CS159 – Advanced Programming
Sample Exam 2

James Madison University
Spring 2024

This work complies with the JMU Honor Code.

Name:____

Signature:____ Section:____________________

Instructions. Answer all of the following questions. This is a “closed book” examination and you must work entirely on your own.

1 Given the following declarations:

```
double speedGoal;
double[] speeds;
int[] times;
String dogName;
String[] dog;
ArrayList<Integer> ranking;

// assume all of these variables are next initialized
```

Indicate each of the following is a primitive type (Pri), a reference type (Ref), or syntactically invalid (Inv). Assume the variables declared above have also been initialized.

1.1____________ `speedGoal` (as in the statement `System.out.print(speedGoal);`)

1.2____________`dogName` (as in the statement `System.out.print(dogName);`)

1.3_____________`dogName[0]` (as in the statement `System.out.print(dogName[0]);`)

1.4____________ `speeds` (as in the statement `System.out.print(speeds);`)

1.5____________ `speeds[0]` (as in the statement `System.out.print(speeds[0]);`)

1.6____________ `rankings[0]` (as in the statement `System.out.print(rankings[0]);`)

1.7____________ `times` (as in the statement `System.out.print(times);`)

1.8_____________`times[0]` (as in the statement `System.out.print(times[0]);`)

2. Given the attached Boat class, and the following declarations and initializations/instantiations in a method name main() in class named as BoatDriver, indicate whether each of the following statements (also in the method named main(), immediately after the declarations and initializations/instantiations) will:

N - not compile
X - Compile but generate an exception at runtime
R - compile and run without generating an exception

```
public class BoatDriver {

    public static void main(String[] args) {
        int number;
        boolean flag;
        String name;
        double val;
        Boat b = new Boat(99, "Destiny", 10);
        Boat[] boats = new Boat[2];

        // This is where the statements below would be
       // (one statement per question)
      }
}
```

2.1_________________number = Boat.capacity;

2.2_________________name = b.toString();

2.3_________________flag = b.equals("Destiny");

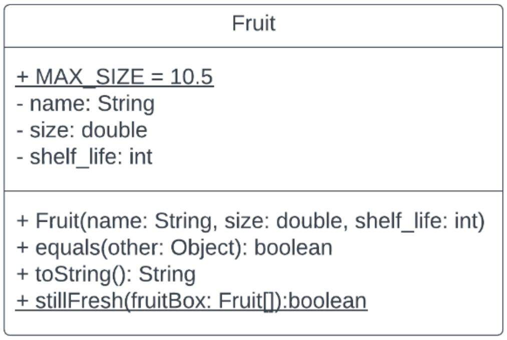
2.4_________________boats[0] = null;

2.5_________________boats[2] = b;

2.6_________________b.capacity += 10;

2.7_________________b = new Boat();

3. Consider the UML diagram below.



Fruit

+ MAX_SIZE = 10.5
- name: String
- size: double
- shelf_life: int

+ Fruit(name: String, size: double, shelf_life: int)
+ equals(other: Object): boolean
+ toString(): String
+ stillFresh(fruitBox: Fruit[]):boolean

a) (4 pts) Write the attribute declarations for Fruit as specified above.

b) (6 pts) Write the constructor for Fruit using the UML above and the following additional instructions. Each attribute should be set to the parameter of the same name, but if the parameter size is bigger than MAX_SIZE, set the attribute size to MAX_SIZE. Assume size is positive.

c) (5 pts) Write the equals method for Fruit using the UML above and the following additional instructions. A Fruit object is equal to another Fruit object if it is a Fruit object and both the name and the size attributes are equal.

d) (5 pts) Write the toString method for Fruit using the UML above and the following additional instructions. Return a String that is the first three characters of name in upper case, followed by a single space, followed by size formatted as a decimal number with two digits after the decimal point.

e) (5 pts) Write the stillFresh method using the UML above and the following additional instructions. Return true if and only if all of the Fruit objects in the fruitBox have a shelf_life greater than 0. Assume that fruitBox is not null.

f) (1 pt) Write a single statement to declare and instantiate a Fruit object variable, snackFruit, that has a name of orange, size of 3.5 and a shelf_life of 4.

g) (2 pts) Write a single statement to declare and initialize a boolean variable, sameFruit, based on if the above snackFruit is equal to another already declared and initialized Fruit object variable, newFruit.

h) (2 pts) Write a single statement to set a boolean variable, goodToEat, based on the results of calling stillFresh on an already declared and initialized Fruit[] variable, fruitBox.

4. (16 points) Given the attached implementations of the Size, EarRing, LipRing, NavelRing, and NoseRing classes, what will be printed by each of the following main classes? Note: None of the classes contain any syntax errors.

```
// The main class for question 2.1
public class EarRingDriver {

   public static void main(String[] args) {
      Size d;
      d = new Size();
      d.radius = 5;
      d.thickness = 8;

      EarRing earRing;
      earRing = new EarRing(d);

      d.radius = 11;
      d.thickness = 14;

      System.out.println(earRing.toString());
   }
}
```

```
// The main class for question 2.2
    public class LipRingDriver {

       public static void main(String[] args) {
          Size d;
          d = new Size();
          d.radius = 5;
          d.thickness = 8;

          LipRing lipRing;
          lipRing = new LipRing(d);

          d.radius = 11;
          d.thickness = 14;

          System.out.println(lipRing.toString());
       }
    }
```

```
// The main class for question 2.3
public class NavelRingDriver {

   public static void main(String[] args) {
      Size d;
      d = new Size();
      d.radius = 5;
      d.thickness = 8;

      NavelRing navelRing;
      navelRing = new NavelRing(d);

      d.radius = 11;
      d.thickness = 14;

      System.out.println(navelRing.toString());
   }
}
```

```
// The main class for question 2.4
public class NoseRingDriver {

   public static void main(String[] args) {
      Size d;
      d = new Size();
      d.radius = 5;
      d.thickness = 8;

      NoseRing noseRing;
      noseRing = new NoseRing(d);

      d.radius = 11;
      d.thickness = 14;

      System.out.println(noseRing.toString());
   }
}
```

5. (5 Points) Given the attached FractionTokenizer class, what would be printed by the following fragment?

```
FractionTokenizer ft;
ft = new FractionTokenizer("2/4,3/4");

try {
   while (ft.hasNext()) {
      double answer;
      answer = ft.next();
      System.out.println(answer);
   }
} catch (NoSuchElementException nse) {
   System.out.println("That does not compute!");
}
```

6. (5 Points) Given the attached FractionTokenizer class, what would be printed by the following fragment?

```
FractionTokenizer ft;
ft = new FractionTokenizer("2/4,3/4");

for (int i=0; i<5; i++) {
   try {
      double answer;
      answer = ft.next();
      System.out.println(answer);
   } catch (NoSuchElementException nse) {
      System.out.println("That does not compute!");
   }
}
```

5. (5 pts) Fill in the JUnit test method below with enough tests to completely cover the following method.

```
public class MethodToTest {
    public static int mystery(boolean x, boolean y) {
        if (x && y) {
            return 0;
        }
        return 1;
    }
}
```

```
@Test
void testMystery() {
```

```
}
```

6. Consider the declarations of Boat class attached. Write a class called BoatUtils with two static methods as specified below.
a) Write saveBoatList method that will write out to a filename the names of all the boats in boats. Each name should be on it’s own line in the file. If either filename or boats is null, return false. If the file does not exist, throw an IllegalArgumentException using a try-catch clause. Return true otherwise.

```
public static boolean saveBoatList(String filename,   ArrayList<Boat>
boats) throws IllegalArgumentException{
```

```
}
```

b) Write loadBoatList method so that it will read in boat names from filename and storing them in the ArrayList of Strings names. If either filename or names is null, return false. If the file does not exist, the method should throw a FileNotFoundException. Return true otherwise.

```
public static boolean loadBoatList(String filename,  ArrayList<String>
names) throws  FileNotFoundException{
```

```
}
```

Attachment

```
public class Boat {

    private int capacity;
    private String name;
    private int idNum;

    public Boat(int max_people, String nickname, int num) {
        capacity = max_people;
        name = nickname;
        idNum = num;
    }

    public boolean equals(Object obj) {
        if (obj instanceof Boat) {
            Boat boat = (Boat) obj;
            if (this.idNum == boat.idNum) {
                return true;
            }
        }
        return false;
    }

    public String getName() {
        return name;
    }

    public int getCapacity() {
        return capacity;
    }

    public String toString() {
        return name + "-" + idNum + "-" + capacity;
    }
}
```

```
public class Size {

    public int thickness;
    public int radius;

    public Size() {
        thickness = 0;
        radius = 0;
    }
}
```

```java
public class EarRing {

    private Size size;
    public EarRing(Size s) {
        this.size = new Size();
        this.size.radius = s.radius;
        this.size.thickness = s.thickness;
    }

    public String toString() {
        return String.format("%d, %d",
                this.size.radius, this.size.thickness);
    }
}
```

```java
public class LipRing {

   private Size size;
    public LipRing(Size s) {
        this.size = new Size();
        this.size = s;
    }

    public String toString() {
        return String.format("%d, %d",
                this.size.radius, this.size.thickness);
    }
}
```

```java
public class NavelRing {

    private Size size;
    public NavelRing(Size s) {
        this.size = s;
        this.size = new Size();
    }

    public String toString() {
        return String.format("%d, %d",
                this.size.radius, this.size.thickness);
    }
}
```

```java
public class NoseRing {

    private Size size;
    public NoseRing(Size s) {
        this.size = s;
    }

    public String toString() {
        return String.format("%d, %d",
                this.size.radius, this.size.thickness);
    }
}
```

```
import java.util.NoSuchElementException;


public class FractionTokenizer {
    private static final String DEFAULT_DELIMITER = ",";

    private int next;
    private String[] tokens;

    public FractionTokenizer(String str) {
        if (str == null) {
            tokens = new String[0];
        } else {
            tokens = str.split(DEFAULT_DELIMITER);
        }
        next = 0;
    }

    public boolean hasNext() {
        return next < tokens.length;
    }

    public double next() throws NoSuchElementException {

        double denominator, numerator, result;
        String token;
        String[] parts;

        if (!hasNext()) {
            throw new NoSuchElementException();
        }

        token = tokens[next];
        next++;

        if ((token.length() < 3) || (token.indexOf('/') < 0)) {
            result = 0.0;
        } else {
            parts = token.split("/");
            numerator = Double.parseDouble(parts[0]);
            denominator = Double.parseDouble(parts[1]);

            result = numerator / denominator;
        }

        return result;
    }
}
```