**AP Computer Science Name:**

**Inheritance Worksheet**

1)Write the superclass Animal which has two methods eat() and run(). The eat() method prints out the message “Animal is eating” and the run() method prints out “Animal is running”.

Write the subclasses Dog and Cat which **extends** the Animal class. The Dog class overrides the run() method and prints out “Dog is running”. Dog has one additional method bark() which prints out “Woof! Woof!”. The Cat class overrides the eat method and prints out “Cat is eating” and has an addition method jump() which prints out “Cat jumps”. **No constructors are needed.**

**Fill in the code below.**

public class Animal {

public void eat(){

System.out.println(“Animal is eating.)

}

public void run(){

System.out.println(“Animal is running.)

}

}

public class Dog extends Animal

{

@Override

public void run(){

System.out.println(“Dog is running.”);

}

public void bark(){

System.out.println(“Woof! Woof!”);

}

}

public class Cat extends Animal

{

@Override

public void eat(){

System.out.println(“Cat is eating.)

}

public void jump(){

System.out.println(“Cat jumps.)

}

}

2) Find the errors in the following driver class testing the Animal, Dog and Cat classes above.

public static void main(String[] args)

{

Animal a=new Animal();

Cat b=new Cat();

Animal c=new Dog();

Dog d=new Animal(); //error! Not every animal is a dog

Dog e=new Dog();

Cat f=new Dog(); //error! Can’t cast sideways.

Animal g=new Cat();

Animal h=new Dog();

a.eat();

b.run(); //ok, inherited

a.jump(); //error, no jump for Animal

b.bark(); //error, no bark for Cat

e.bark(); //ok

b.jump(); //ok

g.eat(); //ok, which one? Cat’s version

h.run(); //ok, which one? Dog’s version

**//new problem**

**g.jump(); //error, Animal has no jump method**

**//even though g holds a Cat object**

**//must cast to call jump**

**((Cat) g).jump();//notice the double sets of ().**

}

3) Assume Circle has been correctly implemented. Use it to implement the subclass Cylinder and the class TestCylinder. For the Circle class, assume that the default values for radius is 1 and color is an empty String. For the Cylinder class, the default value for height is 1.

public class Circle{

private double radius;

private String color;

public Circle(){}

public Circle(double r){}

public Circle(double r, String c)

public void setRadius(double r){}

public double getRadius(){}

public void setColor(String c){}

public String getColor(){}

public double getArea(){}

public String toString()

{//e.g. ”Circle: radius=4 color=red”}

}

public class Cylinder extends Circle

double height;

public Cylinder(){

super();

height=1;

}

public Cylinder(double radius){

super(radius);

height=1;

}

public Cylinder(double r, double h){

super(r);

height=h;

}

public Cylinder(double r, double h, String c){

super(r,c);

height=h;

}

public double getHeight(){

return height;

}

//@Override is an annotation for the compiler.

//method returns **surface area**

**@Override**

public double getArea(){

return 2\*Math.PI\*getRadius()\*getRadius()

+2\*Math.PI\*getRadius()\*height;

**or return 2\*super.getArea()**

**+2\*Math.PI\*getRadius()\*height;**

}

@Override

//returns for example

// “Cylinder: subclass of Circle: radius=3 color=”red” height=4”

**// must use toString of Circle.**

public String toString(){

return “Cylinder: subclass of” +super.toString()+”height=”+height;

}

**//must call getArea() of Circle.**

public double getVolume(){

return super.getArea()\*height;

}

public class TestCylinder(){

public static void main(String[] args){

//create three circles c1, c2 and c3 using

// each of the three constructors.

Circle c1=new Circle();

Circle c2=new Circle(5);

Circle c3=new Circle(5,”red”);

//create a cylinder cy1 with default

//radius, color and height.

Cylinder cy1=new Cylinder();

//create a cylinder cy2 with default

//radius, color and height=12.

Cylinder cy2=new Cylinder(1,12);

//create a cylinder cy3 with default

//color, radius=5 and height=10.

Cylinder cy3=new Cylinder(5.0,10.0,””);

//create a cylinder cy4 with

//color= “red”, radius=2 and height=7.

Cylinder cy4=new Cylinder(2,7,”red”);

//call getArea() of c1 and cy1. Which getArea()?

c1.getArea();

cy1.getArea();

}

}