Andres Alejandro Rosero Toledo / Grupo 1

Ejercicio 1:

A) Cuando se pierde la referencia de un objeto de tipo persona, este se vuelve elegible para el garbage collector, el cual lo hará en un futuro, por ejemplo, en el código al asignarse null a ese campo, se pierde la referencia al objeto que se encontraba en ese campo, además si el objeto tiene un finalize( ), este se invocaría antes de que el recolector de basura pase.

B) Primero llamar al método getDueno de la clase del auto para obtener al objeto persona y luego llamar al método getNombre de la clase de la persona para obtener el nombre de dueño.

C) Primero elegimos una persona del arreglo de personas que queramos asignar como dueño, luego con auto2 llamamos al método setDueno y le pasamos como parámetro a la persona que elegimos.

D) Podría usar el método getVelocidadMaxima( ) del enumerado motor, para lo cual obtendría primero el motor de auto2 con el método de la clase vehículo llamado getMotor( ), así que si solo busco obtenerlo quedaría así: auto2.getMotor( ).getVelocidadMaxima( )

E) Imprimiría “Matando a: Santiago” y luego ya que el Garbage collector sería eficiente entonces imprimiría “Soy null”

F) Imprimiría “Soy Daniel” ya que se está llamando al método toString de ese objeto, el cual imprime un String y el atributo nombre de ese objeto y como en la línea anterior se está igualando el segundo elemento con el tercero del arreglo de personas entonces ese el segundo elemento pasa a tener el mismo nombre que tiene el tercero que en ese caso es Daniel.

G) Debería sobrescribir el método toString de la clase vehículo y entre lo que se va a imprimir, incluir el atributo placa de esa misma clase y obtener el nombre del atributo dueño.

H) auto.getDueno( ).setMejorAmigo(personas[2])

Ejercicio 2:

A) Se imprime:

“char : Entra a int: 103” ya que se aplica algo que se conoce como promoción de tipo, y es que char es esencialmente un numero entero que representa un carácter en la tabla de Unicode, en este caso g seria 103
“short : Entra a int: 2 “ Ya que int es el tipo inmediatamente superior a short
“byte : Entra a int: 1” Ya que int es el tipo inmediatamente superior a byte
“long : Entra a double: 9.99999999E8” Ya que int es un tipo que es inferior jerárquicamente en cuanto a rango de números y double es el tipo inmediatamente superior a long
“integer : Entra a int: 51232” Ya que es del mismo tipo del parámetro que acepta int
“double : Entra a double: 12.4” Ya que es del mismo tipo del parámetro que acepta double
“float : Entra a double: 5.650000095367432” Ya que int es un tipo que es inferior jerárquicamente en cuanto a rango de números y double es el tipo inmediatamente superior a float

B) -Al habilitar la función que recibe un short pasa que los que son iguales o inferiores al tipo short como los son **“short s = 2”** y **“byte b = 1”** entra ahí.

- Al habilitar la función que recibe un float pasa que los que son directamente de ese tipo o los que están por debajo de un float pero por encima de un int entra ahí, como lo son “**long l = 999999999”** y “**float f = 5.65f”**

-Al comentar la función que recibe un double y activar la que recibe un float pasa que como double no tiene ningún tipo “inmediatamente superior” entonces se marca un error en la línea 18, ya que no hay un método que reciba un double

- Al comentar todas las funciones, excepto la que recibe un double pasa que como todos los valores entran al método que recibe un double ya que quitando el que ya es un double, para los demás tipos de valores es el tipo de valor inmediatamente superior, y en el caso del char pasaría al método su representación como entero