Taller # 4 – JAVA

By: Andrés Felipe Muñoz Ortiz

Grupo: 1

Solución Ejercicio1:

A. Sí se pierde la referencia a un objeto y no tiene mas apuntadores, entonces este y su espacio de memoria será eliminado por el GarbaCollector
B. Podemos usar el método getDueño() de la clase Vehículo, y lluego el método getNombre de la clase Persona
   System.out.println(auto.getDueno().getNombre());

C. Para agregar un dueño al Vehiculo auto2, utilizamos el método setDueno, que asigna el objeto Persona proporcionado al atributo dueno del vehículo, lo que nos permite establecer o cambiar el propietario después de crear el objeto Vehículo
   Seria: auto2.setDueno(personas[1]);
D. int velocidadMaxima = auto2.getMotor().getVelocidadMaxima();
   System.out.println("La velocidad máxima del motor es: " + velocidadMaxima);

E. Sí el garbaje collecto fuera muy eficiente entonces eliminaría los objetos no referenciados inmediatamente, imprimiendo:
   Matando a: Alejandro
   Matando a: Daniel
   Matando a: Alexander
   Soy Santiago

F. Se imprimirá Soy Daniel, ya que personas[1] apunta al mismo objeto que personas[2] después de la asignación en la línea 16 por el método toString()
G. Habría que sobrescribir el método toString() en la clase Vehiculo para incluir la placa y el dueño del vehículo en la salida. Quedaría:

```
public String toString() {
    if (dueno == null) {
        return "Placa: " + placa + ", Dueño: Sin dueño";
    } else {
        return "Placa: " + placa + ", Dueño: " + dueno.getNombre();
    }
}
```

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

auto.getDueno() obtiene el objeto Persona que es el dueño del vehículo y setMejorAmigo(personas[2]) asigna el tercer elemento del listado personas como el mejor amigo del dueño del vehículo.

Solución Ejercicio 2:

A. char: Entra a int: 103, esto por el valor ASCII del carácter g, y porque de char pasa a int por jerarquía
B. short : Entra a int: 2, porque el mas cercano a short es int en escala de jerarquías
C. byte : Entra a int: 1, porque es el mas cercano en la escala de jerarquías, byte → short → int
D. long : Entra a double: 9.99999999E8, porqué de long pasa a double debido a que long esta por encima de int en la escala de jerarquías
E. integer : Entra a int: 51232, pues el método literal tiene atributo int
F. double : Entra a double: 12.4, pues esta literal definido como atributo del método
G. float : Entra a double: 5.65, pues se llevan una escala en la jerarquía

2:

a. Ahora, cuando se pase un valor de tipo short, este llamará a funcion(short a) en lugar de ser promovido a int pues son del mismo tipo
b. Ahora, los valores de tipo float llamarán a funcion(float a) en lugar de ser promovidos a double, como en static String funcion(float a) {
   return "Entra a float: " + a;
   }
   float : Entra a float: 5.65
c. Todos los valores que anteriormente llamaban a funcion(double a) ahora serán promocionados a float si es posible, o a int si los tipos están antes del float
d. Todos los valores que pueden ser promocionados a double llamarán a funcion(double a) y todos se ejecutaran con este ya que es el mas alto en la jerarquía