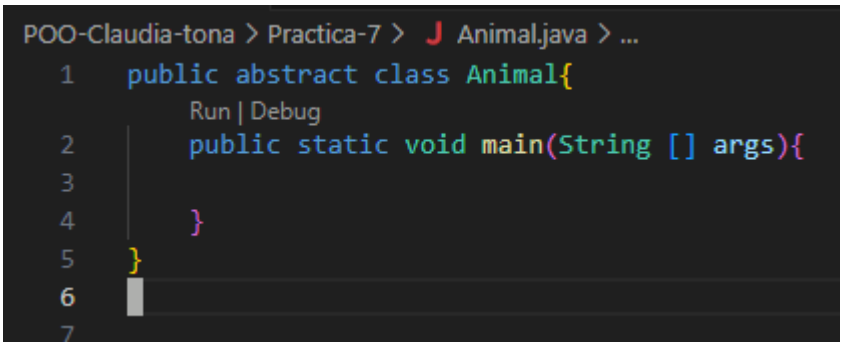
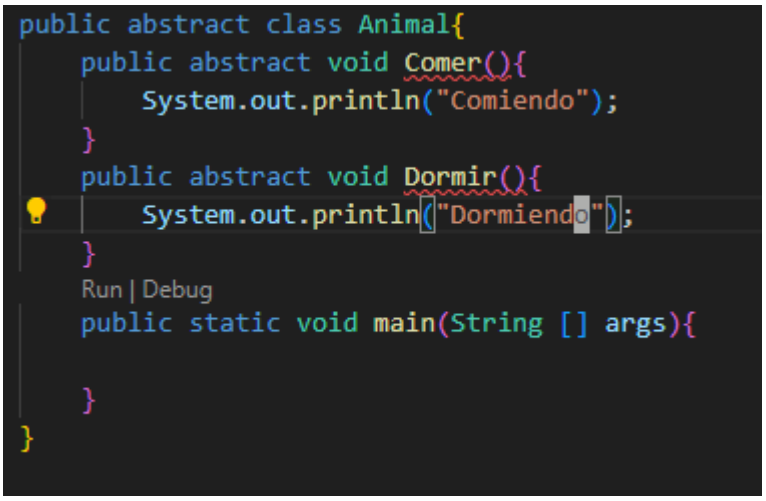
PRÁCTICA 07 - PROGRAMACIÓN ORIENTADA A OBJETOS

Alejandro Vaquera López

Ejercicio 1: Intente crear una instancia de un objeto Animal después de declararlo abstracto. ¿Se compila? Haga que los métodos de comer y dormir de la clase Animal sean abstractos como se muestra. Ahora intente ejecutarlo y observe si hay algún error.



El programa corre sin problemas.



Los métodos abstractos marcan error.

Ejercicio 2:Haga una matriz “Animal” y ejecútela con muchos objetos diferentes de sus subclases y use la instrucción <<for>> para revisar sus elementos y hacer que cada uno de ellos * llame al método “eat ()”. ¿Tiene el comportamiento que esperabas?Usemos esta misma matriz, pero esta vez intentemos una instrucción << para cada >>, ¿funciona de la misma manera?

POO-Claudia-tona > Practica-7 > AnimalMain.java > ...

```java
abstract class Animal{
    public void Comer(){
        System.out.println(x:"Comiendo");
    }
    public void Dormir(){
        System.out.println(x:"Dormiendo");
    }
}

class Tiger extends Animal{
    public void Roar(){
        System.out.println(x:"roar");
    }
}

class Koala extends Animal{
    public void Climb(){
        System.out.println(x:"climb");
    }
}

public abstract class AnimalMain{
    Run | Debug
    public static void main(String [] args){
        Animal [] animales = {new Tiger(), new Koala()};
        for(Animal animal : animales){
            animal.Comer();
        }

    }
}
```

```
C:\Users\Vacintosh\Documents\projects\POO-Claudia-tona\Practica-7>java AnimalMain
Comiendo
Comiendo
```

Ejercicio 3: Cree una clase Car que implemente la interfaz definida anteriormente y ejecute un código para verificar el resultado de sus métodos.

```
C:\Windows\System32\cmd.exe
C:\Users\Vacintosh\Documents\projects\POO-Claudia-tona\Practica-7>javac Safety.java

C:\Users\Vacintosh\Documents\projects\POO-Claudia-tona\Practica-7>java CarMain
Duster Plymouth 1974 is running
Accelerating
Seatbelts are working fine
Seatbelts locked

C:\Users\Vacintosh\Documents\projects\POO-Claudia-tona\Practica-7>
```

POO-Claudia-tona > Practica-7 > CarMain.java > Car

```java
class Car implements Safety{
    String model;
    int mileAge;
    double speed;

    public Car(String model){
        this.model = model;
    }

    public void start(){
        System.out.println(model + " is running");
    }

    public void accelerate(){
        System.out.println(x:"Accelerating");
        speed++;
    }

    public boolean checkSeatBelts(){
        System.out.println(x:"Seatbelts are working fine");
        return true;
    }

    public void lockSeatBelts(){
        System.out.println(x:"Seatbelts locked");
    }
}
```

Ejercicio 4: Usando su clase de automóvil y la interfaz de seguridad, ejecute el código anterior y verifique el resultado. Si hay errores, ¿cómo los solucionaría?

```
1   public interface SafetyLights extends Safety{
2       public void checkLights();
3       public void toggleLights();
4   }
5
```

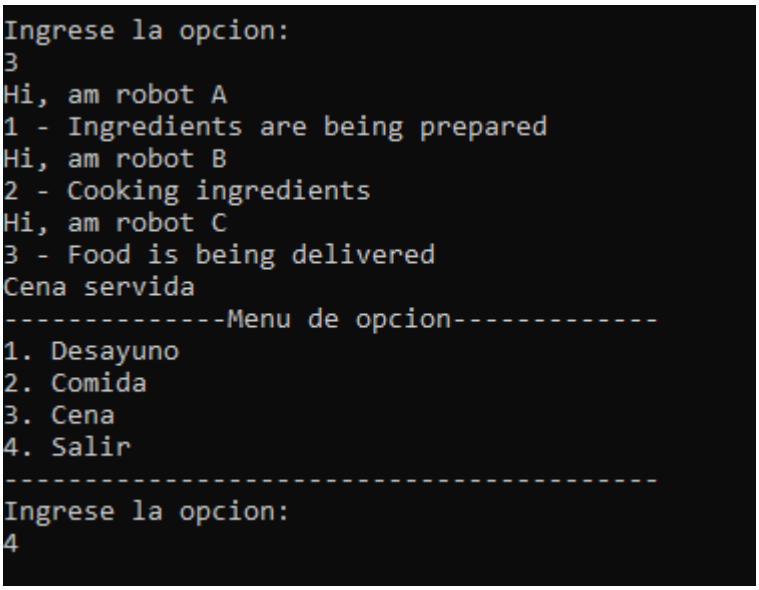
```
C:\Users\Vacintosh\Documents\projects\POO-Claudia-tona\Practica-7>jav
Duster Plymouth 1974 is running
Accelerating
Seatbelts are working fine
Seatbelts locked
Seatbelts are working fine
Seatbelts locked
Lights checked
Lights toggled

C:\Users\Vacintosh\Documents\projects\POO-Claudia-tona\Practica-7>
```

Para agregar una herencia de interfaz se solvento el uso de sus métodos, agregándolos a la superclase.

Práctica 07 – Robot

```
C:\Users\Vacintosh\Documents\projects\POO-Claudia-tona\MagnusOpus>java MagnusOpusMain
-------------Menu de opcion-------------
1. Desayuno
2. Comida
3. Cena
4. Salir
----------------------------------------
Ingrese la opcion:
1
Hi, am robot A
1 - Ingredients are being prepared
Hi, am robot B
2 - Cooking ingredients
Hi, am robot C
3 - Food is being delivered
Desayuno preparado
```

```
-------------Menu de opcion-------------
1. Desayuno
2. Comida
3. Cena
4. Salir
----------------------------------------
Ingrese la opcion:
2
Hi, am robot A
1 - Ingredients are being prepared
Hi, am robot B
2 - Cooking ingredients
Hi, am robot C
3 - Food is being delivered
Comida lista
```



El código de esta práctica se adjuntará a la entrega.