COP 3330 Exam #2 10/19/2026

Please print your name in CAPITAL letters.

Last Name: _____, First Name: _____

Part I: Class Design with Inheritance

The first few questions of this part will deal with adding methods to a Car class which is shown below, partially filled in:

```
public class Car {

	protected double tankSize;
	protected double galGasLeft;
	protected double mpg;
	protected double odometer;
	protected double expenses;

	public Car(double myTankSize, double myMPG) {
		tankSize = myTankSize;
		galGasLeft = tankSize;
		mpg = myMPG;
		odometer = 0;
		expenses = 0;
	}

	public double fillGas(double numGals, double costPerGal) {
		// Question #1 Code
	}

	public double drive(double miles) {
		// Question #2 Code
	}

	public void park(double hours, double costPerHour) {
		// Question #3 Code
	}
}
```

Here is what each instance variable represents:

tankSize – how big the gas tank is, in gallons.
galGasLeft – the number of gallons of gas left in the gastank.
mpg – the miles per gallon the car gets when driven.
odometer – the number of miles the car has been driven in its history
expenses – the amount of money that has been spent on the car (not including its purchase) to use the car (includes buying gas, parking and related expenses)

1) (4 pts) Describe, in English, the initial state of a Car object that is created by the provided constructor.

2) (10 pts) Complete the method fillGas for the Car class. The method takes in two doubles: the number of gallons of gas to fill and the cost of gas per gallon, respectively. **You may assume both are positive.** Your method should update the Car so that the desired amount of gas is added, provided that it doesn't overflow the gastank. In that case, just fill the tank. Your method should then return the number of gallons of gas that were actually added to the Car object. Make sure to modify all relevant components of the Car object.

```
public double fillGas(double numGals, double costPerGal) {









}
```

3) (10 pts) Complete the method drive for the Car class. The method takes in a single double representing the number of miles to drive the car. **You may assume this value is positive.** Your method should update the Car object so that the desired number of miles are driven, so long as there's enough gas in the car. If not, just drive the car until it has an empty gas tank. Your method should return the actual number of miles driven.

```
public double drive(double miles) {









}
```

4) (4 pts) Fill in the park method in the Car class. This method takes in the number of hours to park the car and the cost in dollars per hour for parking the car, respectively. **You may assume both are positive.** Your method should update the Car object (can be a single line of code) based on this operation.

```
public void park(double hours, double costPerHour) {









}
```

5) (12 pts) For this question, complete the new class McClaren, which inherits from Car. The McClaren class should include one new instance variable: **`fastMPG`** which represents the miles per gallon the car gets when going fast. Note: the race method in the class is different than the drive method because the miles per gallon the McClaren gets when racing is equal to fastMPG and not mpg.

```
public class McClaren extends Car {

      private double fastMPG;

      public McClaren(double myTankSize, double myMPG, double raceMPG) {









      }

      public double race(double miles) {

















      }
}
```

6) (25 pts) What is the output of the code handout? (It's a separate piece of paper, front and back.) The output is comprised of 12 lines. Each line has a letter print out first (A, B or C), followed by either 1, 2 or 3 numbers. (Note: A is followed by 1 number, B by 2 and C by 3.) The letters in the output are provided for you, please fill in each slot with a single number that gets printed out. (Note: Each number is worth 1 point. In the real print out each line prints out one after another, but for ease of writing, an extra blank line between each output line is shown.)

```
A __

A __

B __ __

C __ __ __

C __ __ __

A __

A __

B __ __

C __ __ __

C __ __ __

C __ __ __

B __ __
```

7) (20 pts) The following interface requires two methods:

```
public interface Rechargeable {
    public void use(double hours);
    public void recharge(double hours);
}
```

For this question you will write a complete class that implements this interface called Phone. For simplicity's sake, the Phone class will have a single instance variable of type double, `charge`, which is initially set to 1 by the default constructor. **This variable must always be in between 0 and 1, inclusive.** Assume that in 24 hours a Phone will go from full charge (1) to no charge (0) when used and assume that it takes 8 hours total to recharge the battery from 0 (no charge) to 1 (full charge). Write the complete class (declaration, instance var, constructor, use, recharge) on the next page.

8) (10 pts) You are designing a Coordinate class that stores locations on a grid where the row is designated by an uppercase letter and the column is designated by a positive integer. You would like for this class to implement the Comparable interface such that Coordinates that start with an earlier letter in the alphabet come before those which start with a later letter. If two Coordinates have the same letter, then the one that has a smaller number should come first. Below is an incomplete version of the Coordinate class. Complete the compareTo method in the class:

```
public class Coordinate implements Comparable<Coordinate> {

    private char row;
    private int  col;

    public Coordinate(char r, int c) {
        row = r;
        col = c;
    }

    public int compareTo(Coordinate other) {









    }
}
```

10) (5 pts) In what country were Swedish fish originally made, back in 1957?

_____

Java Language Reference

Math Class

Modifier and Type	Method	Description
`static int`	`max(int a, int b)`	Returns the greater of two int values.
`static int`	`min(int a, int b)`	Returns the smaller of two int values.

```
// Arup Guha
// 10/16/2026
// Code for Tracing Question #6 for COP 3330 Exam #2
class A {

	protected int x;

	public A() {
		x = 12;
	}

	public A(int myx) {
		x = myx;
	}

	public void f() {
		x--;
	}

	public void f(int a) {
		x -= a;
	}

	public String toString() {
		return "A "+x;
	}
}
class B extends A {

	protected int y;

	public B() {
		y = 2;
	}

	public void f() {
		x+=2;
	}

	public B(int first, int second) {
		super(first);
		y = second;
	}

	public void f(int a) {
		super.f();
		y += a;
	}

	public void g() {
		y += x;
	}

	public String toString() {
		return "B "+x+" "+y;
	}
}
```

```
public class C extends B {

	private int z;

	public C() {
		z = 4;
	}

	public C(int first) {
		super();
		z = first;
	}

	public void g() {
		super.g();
		z += y;
	}

	public String toString() {
		return "C "+x+" "+y+" "+z;
	}

	public static void main(String[] args) {
		A[] items = new A[5];
		items[0] = new A();
		items[1] = new A(7);
		items[2] = new B(3, 6);
		items[3] = new C();
		items[4] = new C(8);
		for (int i=0; i<items.length; i++)
			System.out.println(items[i]);

		for (int i=0; i<items.length; i++) {
			if (items[i] instanceof B)
				((B)items[i]).g();
			else if (i%2 == 0)
				items[i].f();
			else
				items[i].f(i+4);
		}
		for (int i=0; i<items.length; i++)
			System.out.println(items[i]);

		items[4].f(5);
		System.out.println(items[4]);
		items[2].f();
		System.out.println(items[2]);
	}
}
```