U.D.6

Programación orientada a objetos. Objetos

(2ª parte: Utilización de los objetos)

PRO – IES EL MAJUELO

ÍNDICE

1. Declaración y creación

Características de los objetos:

- Se agrupan en tipos denominados “clases”.
- Permiten poder ocultar los datos que contiene.
- Sus valores determinan su estado.
- Pueden heredar los datos o funciones de otros objetos.
- Se comunican con otros objetos gracias al envío de mensajes.
- Tienen métodos que definen su comportamiento.
- Un método puede hacer uso de otros métodos.

Instanciación

- Declarar un objeto:

```
Alumno alumno1; // Clase Alumno
```

- Crear un objeto:

```
alumno1 = new Alumno(); // Constructor
// Se reserva espacio para guardar los datos //
  de dicho objeto
```

2. Utilización de métodos

- Para hacer uso de los métodos de una clase fuera de ella, solo hay que declarar y crear un objeto; a partir de aquí se pueden usar los métodos de dicho objeto:

```
nombre_objeto.nombre_metodo(argumentos);
```

- Para hacer uso de los métodos dentro de la propia clase, simplemente se indicará el nombre del método:

```
nombre_metodo(argumentos);
```

Parámetros

- Van separados por comas.

```
objeto.metodo("Jose Perez", 23, 1.78, 43);
void metodo (String nombre, int edad, float
                altura, int numpie)
```

- Son variables locales del método.
- No se puede declarar en el método ninguna otra variable con el mismo nombre.
- Un método puede devolver un único dato, a través de la instrucción `return;`.
  - Debe aparecer `return` si el tipo no es `void`.
  - Puede aparecer `return` en todos los casos.

3. Utilización de propiedades

- Para hacer uso de las propiedades de una clase fuera de ella solo hay que declarar y crear un objeto; a partir de aquí, podremos usar las propiedades de dicho objeto siempre que los modificadores de visibilidad lo permitan:

```
nombre_del_objeto.nombre_de_la_propiedad
```

  - Lo adecuado es que las propiedades de un objeto se usen dentro de la misma clase y para poder acceder a esas propiedades fuera de la clase se haga a través de los métodos.

- Para hacer uso de las propiedades dentro de la misma clase, simplemente se hace indicando el nombre de dicha propiedad.

4. Constructores

- Es un método especial que se llama igual que la clase y se ejecuta automáticamente en el momento en que se crea un objeto (*new*).
- Se utilizan para dar valores iniciales a las propiedades del objeto.
- Hay dos tipos:
  - Constructor por defecto: `nombreclase()`
    - Si la clase no tiene constructor, JAVA ejecuta su constructor por defecto (aunque no esté escrito).
  - Constructor definido: `nombreclase(argumentos)`
- No pueden devolver nada (ni siquiera se pone *void*).
- Si el constructor no da valores iniciales a todos los atributos de la clase, JAVA los inicializa con cero (numéricos) o *null* (referenciados).

5. Elementos estáticos

5.1. Propiedades estáticas

- Cada objeto tiene su propio espacio de memoria donde guarda sus datos (propiedades).
- De las funciones miembros (métodos) sólo existe una copia para todos los objetos que pertenecen a una determinada clase.
- Si un dato miembro de una determinada clase se declara *static*, significa que sólo existirá una copia de ese dato, el cual es compartido por todos los objetos de esa clase.
  - Existe aunque no se haya declarado ningún objeto de dicho tipo.
  - Un dato miembro estático es una variable asociada a la clase y no a ningún objeto.

5. Elementos estáticos
5.2. Métodos estáticos

- Es un método de clase, no de ningún objeto.

- Puede usarse sin tener declarado ningún objeto:

```
NombreClasePertenece.MetodoEstatico();
```

o bien

```
NombreObjeto.MetodoEstatico(); // Engañoso
```

- Dentro de un método estático sólo se pueden usar los miembros de la clase que sean estáticos.

6. Almacenamiento en memoria

- Si la variable es de un tipo básico del lenguaje, en el momento que se declara se reserva espacio en memoria para que pueda guardar datos.

- Si la variable es un objeto, en el momento que se declara no se reserva espacio en memoria.
  - Para ello hay que hacer *new* o asignarle la dirección de otro objeto ya creado.

7. Destrucción de objetos

- Los objetos se destruyen automáticamente cuando el programa se sale del bloque donde fue declarado.
- Pero la liberación de memoria (que se reservó con new) se realiza a través de un programa llamado *Garbage Collector* (recolector de basura), integrado en la máquina virtual de JAVA.
  - Revisa la memoria y comprueba si aquellos espacios que hayan sido reservados con un new tienen algún objeto apuntándoles, liberándolos en su defecto.
  - Se ejecuta periódicamente. También puede invocarse: `System.gc()`.
  - Cuando actúa, se ejecuta automáticamente el método *protected* "`void finalize()`" siempre que esté implementado dentro de la clase a la que pertenece el espacio de memoria que se está liberando.
  - Otra forma de hacer que se ejecute sería asignar *null* a un objeto (el objeto ya no apunta al espacio de memoria reservado).