## CSCI 145 PA \_\_10\_\_ Submission

## Due Date:\_\_\_\_May 10 2023\_\_\_\_ Late (date and time):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Name(s):\_\_\_\_\_\_Ivan Leung\_\_\_\_\_\_ & \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Exercise 1 -- need to submit source code and I/O  
 -- check if completely done \_\_x\_\_ ; otherwise, discuss issues below

Pseudocode below if applicable:

Source code below:

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 01 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//DogTest.java

//

//A simple test class that creates a Dog and makes it speak.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **class** DogTest {

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

// Dog dog = new Dog("Spike");

Dog lab = **new** Labrador("Golden", "yellow");

Dog york = **new** Yorkshire("Happy");

// System.out.println(dog.getName() + " says " + dog.speak());

System.***out***.println(lab.getName() + " says " + lab.speak());

System.***out***.println(york.getName() + " says " + york.speak());

System.***out***.println(lab.getName() + "\'s average breed weight: " + lab.avgBreedWeight());

System.***out***.println(york.getName() + "\'s average breed weight: " + york.avgBreedWeight());

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 01 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Dog.java

//

//A class that holds a dog's name and can make it speak.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **abstract** **class** Dog {

**protected** String name;

// ------------------------------------------------------------

// Constructor -- store name

// ------------------------------------------------------------

**public** Dog(String name) {

**this**.name = name;

}

// ------------------------------------------------------------

// Returns the dog's name

// ------------------------------------------------------------

**public** String getName() {

**return** name;

}

// ------------------------------------------------------------

// Returns a string with the dog's comments

// ------------------------------------------------------------

**public** String speak() {

**return** "Woof";

}

**public** **abstract** **int** avgBreedWeight();

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 01 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Labrador.java

//

//A class derived from Dog that holds information about

//a labrador retriever. Overrides Dog speak method and includes

//information about avg weight for this breed.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **class** Labrador **extends** Dog {

**private** String color; // black, yellow, or chocolate?

**private** **int** breedWeight = 75;

**public** Labrador(String name, String color) {

**super**(name);

**this**.color = color;

}

// ------------------------------------------------------------

// Big bark -- overrides speak method in Dog

// ------------------------------------------------------------

**public** String speak() {

**return** "WOOF";

}

// ------------------------------------------------------------

// Returns weight

// ------------------------------------------------------------

**public** **int** avgBreedWeight() {

**return** breedWeight;

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 01 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Yorkshire.java

//

//A class derived from Dog that holds information about

//a Yorkshire terrier. Overrides Dog speak method.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **class** Yorkshire **extends** Dog {

**private** **int** breedWeight = 20;

**public** Yorkshire(String name) {

**super**(name);

}

// -------------------------------------------------------------

// Small bark -- overrides speak method in Dog

// -------------------------------------------------------------

**public** String speak() {

**return** "woof";

}

**public** **int** avgBreedWeight() {

**return** breedWeight;

}

}

Input/output below:

Golden says WOOF

Happy says woof

Golden's average breed weight: 75

Happy's average breed weight: 20

Exercise 2 -- need to submit source code and I/O  
 -- check if completely done \_\_x\_\_ ; otherwise, discuss issues below

Pseudocode below if applicable:

Source code below:

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 01 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//ListTest.java

//

//A simple test program that creates an IntList, puts some

//ints in it, and prints the list.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **class** ListTest {

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

IntList myList = **new** IntList(10);

IntList sortedList = **new** SortedIntList(8);

System.***out***.println("Unsorted Integer List...");

myList.add(100);

myList.add(50);

myList.add(200);

myList.add(25);

myList.add(250);

myList.add(75);

myList.add(125);

myList.add(25);

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

System.***out***.println("Sorted Integer List...");

sortedList.add(100);

sortedList.add(50);

sortedList.add(200);

sortedList.add(25);

sortedList.add(250);

sortedList.add(75);

sortedList.add(125);

sortedList.add(25);

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

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 01 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//IntList.java

//

//An (unsorted) integer list class with a method to add an

//integer to the list and a toString method that returns the contents

//of the list with indices.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **class** IntList {

**protected** **int**[] list;

**protected** **int** numElements = 0;

// -------------------------------------------------------------

// Constructor -- creates an integer list of a given size.

// -------------------------------------------------------------

**public** IntList(**int** size) {

list = **new** **int**[size];

}

// ------------------------------------------------------------

// Adds an integer to the list. If the list is full,

// prints a message and does nothing.

// ------------------------------------------------------------

**public** **void** add(**int** value) {

**if** (numElements == list.length)

System.***out***.println("Can't add, list is full");

**else** {

list[numElements] = value;

numElements++;

}

}

// -------------------------------------------------------------

// Returns a string containing the elements of the list with their

// indices.

// -------------------------------------------------------------

**public** String toString() {

String returnString = "";

**for** (**int** i = 0; i < numElements; i++)

returnString += i + ": " + list[i] + "\n";

**return** returnString;

}

}

**package** pa10;

/\* Java Class: CSCI 145

Author: Ivan Leung

Class: Mon/Wed

Date: May 01 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

**public** **class** SortedIntList **extends** IntList {

**public** SortedIntList(**int** size) {

**super**(size);

}

@Override

**public** **void** add(**int** value) {

**if** (numElements == list.length) {

System.***out***.println("Can't add, list is full");

**return**;

}

**if** (numElements == 0) {

list[0] = value;

++numElements;

**return**;

}

**else** {

**for** (**int** i = 0; i < numElements; ++i) {

**if** (value <= list[i]) {

**for** (**int** j = numElements; j > i; --j) {

list[j] = list[j - 1];

}

list[i] = value;

++numElements;

**return**;

}

}

}

list[numElements] = value;

++numElements;

}

}

Input/output below:

Unsorted Integer List...

0: 100

1: 50

2: 200

3: 25

4: 250

5: 75

6: 125

7: 25

Sorted Integer List...

0: 25

1: 25

2: 50

3: 75

4: 100

5: 125

6: 200

7: 250

Exercise 3 -- need to submit source code and I/O  
 -- check if completely done \_\_x\_\_ ; otherwise, discuss issues below

Pseudocode below if applicable:

Source code below:

**package** pa10;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//ComparePlayers

//

//Reads in two Player objects and tells whether they represent

//the same player.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**import** java.util.Scanner;

**public** **class** ComparePlayers {

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

Player player1 = **new** Player();

Player player2 = **new** Player();

Scanner scan = **new** Scanner(System.***in***);

// Prompt for and read in information for player 1

System.***out***.println("Enter information for player 1");

player1.readPlayer();

// Prompt for and read in information for player 2

System.***out***.println("\nEnter information for player 2");

player2.readPlayer();

scan.close();

// Compare player1 to player 2 and print a message saying

// whether they are equal

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

System.***out***.println(player1.equals(player2) ? "Same player" : "Different players");

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 02 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Player.java

//

//Defines a Player class that holds information about an athlete.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**import** java.util.Scanner;

**public** **class** Player {

**private** String name;

**private** String team;

**private** **int** jerseyNumber;

// -----------------------------------------------------------

// Prompts for and reads in the player's name, team, and

// jersey number.

// -----------------------------------------------------------

**public** **void** readPlayer() {

Scanner scan = **new** Scanner(System.***in***);

System.***out***.print("Name: ");

name = scan.nextLine();

System.***out***.print("Team: ");

team = scan.nextLine();

System.***out***.print("Jersey number: ");

jerseyNumber = scan.nextInt();

}

**public** String getTeam() {

**return** team;

}

**public** **int** getJerseyNumber() {

**return** jerseyNumber;

}

@Override

**public** **boolean** equals(Object other) {

Player player = (Player) other;

**return** (**this**.team.equalsIgnoreCase(player.team) && **this**.jerseyNumber == player.getJerseyNumber());

}

}

Input/output below:

Enter information for player 1

Name: Jimmy

Team: Dodgers

Jersey number: 88

Enter information for player 2

Name: David

Team: Dodgers

Jersey number: 88

Same player

*Add more exercises as needed*

Exercise 4 -- need to submit source code and I/O  
 -- check if completely done \_\_x\_\_ ; otherwise, discuss issues below

Pseudocode below if applicable:

Source code below:

**package** pa10;

/\* Java Class: CSCI 145

Author: Ivan Leung

Class: Mon/Wed

Date: May 02 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

**import** java.util.Scanner;

**public** **class** CourseTest {

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

Course course = **new** Course();

Course offeredCourse = **new** OfferedCourse();

Scanner scan = **new** Scanner(System.***in***);

System.***out***.print("Enter course number: ");

course.setCourseNumber(scan.nextLine());

System.***out***.print("Enter course name: ");

course.setCourseTitle(scan.nextLine());

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

System.***out***.print("Enter offered course number: ");

offeredCourse.setCourseNumber(scan.nextLine());

System.***out***.print("Enter offered course name: ");

offeredCourse.setCourseTitle(scan.nextLine());

System.***out***.print("Enter offered course instructor: ");

((OfferedCourse) offeredCourse).setInstructorName(scan.nextLine());

System.***out***.print("Enter offered course term: ");

((OfferedCourse) offeredCourse).setTerm(scan.nextLine());

System.***out***.print("Enter offered course date/time: ");

((OfferedCourse) offeredCourse).setClassTime(scan.nextLine());

scan.close();

System.***out***.println("Course Information:\n" + course);

System.***out***.println("\nOffered Course Information:\n" + offeredCourse);

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 02 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

**public** **class** Course {

// **TODO**: Declare private fields - courseNumber, courseTitle

**private** String courseNumber;

**private** String courseTitle;

// **TODO**: Define default constructor – default to “unknown”

Course() {

courseNumber = "Unknown";

courseTitle = "Unkown";

}

// **TODO**: Define mutator methods -

// setCourseNumber(), setCourseTitle()

**public** **void** setCourseNumber(String number) {

courseNumber = number;

}

**public** **void** setCourseTitle(String title) {

courseTitle = title;

}

// **TODO**: Define accessor methods -

// getCourseNumber(), getCourseTitle()

**public** String getCourseNumber() {

**return** courseNumber;

}

**public** String getCourseTitle() {

**return** courseTitle;

}

// **TODO**: Override toString()

@Override

**public** String toString() {

**return** ("\tCourse Number: " + courseNumber + "\n\tCourse Title: " + courseTitle);

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 02 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

**public** **class** OfferedCourse **extends** Course {

// **TODO**: Declare private fields - instructorName, term, classTime

**private** String instructorName;

**private** String term;

**private** String classTime;

// **TODO**: Define default constructor - default to “unknown”

OfferedCourse() {

**super**();

}

// **TODO**: Define mutator methods -

// setInstructorName(), setTerm(), setClassTime()

**public** **void** setInstructorName(String name) {

instructorName = name;

}

**public** **void** setTerm(String t) {

term = t;

}

**public** **void** setClassTime(String time) {

classTime = time;

}

// **TODO**: Define accessor methods -

// getInstructorName(), getTerm(), getClassTime()

**public** String getInstructorName() {

**return** instructorName;

}

**public** String getTerm () {

**return** term;

}

**public** String getClassTime() {

**return** classTime;

}

// **TODO**: Override toString()

@Override

**public** String toString() {

**return** ("\tOffered Course Number: " + getCourseNumber() + "\n\tOffered Course Title: " + getCourseTitle()

+ "\n\tOffered Course Instructor: " + instructorName + "\n\tOffered Course Term: " + term

+ "\n\tOffered Course Date/Time: " + classTime);

}

}

Input/output below:

Enter course number: CSCI 145

Enter course name: Java Programming

Enter offered course number: CSCI 145 (20400)

Enter offered course name: Java Programming

Enter offered course instructor: T. Vo

Enter offered course term: Spring 2023

Enter offered course date/time: MW - 1:15-4:15 pm

Course Information:

Course Number: CSCI 145

Course Title: Java Programming

Offered Course Information:

Offered Course Number: CSCI 145 (20400)

Offered Course Title: Java Programming

Offered Course Instructor: T. Vo

Offered Course Term: Spring 2023

Offered Course Date/Time: MW - 1:15-4:15 pm

Answer for Question 1

Answer for Question 2

One way to determine if we should use inheritance or aggregation is that I would ask myself one question, “is the new class more or less a class of X?” If it is, then we will utilize inheritance.

Extra Credit – provide if applicable

Pseudocode below if applicable:

Source code below:

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 02 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

**import** java.text.NumberFormat;

**public** **class** CoinTest {

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

**final** **double** fair = 0.50;

**final** **double** bias10 = 0.10;

**final** **double** bias75 = 0.75;

BiasedCoin fcoin = **new** BiasedCoin(fair);

BiasedCoin bcoin1 = **new** BiasedCoin(bias10);

BiasedCoin bcoin2 = **new** BiasedCoin(bias75);

**int** totalHeads1 = 0;

**int** totalHeads2 = 0;

**int** totalHeads3 = 0;

NumberFormat percent = NumberFormat.*getPercentInstance*();

**for** (**int** i = 0; i < 100; ++i) {

fcoin.flip();

bcoin1.flip();

bcoin2.flip();

**if** (fcoin.isHeads())

++totalHeads1;

**if** (bcoin1.isHeads())

++totalHeads2;

**if** (bcoin2.isHeads())

++totalHeads3;

}

System.***out***.println("Total Heads...");

System.***out***.println("Fair coin (" + percent.format(fair) + " heads): " + totalHeads1);

System.***out***.println("Coin with " + percent.format(bias10) + " heads: " + totalHeads2);

System.***out***.println("Coin with " + percent.format(bias75) + " heads: " + totalHeads3);

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 02 2023

Description:

`

I certify that the code below is my own work.

Exception(s): N/A

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Coin.java Author: Lewis/Loftus

//

//Represents a coin with two sides that can be flipped.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **class** Coin {

**protected** **final** **int** HEADS = 0;

**protected** **final** **int** TAILS = 1;

**protected** **int** face;

// -----------------------------------------------------------------

// Sets up the coin with no bias and flipping it initially.

// -----------------------------------------------------------------

**public** Coin()

{

flip();

}

// -----------------------------------------------------------------

// Flips the coin by randomly choosing a face value.

// -----------------------------------------------------------------

**public** **void** flip() {

face = (**int**) (Math.*random*() \* 2);

}

// -----------------------------------------------------------------

// Returns true if the current face of the coin is heads.

// -----------------------------------------------------------------

**public** **boolean** isHeads() {

**return** (face == HEADS);

}

// -----------------------------------------------------------------

// Returns the current face of the coin as a string.

// -----------------------------------------------------------------

**public** String toString()

{

String faceName;

**if** (face == HEADS)

faceName = "Heads";

**else**

faceName = "Tails";

**return** faceName;

}

}

**package** pa10;

/\* Java Class: CSCI 145

Modified by: Ivan Leung

Class: Mon/Wed

Date: May 02 2023

Description:

I certify that the code below is my own work.

Exception(s): N/A

\*/

**public** **class** BiasedCoin **extends** Coin {

**private** **double** bias;

BiasedCoin(**double** bias) {

**super**();

**this**.bias = bias;

}

@Override

**public** **void** flip() {

face = (Math.*random*() < bias ? HEADS : TAILS);

}

}

Input/output below:

Total Heads...

Fair coin (50% heads): 46

Coin with 10% heads: 13

Coin with 75% heads: 78