {
        System.out.println("addAccount");
        BankAccount account = null;
        Bank instance = null;
        instance.addAccount(account);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

    /**
     * Test of payInterest method, of class Bank.
     */
    @Test
    public void testPayInterest() {
        System.out.println("payInterest");
        Bank instance = null;
        instance.payInterest();
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

}
```

```java
/*
 * File: CheckingAccountNGTest.java
 * Author: David G. Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample - EE333 Spring 2019
 * Vers: 1.0.0 01/24/2019 dgg - initial coding
 *
 * Credits:  (if any for sections of code)
 */

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

import static org.testng.Assert.*;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

/**
 *
 * @author David G. Green DGreen@uab.edu
 */
public class CheckingAccountNGTest {

    public CheckingAccountNGTest() {
    }

    @BeforeMethod
    public void setUpMethod() throws Exception {
    }

    @AfterMethod
    public void tearDownMethod() throws Exception {
    }

    /**
     * Test of toString method, of class CheckingAccount.
     */
    @Test
    public void testToString() {
        System.out.println("toString");
        CheckingAccount instance = null;
        String expResult = "";
        String result = instance.toString();
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

    /**
     * Test of clearCheck method, of class CheckingAccount.
     */
    @Test
    public void testClearCheck() {
        System.out.println("clearCheck");
        int cents = 0;
        CheckingAccount instance = null;
        boolean expResult = false;
        boolean result = instance.clearCheck(cents);
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }
```

```java
}
```

```java
/*
 * File: SavingsAccountNGTest.java
 * Author: David G. Green DGreen@uab.edu
 * Assignment:  BankInheritanceExample - EE333 Spring 2019
 * Vers: 1.0.0 01/24/2019 dgg - initial coding
 *
 * Credits:  (if any for sections of code)
 */

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

import static org.testng.Assert.*;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

/**
 *
 * @author David G. Green DGreen@uab.edu
 */
public class SavingsAccountNGTest {

    public SavingsAccountNGTest() {
    }

    @BeforeMethod
    public void setUpMethod() throws Exception {
    }

    @AfterMethod
    public void tearDownMethod() throws Exception {
    }

    /**
     * Test of toString method, of class SavingsAccount.
     */
    @Test
    public void testToString() {
        System.out.println("toString");
        SavingsAccount instance = null;
        String expResult = "";
        String result = instance.toString();
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

    /**
     * Test of payInterest method, of class SavingsAccount.
     */
    @Test
    public void testPayInterest() {
        System.out.println("payInterest");
        float rate = 0.0F;
        SavingsAccount instance = null;
        instance.payInterest(rate);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }

}
```