Actividad 1:

*package* Ejercicio1\_24abril;

*class* Letter {

*public* String a() {

*return* "A";

}

*public* String b() {

*return* "B";

}

*public* String c() {

*return* "C";

}

}

*class* A *extends* Letter { }

*class* B *extends* Letter { }

*class* C *extends* Letter { }

*package* Ejercicio1\_24abril;

*public class* Main {

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

A objA = *new* A();

B objB = *new* B();

C objC = *new* C();

}

}

Actividad 2:

*public class* Punto2D {

*protected double* X;

*protected double* Y;

*public* Punto2D(*double* x, *double* y) {

X = x;

Y = y;

}

*public double* getX() {

*return* X;

}

*public double* getY() {

*return* Y;

}

*public void* mostrar() {

System.***out***.println("Punto2D: (" + X + ", " + Y + ")");

}

}

*public class* Punto3D *extends* Punto2D {

*private double* Z;

*private* String Color;

*public* Punto3D(*double* x, *double* y, *double* z, String color) {

*super*(x, y);

Z = z;

Color = color;

}

*public double* getZ() {

*return* Z;

}

*public* String getColor() {

*return* Color;

}

@Override

*public void* mostrar() {

System.***out***.println("Punto3D: (" + X + ", " + Y + ", " + Z + ")" + " Color: " + Color);

}

}

*package* Ejercicio2\_24abril;

*public class* Main {

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

Punto2D p2 = *new* Punto2D(1, 2);

p2.mostrar();

Punto3D p3 = *new* Punto3D(1, 2, 3, "Rojo");

p3.mostrar();

}

}

Actividad 3:

No es correcto que Orden herede de Cliente porque una orden no es un cliente, sino que pertenece a un cliente. Heredar implicaría que Orden es un tipo de Cliente y mezclaría responsabilidades, lo que dificulta el mantenimiento y rompe principios del diseño orientado a objetos. Si existiesen otras clases que heredan de clientes por ejemplo, y se implemente algún método nuevo que es abstracto (es decir, debe heredarse si o si), esto rompería la clase Orden ya que esta debería heredar si o si ese método. En su lugar, se debe usar composición, haciendo que Orden tenga un atributo que instancie Cliente.