Punto 2 Taller 4 Java

Ejercicio 1

A) En teoría, si un objeto Persona perdiese su referencia, debería ser eliminado por el garbage collector y ejecutar su finalize(), imprimiendo la línea “matando a: {nombre}”, pero en la práctica esto no sucede así, ya que el objeto no es eliminado sino hasta que se ejecute el garbage collector, lo que puede que no ocurra en la línea inmediatamente siguiente a donde se perdió la referencia.

B) Una manera de conseguir el nombre del dueno de auto sería mediante su método getDueno(), lo que retornaría “Soy Alexander” gracias al toString() de Persona. Una manera más específica sería mediante el método getNombre del objeto Persona asociado con auto: auto.getDueno().getNombre(), lo que solo imprimiría “Alexander”

C) Mediante el método setDueno() de la clase vehículo es posible agregar un dueno a auto2, pasando como argumento un objeto cualquiera de la lista personas, ya que, hasta este punto en el programa, no se ha hecho nulo a ningún elemento

D) auto2.getMotor().getVelocidadMaxima()

E) Se imprimiría: Matando a: Alejandro

Matando a: Daniel

Matando a: Alexander

Matando a: Santiago

Soy Santiago

F) Imprime: Soy Daniel. Esto sucede porque el apuntador personas[1] ahora apunta al mismo espacio de memoria que personas[2], que contiene al objeto de clase Persona de nombre “Daniel”. Al hacer esto, el objeto al que previamente apuntaba personas[1] (objeto Persona de nombre “Jaime”) pierde su referencia, lo que lo hace elegible para ser recolectado por el garbage collector

G) Habría que definir el método toString() para la clase vehículo y especificar que muestre el atributo placa y el atributo dueno, similar a:

```
public String toString() {
        return placa + " " + dueno;
   }
```

En el apartado de “dueno” retornaría null, ya que este objeto ha sido inicializado sin este atributo

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

Ejercicio 2

A) Este código imprime lo siguiente:

char : Entra a int: 103

Esta línea se da debido a que no hay una definición de función() que reciba un dato de tipo char, así que el compilador convierte el valor de c, o sea “g” a su equivalente en Unicode: 123, lo que es interpretado como un int

short : Entra a int: 2
byte : Entra a int: 1

En estos dos casos, los valores s y b entran a la definición del método funcion() donde admite un dato de tipo int, ya que no hay definiciones (no comentadas) de este que admitan datos de tipo short o byte, siendo int el tipo de dato inmediatamente siguiente en la jerarquía de tamaños

long : Entra a double: 9.99999999E8

Por ahora solo hay dos definiciones del método funcion(): Una que admite un dato de tipo int y otra que admite de tipo double. Por esto, si se invoca a funcion() y se pasa como parámetro a un byte, short o int, entrarán a la definición que reciba un entero, ya que hay suficiente espacio de memoria para almacenarlos. En camibo si se pasan datos de tipo long, float o double, estos entrarán a la definición que permite datos de tipo double por la misma razón, como es en este caso

integer : Entra a int: 51232
double : Entra a double: 12.4

En estos casos, el dato entra directamente a la definición que admite su mismo tipo, ya que coinciden

float : Entra a double: 5.650000095367432

El dato de tipo float entra a la definición que admite double ya que es el tipo de dato disponible inmediatamente superior

B) Activando la función que recibe short:

Cambios evidenciados:

short : Entra a short: 2
byte : Entra a short: 1

En estos dos casos, se entra a la definición que recibe un short, ya que el dato disponible de tamaño inmediatamente superior para s y b es short

Activando la función que recibe float:

Cambios evidenciados:

long : Entra a float: 1.0E9
float : Entra a float: 5.65

En el caso de long, el tipo de dato float se vuelve el tipo disponible inmediatamente superior en tamaño, pero debe ser impreso de esta forma para acomodarse al formato del tipo de dato float.

En el caso de float, entra a esta definición ya que los tipos de datos coinciden.

Comentando la función que recibe double y activando la que recibe float:

Cambios evidenciados: **Error**

Esto se debe a que la variable “d” es de tipo double, el tipo de dato de mayor tamaño en nuestro programa, pero no hay definición del método funcion() que admita double o superior, siendo la definición más grande la de float, que sigue siendo menor en tamaño

Habilitando únicamente la función que recibe double:

Cambios evidenciados:

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: 5.650000095367432

Aquí, todos los datos entran a esta definición, ya que double es de mayor tamaño que todos, por lo que se permite almacenarlos, agregando decimales para convertirlos correctamente a este tipo de dato