TIPO ABSTRACTO DE DATOS

Un tipo abstracto de dato, es una agrupación de elementos con características similares, que no son más que conceptos escritos de tal forma que un ordenador pueda entender.

Por ejemplo, en el proyecto NowMeal teníamos la clase Transporte.java:

```java
public abstract class Transporte {

    // atributos
    private String codigo;
    private Mapa mapa;

    // constructor
    public Transporte(String codigo, Mapa mapa) {
        this.codigo = codigo;
        this.mapa = mapa;
    }

    // metodos para calcular el coste
    public double coste(String codPosDestino) {
        return coste(codigo, codPosDestino);
    }

    public abstract double coste(String codPosOrigen, String codPosDestino);

    //getters
    public Mapa getMapa() {
        return mapa;
    }

    public String getCodigo() {
        return codigo;
    }
}
```

Si nos fijamos, la clase en es abstracta, esto se debe a que el método **coste(String, String)** es de tipo abstracto también, y **cuando tenemos un método abstracto o más, la clase tiene que ser abstracta automáticamente**, como veis un Transporte es solo un concepto, una idea, en cambio, un Coche o una Furgoneta propia/subcontratada son “reales”, por eso mismo Transporte.java es abstracta, y las otras clases (menos Furgoneta.java) no lo son.

Furgoneta.java es abstracta ya que no queremos crear un Objeto de este tipo, solo queremos crear **FurgonetaPropia** o **FurgonetaSubcontratada**.

A su vez, cuando declaramos un método como abstracto, las clases hijas o subclases tienen que implementar este método, a no ser, que la subclase también sea abstracta.

Existen otros tipos abstractos de datos, por ejemplo, las clases IList, Stack, IQueue y Node son abstractas, ya que son meras representaciones de una Lista, una Pila, una Cola y un Nodo, pero no son la definición del objeto en sí, no tenemos que entender cómo funciona esto, con saber lo que hacen éstas clases con el PDF 17, nos vale.