**Assignment #6:**

This assignment consists of three parts: A, B and C.

**Part A:**

**Class Mreview (in "Mreview.java")**

**package** Assignment6;

**import** java.util.\*;

**public** **class** Mreview **implements** Comparable<Mreview>

{

**private** String title;

**private** ArrayList<Integer> ratings;

**public** Mreview()

{

**this**.title ="";

ratings = **new** ArrayList<Integer>();

}

**public** Mreview(java.lang.String ttl)

{

**this**.title = ttl;

ratings = **new** ArrayList<Integer>();

}

**public** Mreview(java.lang.String ttl, **int** firstRating)

{

**this**.title = ttl;

ratings = **new** ArrayList<Integer>(1);

ratings.add(firstRating);

}

**public** java.lang.String getTitle()

{

**return** title;

}

**public** **void** addRating(**int** r)

{

ratings.add(r);

}

**public** **double** aveRating()

{

**double** sum = 0;

**double** ave = 0;

**for**(**int** i=0; i< ratings.size(); i++)

{

sum += ratings.get(i);

}

ave = sum/ratings.size();

**return** ave;

}

**public** **int** numRatings()

{

**return** ratings.size();

}

**public** **int** compareTo(Mreview obj)

{

**return** **this**.title.compareTo(obj.title);

}

**public** **boolean** equals(java.lang.Object obj)

{

obj = **new** Mreview();//Casting to make the parameter a Mreview object.

**if**(**this**.title.equals(obj)){

**return** **true**;

}

**else**

**return** **false**;

}

**public** java.lang.String toString()

{

String display = "";

display = **this**.getTitle()+",average "+String.*format*("$%.2f", **this**.aveRating())+" out of "+**this**.numRatings()+" ratings.";

**return** display;

}

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

{

Mreview m1 = **new** Mreview("Avatar");

m1.ratings.add(4);

m1.ratings.add(4);

m1.ratings.add(5);

System.***out***.println(m1);

Mreview m2 = **new** Mreview("Titanic");

m2.ratings.add(4);

m2.ratings.add(5);

m2.ratings.add(5);

System.***out***.println(m2);

Mreview m3 = **new** Mreview("Kill Bill");

m3.ratings.add(5);

m3.ratings.add(3);

m3.ratings.add(2);

System.***out***.println(m3);

}

}

|  |
| --- |
|  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Part B:**

Implement the following two classes which are in an inheritance relation:

**package** Assignment6;

**public** **class** Account {

**protected** String FirstName, LastName;

**protected** **double** CurBalance;

**public** Account(String fname, String lname, **double** cb)

{

FirstName = fname;

LastName = lname;

CurBalance = cb;

}

**public** String getAcctType()

{

**return** **this**.getClass().getName();

}

**public** **double** DebitTransaction(**double** debitAmount)

{

CurBalance -= debitAmount;

**return** CurBalance;

}

**public** **double** CreditTransaction(**double** creditAmount)

{

CurBalance += creditAmount;

**return** CurBalance;

}

**public** String toString()

{

**return** "Account name: " + FirstName + " " + LastName + ", Account Type: " + getAcctType() + ", Balance: " + String.*format*("$%.2f", CurBalance);

}

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

{

Account ac1 = **new** Account("MartinLuther", "King", 100);

System.***out***.println(ac1);

ac1.DebitTransaction(30.25);

System.***out***.println(ac1);

ac1.CreditTransaction(10.10);

System.***out***.println(ac1);

}

}

### 2. Class CheckingAccount (partially filled ["CheckingAccount.java"](http://condor.depaul.edu/ntomuro/courses/301/assign/hw1files/CheckingAccount.java))

This account type has to have a pre-set minimum balance. For every transaction resulting in (i.e., AFTER the transaction) the amount less than the minimum balance, a pre-set penalty fee is charged.  The balance can become negative.  Following are the methods that this class provides. Your job is to fill the ones marked with **(\*)**. :

1. **public CheckingAccount(String fname, String lname, double cb) (\*)**  
   -- The three parameters should be utilized to initialize the members inherited from the base class Account.    
   -- Call the super class Account's constructor to do so.
2. **public double DebitTransaction(double debitAmount) (\*)**  
   -- This method overrides the method with the same name inherited from the super class Account.  
   -- First call the super class' method (with the given parameter).  Then call ChargeFee() of this class to possibly charge a fee (if the resulting balance went down below the MinBalance).  
   -- Then returns the resulting CurBalance.
3. **public double CreditTransaction(double creditAmount) (\*)**  
   -- This method overrides the method with the same name inherited from the super class Account.  
   -- First call the super class' method (with the given parameter).  Then call ChargeFee() of this class to possibly charge a fee (if the resulting balance is still below the MinBalance).Adds the transaction amount to the account and possibly charges a transaction fee.  
   -- Then returns the resulting CurBalance.
4. **private void ChargeFee() (\*)**  
   -- This is a private method.  It is called internally by the debit and credit transaction methods.  
   -- This method subtracts Fee from CurBalance if the current CurBalance is below (<) MinBalance.

**package** Assignment6;

**public** **class** CheckingAccount **extends** Account{

**static** **private** **double** *MinBalance* = 100;

**static** **private** **double** *Fee* = 10;

**public** CheckingAccount(String fname, String lname, **double** cb){

**super**(fname, lname, cb);

}

@Override

**public** **double** DebitTransaction(**double** debitAmount){

CurBalance = **super**.DebitTransaction(debitAmount);

**if**(CurBalance < *MinBalance*){

ChargeFee();

}

**return** CurBalance;

}

@Override

**public** **double** CreditTransaction(**double** creditAmount){

CurBalance = **super**.CreditTransaction(creditAmount);

**if**(CurBalance < *MinBalance*){

ChargeFee();

}

**return** CurBalance;

}

**private** **void** ChargeFee(){

CurBalance -= *Fee*;

}

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

{

CheckingAccount ch1 = **new** CheckingAccount("Sandhiya", "Ramesh", 50); System.***out***.println(ch1);

ch1.DebitTransaction(0.25);

System.***out***.println(ch1);

ch1.CreditTransaction(7.00);

System.***out***.println(ch1);

ch1.CreditTransaction(1000000);

System.***out***.println(ch1);

}

}

**Part C:**

This assignment requires you to develop an object oriented software system in Java that will keep track of pets treated and boarded in an animal hospital. Detail class specifications (data members, methods and access modifiers) are described below. Please note the following requirements:

• You need to implement each class in a separate file.

• While implementing the design you may want to follow the order the classes are specified below and test each class individually.

**Class: Pet (File name: Pet.java)**

The class should have the following three private data members, Pet name (a String), owner name (a String), and color (a String), and one protected data members for sex (an integer, but it will only hold one of the following four public static final int values: MALE, FEMALE, SPAYED and NEUTERED. You should define these four static finals in your class).

Following are the public methods that this class should provide:

Pet (String name, String ownerName, String color); //Constructor

String getPetName();

String getOwnerName();

String getColor();

void setSex(int sexid);

String getSex(); // Should return the string equivalent of the gender, e.g the string “MALE” etc.

String toString(); // Should return the name, owner’s name, age, color, and gender (use getSex());

**A Sample (preferred) return value by toString is as follows:**

Spot owned by Mary

Color: Black and White

Sex: Male

**Interface: Boardable (File name: Boardable.java)**

This interface, should include the following public methods:

void setBoardStart(int month, int day, int year);

void setBoardEnd(int month, int day, int year);

boolean boarding(int month, int day, int year);

See the Cat and Dog classes for what these methods should do when implemented. Note, the month will be in the range 1-12, day in the range 1-31, and year will be a four digit number.

**Class: Cat (File name: Cat.java)**

This class should extend the Pet class and implement the Boardable interface. In addition to the data members and methods inherited from Pet, the Cat class should have a private hairLength data member, which is a string. Following are the public methods that this class should provide Cat (String name, String ownerName, String color, String hairLength);

// Do not forget to call super.

String getHairLength(); // returns the string hairLength

String toString()

/\* method that returns a String that identifies the pet as Cat and returns a complete description of the cat, including the values stored in the Pet parent class.\*/

**A Sample (preferred) return value by toString is as follows:**

CAT:

Tom owned by Bob

Color: black

Sex: spayed

Hair: short

In order to implement the Boardable interface define new data members to store the boarding start and end dates, implement the setBoardStart and setBoardEnd methods to store values for these data members. Also implement the boarding method to return true if the given data is between the start and end dates, otherwise it returns false. Note: You should also return true if the given date is equal to the start or end date.

**Class: Dog (File name: Dog.java)**

This class should extend the Pet class and implement the Boardable interface. In addition to the data members and methods inherited from Pet, the Dog class should have a private size data member, which is a string. Following are the public methods that this class should provide:

Dog (String name, String ownerName, String color, String size);

// Constructor must set the size. Do not forget to call super.

String getSize(); // returns the string size String toString();

/\* method that returns a String that identifies the pet as Dog and returns a complete description of the dog, including the values stored in the Pet parent class. \*/

**A Sample (preferred) return value by toString is as follows:**

DOG:

Spot owned by Susan

Color: white

Sex: spayed

Size: medium

In order to implement the Boardable interface define new data members to store the boarding start and end dates, implement the setBoardStart and setBoardEnd methods to store values for these data members. Also implement the boarding method to return true if the given data is between the start and end dates, otherwise it returns false. Note: You should also return true if the given date is equal to the start or end date.

**PET.JAVA**

**package** Assignment6;

**public** **class** Pet {

**private** String petName;

**private** String ownerName;

**private** String color;

**protected** **int** sex;

**public** **static** **final** **int** ***MALE*** = 1;

**public** **static** **final** **int** ***FEMALE*** = 2;

**public** **static** **final** **int** ***SPAYED*** = 3;

**public** **static** **final** **int** ***NEUTERED*** = 4;

**public** Pet(String petName, String ownerName, String color){

**this**.petName = petName;

**this**.ownerName = ownerName;

**this**.color = color;

}

**public** String getPetName(){

**return** petName;

}

**public** String getOwnerName(){

**return** ownerName;

}

**public** String getColor(){

**return** color;

}

**public** **void** setSex(**int** sexid){

**if**(sexid > 0 && sexid < 5){

sex = sexid;

}

**else**

System.***out***.println("Sexid can just be 1, 2, 3, and 4!");

}

**public** String getSex(){

String petsex ="";

**if** (sex == ***MALE***){

petsex = "MALE";

}**else** **if**(sex == ***FEMALE***){

petsex = "FEMALE";

}**else** **if**(sex == ***SPAYED*** ){

petsex = "SPAYED";

}**else** **if** (sex == ***NEUTERED***){

petsex ="NEUTERED";

}**else**{

petsex = "not given";

}

**return** petsex;

}

**public** String toString(){

String display ="";

display += getPetName()+", owned by "+ getOwnerName()+"\nColor: "+getColor()+"\nSex: "+getSex();

**return** display;

}}

**BOARABLE.JAVA**

**package** Assignment6;

**public** **interface** Boardable {

**public** **void** setBoardStart(**int** month, **int** day, **int** year);

**public** **void** setBoardEnd(**int** month, **int** day, **int** year);

**public** **boolean** boarding(**int** month, **int** day, **int** year);

}

**CAT.JAVA**

**package** Assignment6;

**import** java.util.Calendar;

**public** **class** Cat **extends** Pet **implements** Boardable {

**private** String hairLength;

**private** Calendar BoardStartDate = Calendar.*getInstance*();

**private** Calendar BoardEndDate = Calendar.*getInstance*();

**public** Cat(String petName, String ownerName, String color, String hairLength) {

**super**(petName, ownerName, color);

**this**.hairLength = hairLength;

}

**public** String gethairLength() {

**return** hairLength;

}

**public** **void** setBoardStart(**int** month, **int** day, **int** year) {

**this**.BoardStartDate.set(month, day, year);

}

**public** **void** setBoardEnd(**int** month, **int** day, **int** year) {

**this**.BoardEndDate.set(month, day, year);

}

**public** **boolean** boarding(**int** month, **int** day, **int** year) {

Calendar date = Calendar.*getInstance*();

date.set(month, day, year);

**if** (date.before(BoardStartDate) || date.after(BoardEndDate))

**return** **false**;

**else**

**return** **true**;

}

**public** String toString() {

String display = "";

display += "Cat:\n" + **super**.toString() + "\nHair: " + **this**.gethairLength();

**return** display;

}

}

**DOG.JAVA**

**package** Assignment6;

**import** java.util.\*;

**public** **class** Dog **extends** Pet **implements** Boardable {

**private** String size;

**private** Calendar BoardStartDate = Calendar.*getInstance*();

**private** Calendar BoardEndDate = Calendar.*getInstance*();

**public** Dog(String petName, String ownerName, String color, String size) {

**super**(petName, ownerName, color);

**this**.size = size;

}

**public** String getsize() {

**return** size;

}

**public** **void** setBoardStart(**int** month, **int** day, **int** year) {

**this**.BoardStartDate.set(month, day, year);

}

**public** **void** setBoardEnd(**int** month, **int** day, **int** year) {

**this**.BoardEndDate.set(month, day, year);

}

**public** **boolean** boarding(**int** month, **int** day, **int** year) {

Calendar date = Calendar.*getInstance*();

date.set(month, day, year);

**if** (date.before(BoardStartDate))

**return** **false**;

**if** (date.after(BoardEndDate))

**return** **false**;

**else**

**return** **true**;

}

**public** String toString() {

String display = "";

display += "Dog:\n" + **super**.toString() + "\nSize: " + **this**.getsize();

**return** display;

}

}

**TESTPET.JAVA**

**package** Assignment6;

**public** **class** TestPet {

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

Cat cat = **new** Cat("Tom", "Bob", "black","short");

cat.setSex(3);

System.***out***.println(cat);

cat.setBoardStart(1, 22, 2017);//

cat.setBoardEnd(1, 26, 2017);

System.***out***.println("Test date if it is between the start-board date and end-board date:");

System.***out***.println(cat.boarding(1, 21, 2017));

System.***out***.println(cat.boarding(1, 22, 2017));

System.***out***.println(cat.boarding(2, 27, 2017));

Dog dog = **new** Dog("Spot", "Susan", "white", "medium");

dog.setSex(3);

System.***out***.println(dog);

dog.setBoardStart(3, 22, 2017);

dog.setBoardEnd(4, 24, 2017);

System.***out***.println("Test date if it is between the start-board date and end-board date:");

System.***out***.println(dog.boarding(3, 21, 2017));

System.***out***.println(dog.boarding(4, 24, 2017));

System.***out***.println(dog.boarding(4, 29, 2017));

}

}