Preguntas

A. ¿Qué pasaría si se pierde la referencia de un objeto de tipo Persona?

Si se pierde la referencia de un objeto Persona, y no hay otras referencias apuntando a ese objeto, el garbage collector lo marcará como elegible para eliminación. Esto significa que eventualmente el objeto será destruido y su método **finalize()** será invocado, imprimiendo el mensaje "**Matando a: [nombre]".**

B. ¿Cómo podríamos conocer el nombre del dueño de la variable auto de la línea 11 del método main?

El dueño del vehículo auto puede obtenerse utilizando el método getNombre() del objeto Persona asociado al atributo dueno. Ejemplo:

```
System.out.println("Dueño del auto: " + auto.getDueno().getNombre());
```

C. ¿De qué manera podemos agregar un dueño al Vehiculo de la línea 13 del método main?

Podemos usar el método setDueno de la clase Vehiculo. Por ejemplo:

```
auto2.setDueno(personas[0]);
```

D. Usando la variable auto2 de la línea 13 del método main, obtenga el valor del atributo velocidadMaxima del motor del vehículo. Adjunte su propuesta.

Podemos obtener la velocidad máxima del motor utilizando el método getVelocidadMaxima() del enum Motor. Ejemplo:

```
int velocidadMaxima = auto2.getMotor().getVelocidadMaxima();

System.out.println("Velocidad máxima: " + velocidadMaxima);
```

E. Suponga que, al momento de perder la referencia al objeto, se borra del sistema, es decir el garbage collector es muy eficiente, ¿Qué imprimiría al ejecutar el método main por consola?

1. Al perder la referencia de personas[4], el objeto asociado a personas[4] será recolectado y se imprimirá:
   "Matando a: Alexander".

2. La línea personas[3].finalize(); también imprime explícitamente:
   "Matando a: Santiago".
3. Finalmente, se imprime la referencia a personas[3]:
   "Soy Santiago".

Salida esperada:

Matando a: Alexander

Matando a: Santiago

Soy Santiago

F. ¿Qué ocurre al momento de ejecutar la siguiente línea después de la línea 16 System.out.println(personas[1])? Explique.
La referencia de personas[1] fue asignada a personas[2] en la línea personas[1] = personas[2];. Por lo tanto, al imprimir personas[1], se ejecutará el método toString() del objeto originalmente referenciado por personas[2]. Se imprimirá:

Soy Daniel

G. ¿Qué modificación al código debo hacer para que al momento de ejecutar System.out.println(auto2), me aparezca la placa del vehículo y el dueño del vehículo?
Debemos sobrescribir el método toString() de la clase Vehiculo. Ejemplo:

```
@Override

public String toString() {

        String duenoNombre = (dueno != null) ? dueno.getNombre() : "Sin dueño";

        return "Placa: " + placa + ", Dueño: " + duenoNombre;

}
```

H. Usando la variable auto de la línea 11 del método main, y usando el atributo dueno, asigne de mejorAmigo al tercer elemento del listado personas. Adjunte su propuesta.
Podemos acceder al atributo dueno de auto y llamar a setMejorAmigo para asignar el valor deseado. Ejemplo:

```
auto.getDueno().setMejorAmigo(personas[2]);
```

PREGUNTAS 2

A. Según el siguiente código, indique qué se imprime por consola y explique el porqué de cada línea donde se imprime.

- **char c = 'g';**: El tipo char se puede promocionar a int. Por lo tanto, llama a la función que recibe un int.
  Salida: "char : Entra a int: 103" (103 es el valor ASCII del carácter 'g').

- **short s = 2;**: El tipo short se promociona a int porque no hay un método activo que reciba un short.
  Salida: "short : Entra a int: 2".

- **byte b = 1;**: El tipo byte también se promociona a int por las mismas razones que el caso de short.
  Salida: "byte : Entra a int: 1".

- **long l = 999999999;**: El tipo long no tiene una conversión directa a int porque el rango de long es mayor. Por lo tanto, se promociona a double para invocar la función correspondiente.
  Salida: "long : Entra a double: 9.99999999E8" (notación científica).

- **int i = 51232;**: Coincide directamente con el método que recibe un int.
  Salida: "integer : Entra a int: 51232".

- **double d = 12.4;**: Coincide directamente con el método que recibe un double.
  Salida: "double : Entra a double: 12.4".

- **float f = 5.65f;**: El tipo float se promociona automáticamente a double porque no hay un método activo que reciba un float.
  Salida: "float : Entra a double: 5.65".

B. Realice los siguientes cambios, teniendo siempre como referencia el código inicial. Explique cómo y por qué cambia lo que se imprime por pantalla.

• Active la función que recibe un short.

• Active la función que recibe un float.

• Comente la función que recibe un double y active la que recibe un float.

• Comente todas las funciones, excepto la que recibe un double

1. **Activar la función que recibe un short:**

```
static String funcion(short a) {

    return "Entra a short: " + a;

}
```

Ahora, cuando se pasa un short, se llamará directamente al método que recibe un short. Esto cambia el resultado para short s = 2;.
Nueva salida para short: "short : Entra a short: 2".

2. **Activar la función que recibe un float:**

```
static String funcion(float a) {

    return "Entra a float: " + a;

}
```

Ahora, cuando se pasa un float, se llamará directamente al método que recibe un float. Esto cambia el resultado para float f = 5.65f;.
Nueva salida para float: "float : Entra a float: 5.65".

3. **Comentar la función que recibe un double y activar la que recibe un float:**

```
// static String funcion(double a) { ... }

static String funcion(float a) {

    return "Entra a float: " + a;

}
```

Esto afecta las variables double d y float f.

- Para float f = 5.65f;: La salida sigue siendo "float : Entra a float: 5.65".
- Para double d = 12.4;: No hay método que reciba un double, lo que resulta en un error de compilación porque no hay coincidencia directa.

4. **Comentar todas las funciones excepto la que recibe un double:**

```
static String funcion(double a) {

    return "Entra a double: " + a;
```

}

Ahora, todos los tipos que pueden promocionarse a double llamarán a este método.
Nueva salida:

- "char : Entra a double: 103.0"
- "short : Entra a double: 2.0"
- "byte : Entra a double: 1.0"
- "long : Entra a double: 9.99999999E8"
- "integer : Entra a double: 51232.0"
- "double : Entra a double: 12.4"
- "float : Entra a double: 5.65"