Enumeraciones

Definición y estructura

- Una enumeración define un tipo de dato que puede tomar una lista de valores posibles

- Definición:

```
modificador enum NombreEnumeracion{
    valor1, valor2, valor3;
}
```

Ejemplo

```
public enum Dias{
    LUNES, MARTES, MIERCOLES, JUEVES;
}
```

- Una variable de un tipo enumerado sólo puede tomar los valores definidos en dicho tipo:

```
Dias obj;
obj=Dias.LUNES;
```

Uso en instrucciones de control

- La variables de tipo enumerado se pueden utilizar con el operador == para comprobar la igualdad:

```
if(obj==Dias.MARTES){
    …
}
```

- También puede utilizar un tipo enumerado en un switch:

```
switch(obj){
    case LUNES:
        …
    case MARTES:
        …
}
```

como valores de los case se emplean **directamente las constantes** de la enumeración

Constructores

- Una enumeración también puede tener constructores.
- Cada constante de la enumeración tiene asignado los valores que serán pasados a los parámetros del constructor:

```
enum Dias{
    LUNES(20), MARTES(8), MIERCOLES(12), JUEVES(1);
    int data;
    Dias(int n){
        data=n;
    }
}
```

Los constructores son implícitamente privados. No admiten otro modificador

- Al crear un objeto enumerado se llama al constructor con los valores asociados a la constante:

```
Dias d=Dias.MIERCOLES;
System.out.println(d.data); //12
```

Métodos de una enumeración

- Toda enumeración es una subclase de java.lang.Enum, que proporciona los siguientes métodos:
  - values(). Método estático que devuelve un array con todos los valores de la enumeración
  - name(). Cadena de caracteres con el nombre del valor
  - ordinal(). Posición del valor dentro de la enumeración, siendo 0 la posición del primero
  - toString(). Devuelve el mismo resultado que name()

```
for(var v:Dias.values()) {
    System.out.println("Name: "+v.name()+
          " Ordinal:"+v.ordinal()+" toString: "+v.toString());
}
```

⇨

Name: LUNES Ordinal:0 toString: LUNES
Name: MARTES Ordinal:1 toString: MARTES
Name: MIERCOLES Ordinal:2 toString: MIERCOLES
Name: JUEVES Ordinal:3 toString: JUEVES

Revisión conceptos

Dada la siguiente enumeración, indica que se mostrará al ejecutar el código que aparece a continuación:

```
enum Estados{
    ON(2), OFF(1), UNKNOWN(4);
}
```

```
for(var e:Estados.values()) {
    System.out.println("Val: "+e+" Ordinal:"+e.ordinal());
}
```


Respuesta

Se produce un **error de compilación** en la enumeración, ya que si se le asignan valores a cada uno de sus elementos es obligatorio definir un constructor que reciba como parámetro estos valores

Revisión conceptos

Dada la siguiente enumeración, indica que se mostrará al ejecutar el código que aparece a continuación:

```
enum Estados{
    ON(2), OFF(1), UNKNOWN(4);
    int s;
    Estados(int k){s=k;}
}
for(var e:Estados.values()) {
    System.out.println("Val: "+e.s+" Ordinal:"+e.ordinal());
}
```


Respuesta

```
Val: 2 Ordinal:0
Val: 1 Ordinal:1
Val: 4 Ordinal:2
```