```
##########################################

Student Name = Jitong Ding

##########################################



CSE017 Grading sheet for Jitong Ding

Homework Assignment Bank

Total points                          maximum: 100

Completeness: All class/methods included (40)    [   40  ]

Compilation: Program compiles (20)                [   20  ]

Execution: Program executes properly (30)         [   30  ]

Style: Program obeys style rules (10)             [    8  ]

Subtotal                                              98

Late Penalty                                      [       ]

Total Points                                          98


##########################################
##########################################
```

```
/*
CSE 17
Jitong Ding
jid221
Program #2 DEADLINE: October 16, 2018
Program Description: Simple Bank
*/

import java.io.File;
import java.util.Scanner;

public class Bank{

  /** Private data field */
  private String name;
  private BankAccount[] accounts;
  private int totalAccounts;
  public int MAX_ACCOUNTS = 20;

  /** Construct a new Bank with name, totalAccounts, accounts */
  public Bank(String aName){
    name = aName;
    totalAccounts =0;
    accounts = new BankAccount[MAX_ACCOUNTS];
  }

  /** A method to return name*/
  public String getName() throws Exception{
    return name;
  }

  /** A method to add the given account to accounts and update totalAccounts acc
ordingly*/
  public void addAccount(BankAccount newAcct) throws Exception{
    accounts[totalAccounts] = newAcct;
    totalAccounts++;
  }

  /** A menthod for printing the bank name and information about each account*/
  public void printBankSummary() throws Exception{
    System.out.println("Bank Name: "+name);
    for(int i =0;i<totalAccounts;++i){
      accounts[i].printAccountInfo();
  }
  }

  /** A method to accrue one months interest to each savings account, using its
interestRate.*/
  public void accrueInterestToSavingsAccounts() throws Exception{
    for(int i =0;i<totalAccounts;++i){
      if (accounts[i] instanceof SavingsAccount){
        ((SavingsAccount)accounts[i]).accrueInterest();
    }
  }
  }

  /** A method to subtract the monthlyFee from the balance of each CheckingAccou
nt. */
  public void applyFeesToCheckingAccounts() throws Exception{
    for(int i =0;i<totalAccounts;++i){
      if (accounts[i] instanceof CheckingAccount){
        ((CheckingAccount)accounts[i]).applyFee();
    }
  }
  }

  /** A method to read the messages from the file by using scanner and adding th
e checking or savings account to a BankAccount Array.*/
  public void loadAccountsFromFile(File acctFile) throws Exception{
    Scanner scan = new Scanner(acctFile);
    scan.useDelimiter("\\t|[\\n\\r\\f]+");
    BankAccount[] account = new BankAccount[20];
    int i = 0;
```



```
    /** A while loop to add the information read from the file into the decleare
d variables */
    while(scan.hasNextLine()){
      String line = scan.next();
      int accNum = scan.nextInt();
      String accOwner = scan.next();
      double accBanlance = scan.nextDouble();
      double num = scan.nextDouble();
      /** An if-else statement to tell whether is checking or savings accounts*/
      if(line.equals("C")){
        account[i] = new CheckingAccount(accNum,accOwner,accBanlance, num);
      }
      else{
        account[i] = new SavingsAccount(accNum,accOwner,accBanlance, num);
      }
      i++;
    }
    for(int j=0; j<i;++j){
      addAccount(account[j]);
    }
    scan.close();
  }


   /** The main method */
  public static void main(String[] args) throws Exception{
     if(args.length != 1){
      System.out.println("Usage: java TextDecoder filename");
      System.exit(0);
    }
     File messageFile = new File(args[0]);
     Bank aBank = new Bank("Java S&L");
     aBank.loadAccountsFromFile(messageFile);
     aBank.printBankSummary();
     aBank.accrueInterestToSavingsAccounts();
     aBank.applyFeesToCheckingAccounts();
     System.out.println();
     aBank.printBankSummary();
}
}
```

```
/*
CSE 17
Jitong Ding
jid221
Program #2 DEADLINE: October 16, 2018
Program Description: Simple Bank
*/

public class BankAccount{

  /**  Protected data field */
  protected int accountNum;
  protected String customerName;
  protected double balance;

  /** Construct a new BankAccount with accountNum, customerName, balance */
  public BankAccount(int theAccountNum, String aCostomerName, double aBalance){
    accountNum = theAccountNum;
    customerName = aCostomerName;
    balance = aBalance;
  }

  /** Construct a new BankAccount with accountNum, customerName, balance equals
to 0*/
  public BankAccount(int anAccountNum, String theCostomerName){
    accountNum = anAccountNum;
    customerName = theCostomerName;
    balance = 0;
  }

  /** A method to return accountNum*/
  public int getAccountNum(){
    return accountNum;
  }

   /** A method to return customerName*/
  public String getCustomerName(){
    return customerName;
  }

   /** A method to return balance*/
  public double getBalance(){
    return balance;
  }

   /** A method to add depositAmt to the accounts balance*/
  public void makeDeposit(double depositAmt){
    balance += depositAmt;
  }

  /** A method to print information about the Bank account*/
  public void printAccountInfo(){
    System.out.printf("%-27s%8.2f\n", accountNum+customerName, balance);
}
}

```

Did not reference other constructor using this() (-1)

```
/*
CSE 17
Jitong Ding
jid221
Program #2 DEADLINE: October 16, 2018
Program Description: Simple Bank
*/

/** A subclass of the BankAccount class*/
public class CheckingAccount extends BankAccount{

  /** A private data field named monthlyFee for the monthlyFee of the CheckingAccount. */
  private double monthlyFee;

   /** Construct a new CheckingAccount with accountNum, customerName, balance and monthlyFee*/
  public CheckingAccount(int theAccountNum, String theCustomerName, double aBalance, double monthlyFee){
     super(theAccountNum,theCustomerName,aBalance);
     this.monthlyFee = monthlyFee;
   }

    /** A method to return monthlyFee*/
  public double getMonthlyFee(){
    return monthlyFee;
   }

  /** A method return the new monthlyFee.*/
  public void setMonthlyFee(double monthlyFee){
    this.monthlyFee = monthlyFee;
   }

  /** A method to subtracts monthlyFee from the balance.*/
  public void applyFee(){
    balance -= monthlyFee;
   }

    /** A method to print information about the checking account*/
  public void printAccountInfo(){
    System.out.printf("%-27s%8.2f  %-12s %1.2f\n", (accountNum)+" "+(customerName), balance,"Monthly fee:"+" $",monthlyFee);
}
}
```

```
/*
CSE 17
Jitong Ding
jid221
Program #2 DEADLINE: October 16, 2018
Program Description: Simple Bank
*/

/** A subclass of the BankAccount class*/
public class SavingsAccount extends BankAccount{

  /** A private data field named interestRate for the interestRate of the SavingsAccount. */
  private double interestRate;

   /** Construct a new SavingsAccount with accountNum, customerName, balance and interestRate*/
  public SavingsAccount(int theAccountNum, String theCustomerName, double aBalance, double interestRate){
    super(theAccountNum,theCustomerName,aBalance);
    this.interestRate = interestRate;
  }

  /** Construct a new SavingsAccount with accountNum, customerName, balance which equals 0 and interestRate*/
  public SavingsAccount(int anAccountNum, String theCostomerName, double interestRate){
    super(anAccountNum,theCostomerName);
    this.interestRate = interestRate;
  }

   /** A method to return interestRate*/
  public double getInterestRate(){
    return interestRate;
  }

  /** A method to add interestRate * balance to the balance.*/
  public void accrueInterest(){
    balance += (balance * interestRate);
  }

   /** A method to print information about the checking account*/
  public void printAccountInfo(){
    System.out.printf("%-27s%8.2f  %-14s%3.1f%%\n", (accountNum)+" "+(customerName), balance,"Interest Rate: ",interestRate*100);
}
}
```

Did not reference other constructor using this() (-1)

```
##########################################
##########################################


############ Compiled Result ###############


Source Code Compilation:



##########################################


##########################################
############ Execution Result ##############
##########################################


##########################################
Test1(acctinfo.txt) output - testOutput1.txt


Bank Name: Java S&L
42001 Gordon Gecko         85234.12   Interest Rate:  0.1%
44001 Flower Power            12.83   Monthly fee: $ 9.95
44002 Joe Schmo              392.52   Monthly fee: $ 4.50

Bank Name: Java S&L
42001 Gordon Gecko         85319.35   Interest Rate:  0.1%
44001 Flower Power             2.88   Monthly fee: $ 9.95
44002 Joe Schmo              388.02   Monthly fee: $ 4.50


##########################################
Test2(emptyFile.txt) output - testOutput2.txt


Bank Name: Java S&L

Bank Name: Java S&L


##########################################
Test3(oneAccountLongNames.txt) output - testOutput3.txt


Bank Name: Java S&L
9701 John Trevor Smith     55661.44   Monthly fee: $ 1.75

Bank Name: Java S&L
9701 John Trevor Smith     55659.69   Monthly fee: $ 1.75


##########################################
Test4(maxRecordsFile.txt) output - testOutput4.txt


Bank Name: Java S&L
55500 First1 Last1         11110.00   Interest Rate:  1.1%
55501 First2 Last2         22210.00   Monthly fee: $ 1.21
55502 First3 Last3         33310.00   Monthly fee: $ 1.32
55503 First4 Last4         44410.00   Monthly fee: $ 1.43
55504 First5 Last5         55510.00   Interest Rate:  1.5%
55505 First6 Last6         66610.00   Monthly fee: $ 1.65
55506 First7 Last7         77710.00   Interest Rate:  1.7%
55507 First8 Last8         88810.00   Interest Rate:  1.8%
```



```
55508 First9 Last9         99910.00   Monthly fee: $ 1.98
55509 FirstA LastA         12310.00   Interest Rate:  0.1%
55510 FirstB LastB         45610.00   Monthly fee: $ 2.12
55511 FirstC LastC         78910.00   Monthly fee: $ 3.21
55512 FirstD LastD         12210.00   Monthly fee: $ 4.32
55513 FirstE LastE         13310.00   Monthly fee: $ 5.43
55514 FirstF LastF         14410.00   Interest Rate:  0.6%
55515 FirstG LastG         15510.00   Interest Rate:  0.7%
55516 FirstH LastH         16610.00   Interest Rate:  0.8%
55517 FirstI LastI         17710.00   Monthly fee: $ 3.96
55518 FirstJ LastJ         18810.00   Interest Rate:  2.1%
55519 FirstK LastK         19910.00   Interest Rate:  2.2%

Bank Name: Java S&L
55500 First1 Last1         11232.21   Interest Rate:  1.1%
55501 First2 Last2         22208.79   Monthly fee: $ 1.21
55502 First3 Last3         33308.68   Monthly fee: $ 1.32
55503 First4 Last4         44408.57   Monthly fee: $ 1.43
55504 First5 Last5         56342.65   Interest Rate:  1.5%
55505 First6 Last6         66608.35   Monthly fee: $ 1.65
55506 First7 Last7         79031.07   Interest Rate:  1.7%
55507 First8 Last8         90408.58   Interest Rate:  1.8%
55508 First9 Last9         99908.02   Monthly fee: $ 1.98
55509 FirstA LastA         12322.31   Interest Rate:  0.1%
55510 FirstB LastB         45607.88   Monthly fee: $ 2.12
55511 FirstC LastC         78906.79   Monthly fee: $ 3.21
55512 FirstD LastD         12205.68   Monthly fee: $ 4.32
55513 FirstE LastE         13304.57   Monthly fee: $ 5.43
55514 FirstF LastF         14496.46   Interest Rate:  0.6%
55515 FirstG LastG         15618.57   Interest Rate:  0.7%
55516 FirstH LastH         16742.88   Interest Rate:  0.8%
55517 FirstI LastI         17706.04   Monthly fee: $ 3.96
55518 FirstJ LastJ         19205.01   Interest Rate:  2.1%
55519 FirstK LastK         20348.02   Interest Rate:  2.2%
```