CS 1331 Fall 2016 Exam 2

Fall 2016

ANSWER KEY

- Failure to properly fill in the information on this page will result in a deduction of up to 5 points from your exam score.
- Signing signifies you are aware of and in accordance with the **Academic Honor Code of Georgia Tech** and that you will not discuss this exam with other students.
- Calculators and cell phones are NOT allowed.
- This is an object-oriented programming test. Java is the required language. Java is case-sensitive. DO NOT WRITE IN ALL CAPS. A Java program in all caps will not compile. Good variable names and style are required. Comments are not required.

Question	Points per Page	Points Lost	Points Earned	Graded By
Page 1	16	-	=	
Page 2	6	-	=	
Page 3	10	-	=	
Page 4	14	-	=	
Page 5	3	-	=	
Page 6	14	-	=	
TOTAL	78	-	=	

1. **Multiple Choice** Circle the letter of the correct choice.

[2] (a) A Polymorphic object is known as _____ at runtime

A. Static
B. Dynamic
C. Transformed
D. Poly

[2] (b) The reserved word used for the lower class in the inheritance hierarchy is

A. Inherits
B. Extends
C. Receives
D. Implements

[2] (c) The type of relationship that exists in inheritance is

A. Has-a
B. Can-be
C. Will-be
D. Is-a

[2] (d) To create a polymorphic Television object that is an Electronic object at compile time use:

A. Television tele = new Electronic()
B. Electronic el = new Television()
C. Television tele = new Television()
D. Electronic el = new Electronic()

[2] (e) True or False: Java supports multiple inheritance.

A. True
B. False
C. Only if both inherited classes are abstract classes.

[2] (f) When a class hides information from its users it is taking part in a process known as ___

A. Abstraction
B. Instantiation
C. Encapsulation
D. Inheritance
E. Polymorphism

[2] (g) Given the following enum class and a variable myTA that has been declared to be of that class: public enum TA { MARC, FARHAN, HARSH, CHRIS, ANIRUDDHA, TAYLOR; } What should be placed in a conditional statement to correctly check if myTA is Taylor?

A. myTA == TAYLOR
B. new TA(TAYLOR)
C. myTA == TA.TAYLOR
D. myTA == new TA("TAYLOR")
E. myTA.equals("TAYLOR")

[2] (h) Car c = new Car(); is an example of what? Choose the best answer

A. Abstraction
B. Instantiation
C. Encapsulation
D. Inheritance
E. Polymorphism

 Points available: 16 - points lost: _____ = points earned: _____. Graded by: _____

[2] (i) Assume the following code has been executed:

```
int[] nums = {3, 1, 4, 1, 5, 9};
int[] numsCopy = nums;
int[] otherNums;
nums[3] = 7;
numsCopy[3] = 9;
```

What is the value of nums[3] ?

A. 7
B. 4
C. 9
D. 1

[2] (j) Now assume that this line of code has been run (after the above code has run):

```
nums[4] = numsCopy[4] * 2;
```

What is the value of numsCopy[4] ?

A. 5
B. 10
C. 18
D. 9

[2] (k) Next, the following line of code is run:

```
otherNums = new int[6];
for(int i = 0; i < nums.length; i++) {
    otherNums[i] = nums[i];
}
otherNums[5] = 6;
nums[5] = 2;
```

Which of the following are true?

A. otherNums[5] == 2 && numsCopy[5] == 6
B. otherNums[5] == 6 && numsCopy[5] == 9
C. otherNums[5] == 2 && numsCopy[5] == 2
D. otherNums[5] == 6 && numsCopy[5] == 2

 Points available: 6 - points lost: _____ = points earned: _____. Graded by: _____

2. Consider the following classes. You can assume each class has been properly created and all required methods have been fully implemented:

```
public interface Fightable
    * boolean canFight()

public abstract class Superhero implements Fightable
    * public abstract void fightCrime()

public class TeenTitan extends Superhero
    * public void describeCostume()

public class Raven extends TeenTitan
    * public void describeCostume()

public class Superman extends Superhero
    * public String getSecretIdentity()
    * public String describeCostume()
```

For each of the below statements, write whether it will produce a compile time exception, a runtime exception, or no exception. If it produces an exception describe the problem in one sentence. If it produces no exception, write which class's method will get executed. For example:

```
TeenTitan t = new TeenTitan();
t.describeCostume();
"No exception. TeenTitan's describeCostume"
```

[2] (a) TeenTitan t1 = new Raven();
t1.describeCostume();

```
No exception.
Raven's describeCostume method.
```

[2] (b) Superhero s1 = new Raven();
s1.describeCostume();

```
Compile time exception.
Superhero does not have a describeCostume method.
```

[2] (c) Superhero s2 = new Superhero();
s2.fightCrime();

```
Compile-time exception.
Superhero is abstract.
```

[2] (d) Superhero s3 = new Raven();
((Raven) s3).fightCrime();

```
No exception.
Raven's or TeenTitan's fightCrime method.
```

[2] (e) Raven r1 = new Raven();
r1.fightCrime();

```
No exception.
Raven's or TeenTitan's fightCrime method.
```

[2] (f) Superman sm1 = new SuperHero();
sm1.fightCrime();

```
Compile-time exception.
SuperHero is abstract.
```

[2] (g) Superhero sm2 = new Superman();
sm2.fightCrime();

```
No exception.
Superman's fightCrime method.
```

[2] (h) Raven r2 = new Superman();
r2.getSecretIdentity();

```
Compile-time exception.
Superman is not a subclass of Raven.
```

[2] (i) Raven r3 = new Raven();
r3.getSecretIdentity();

```
Compile-time exception.
Raven does not have a getSecretIdentitiy method.
```

[2] (j) Superhero s4 = new TeenTitan();
s4.fightCrime();

```
No exception.
TeenTitan's fightCrime method.
```

[2] (k) Superhero s5 = new Fightable();
System.out.println(s5.canFight());

```
Compile-time exception.
Fightable is an interface.
```

[2] (l) Superhero s6 = new Superman();
System.out.println(s6.canFight());

```
No exception.
Superman's canFight method.
```

3. **Short Answer** Answer the questions in the allotted space.

Given the following classes:

```
public class Dog {
	protected String sound;

	public Dog(String sound) {
		this.sound = sound;
	}

	public void setBark(String sound) {
		this.sound = sound;
	}
}

public class Chihuahua extends Dog {
	public Chihuahua(String sound) {
		super(sound);
	}

	public String bark() {
		return sound;
	}

	public String toString() {
		return sound;
	}
}

public class Driver {
	public static void main(String[] args) {
		Dog diego = new Chihuahua("woof woof");
		Chihuahua pedro = new Chihuahua("arf");
		System.out.println(diego.bark());
		pedro = (Chihuahua)diego;
		System.out.println(pedro.bark());
		diego.setBark("arooooof");
		System.out.println(pedro.bark());
	}
}
```

[3] (a) In the Driver class, locate the line of code that does not compile and fix it below. (Write the original line and what the line should have been.)
`System.out.println(diego.bark());\\`should be replaced with...
System.out.println(((Chihuahua)diego).bark());

 Points available: 3 - points lost: _____ = points earned: _____. Graded by: _____

[3] (b) Now assuming the change in part a has been made and the code compiles, write the output of the Driver class below.
woof woof
woof woof
arroooof

[3] (c) Now replace all the code in the Driver's main method with the following four lines of code. What would the output of this code be?

```
Dog[] dogs = {new Chihuahua("la"), new Dog("woof"), new Dog("arf")};
for (Dog d:dogs) {
        System.out.println(d);
}
```

la
Dog@fjdksljfdl
Dog@jfdklsjfd

[4] (d) Suppose we make the Dog class abstract (public abstract class Dog) and a method (public abstract void walk();) is added to it. Make any necessary changes to the Chihuahua class in the space given below. public void walk
//any implementation so long as they do not return any values

[4] (e) Suppose we change the class header to indicate that the Chihuahua class implements Comparable. What would we need to add to the Chihuahua class? You do not need to write any code for this question, just explain what would have to be added.
A compareTo method that returns an integer would have to be written/implemented and included in the Chihuahua class.

[15] 4. **Write the code.**

You are given an abstract circus performer class. Your task is to write a concrete subclass of CircusPerformer. You may name it whatever you like. However, your subclass must add the following functionality:

* Have a private instance variable numberOfTeeth of type int.
* Have a constructor that takes in a String name and an int numberOfTeeth that appropriately sets all instance data.
* Provide a new implementation of the toString method, that returns a String in the format, "name has numberOfTeeth teeth."

Write your code on the following page.
Remember that there will be additional requirement(s) since the CircusPerformer class is abstract!

```
public abstract class CircusPerformer {

    private String name;

    public CircusPerformer(String name) {
        this.name = name;
    }

    public void sayHi() {
        System.out.println("Hi! My name is " + name);
    }

    @Override
    public String toString() {
        return this.name;
    }

    public abstract void perform();
}
```

```
public class FireEater extends CircusPerformer { // 1pt

    private int numberOfTeeth; // 1pt

    public FireEater(String name, int numberOfTeeth) { // 1pt
        super(name); // 2pt
        this.numberOfTeeth = numberOfTeeth; // 1pt
    }

    @Override
    public void perform() { // 1pt
        System.out.println("Eats fire!"); // 1pt
    }

    @Override
    public String toString() { // 1pt
        return super.toString() + " has " + numberOfTeeth + " teeth."; // 1pt
    }
}
```

 Points available: 0 - points lost: _____ = points earned: _____. Graded by: _____