



Universidad Tecnológica de Durango

Tecnologías de la Información

Programación Orientada a Objetos

Prácticas

“Evidencias de Prácticas”

Alumnos:

- Garcia Hernández Ana Gabriela

3°B Bis

Docente:

- Ing. Dagoberto Fiscal Gurrola, M.T.I.

Noviembre 2025

Tabla de Ilustraciones

Ilustración 1 Evidencia del GitHub	3
Ilustración 2 Evidencia de cochesBD.py	4
Ilustración 3 Evidencia del main.py de carros	5
Ilustración 4 Evidencia de nota.py.....	6
Ilustración 5 Evidencia de usuario.py	7
Ilustración 6 Evidencia del main.py de notas	8

Actividad 1

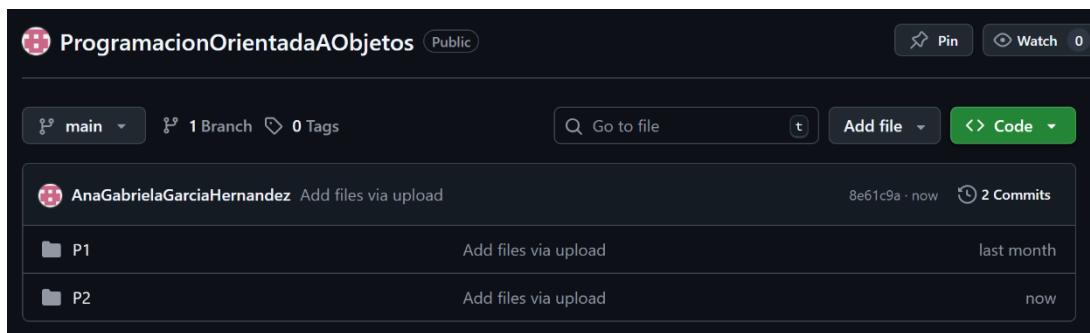
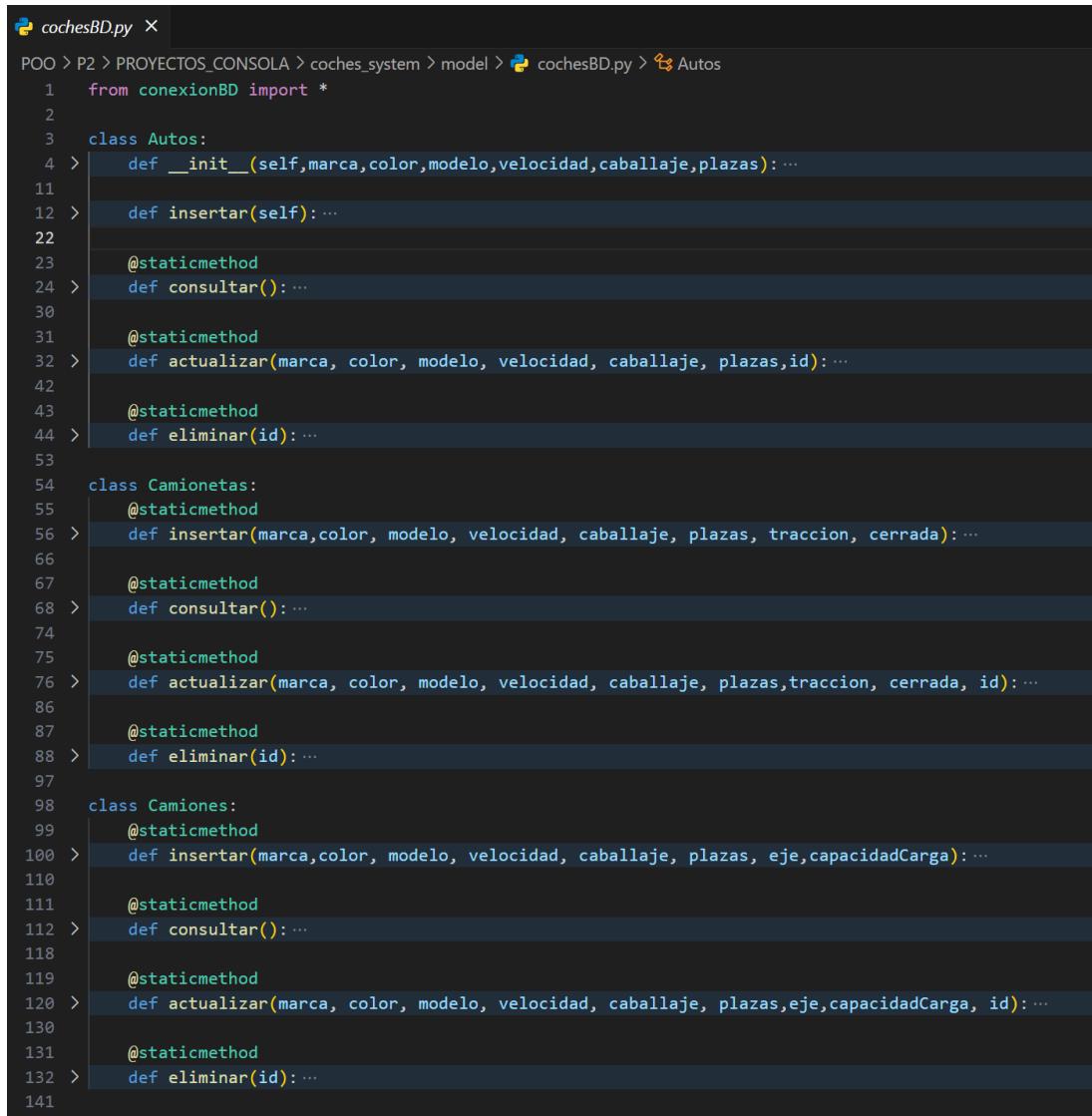


Ilustración 1 Evidencia del GitHub

Da clic: [ENLACE PARA ACCEDER AL GITHUB](#)

En la imagen anterior se muestra la evidencia del repositorio en GitHub, el cual ahora tiene las dos carpetas de parcial uno y dos, donde en el parcial dos se publicaron solo dos proyectos que se realizaron en clase y con el apoyo del profesor de la materia, los cuales fueron notas_system y coches_system.

Actividad 2. cochesBD.py

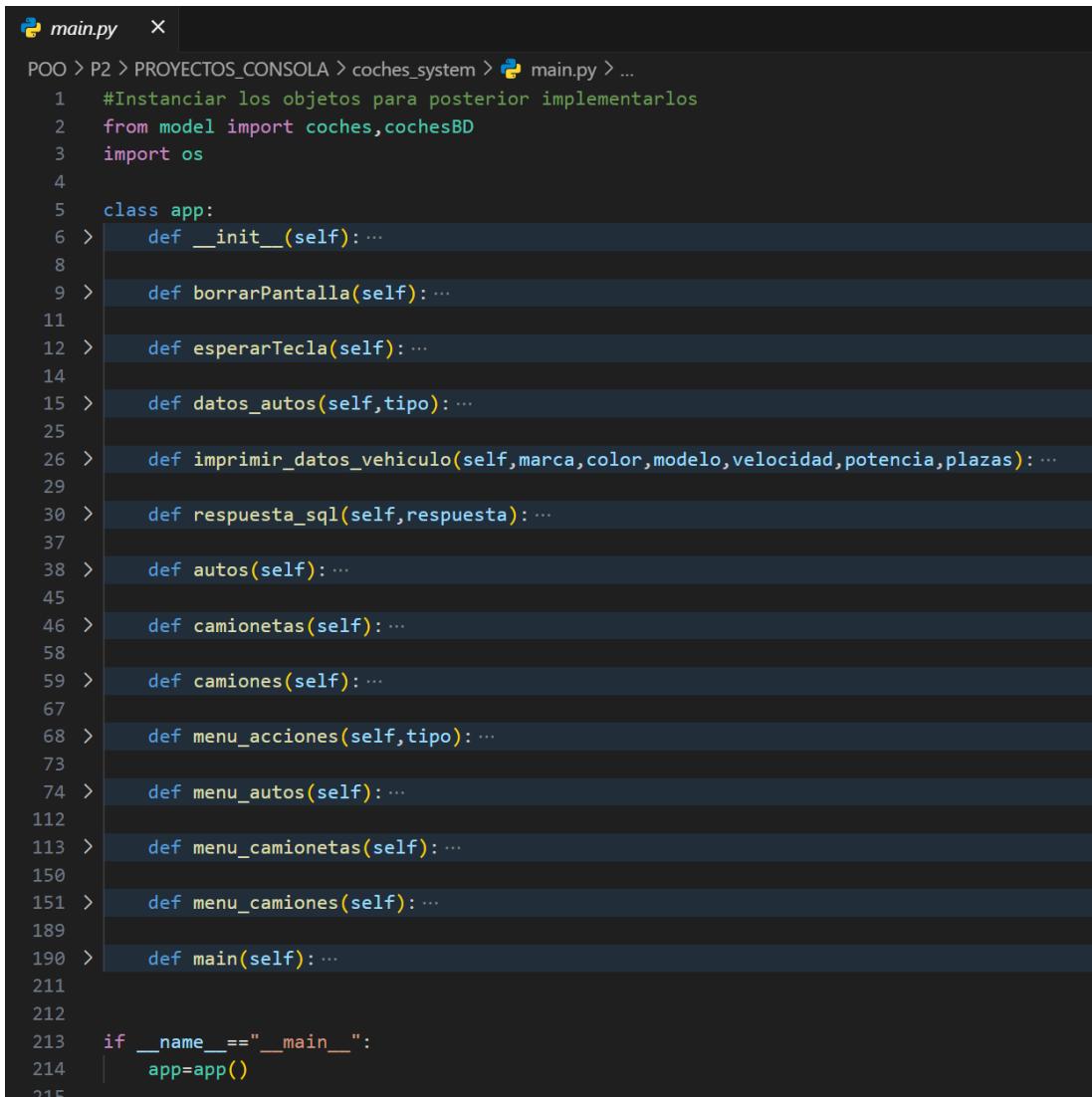


```
POO > P2 > PROYECTOS_CONSOLA > coches_system > model > cochesBD.py > Autos
 1  from conexionBD import *
 2
 3  class Autos:
 4      >     def __init__(self,marca,color,modelo,velocidad,caballaje,plazas): ...
 5
 6      >     def insertar(self): ...
 7
 8      @staticmethod
 9      >     def consultar(): ...
10
11      @staticmethod
12      >     def actualizar(marca, color, modelo, velocidad, caballaje, plazas,id): ...
13
14      @staticmethod
15      >     def eliminar(id): ...
16
17
18  class Camionetas:
19      @staticmethod
20      >     def insertar(marca,color, modelo, velocidad, caballaje, plazas, traccion, cerrada): ...
21
22      @staticmethod
23      >     def consultar(): ...
24
25      @staticmethod
26      >     def actualizar(marca, color, modelo, velocidad, caballaje, plazas,traccion, cerrada, id): ...
27
28      @staticmethod
29      >     def eliminar(id): ...
30
31
32  class Camiones:
33      @staticmethod
34      >     def insertar(marca,color, modelo, velocidad, caballaje, plazas, eje,capacidadCarga): ...
35
36      @staticmethod
37      >     def consultar(): ...
38
39      @staticmethod
40      >     def actualizar(marca, color, modelo, velocidad, caballaje, plazas,eje,capacidadCarga, id): ...
41
42      @staticmethod
43      >     def eliminar(id): ...
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
```

Ilustración 2 Evidencia de cochesBD.py

En la ilustración anterior se muestra el modulo de cochesBD donde se determinan las clases de autos, camionetas y camiones, sus atributos y métodos, los cuales son insertar, consultar, actualizar y eliminar, por cada uno, donde algunos solicitaban los elementos y otros únicamente los id.

Actividad 3. main.py de coches



```

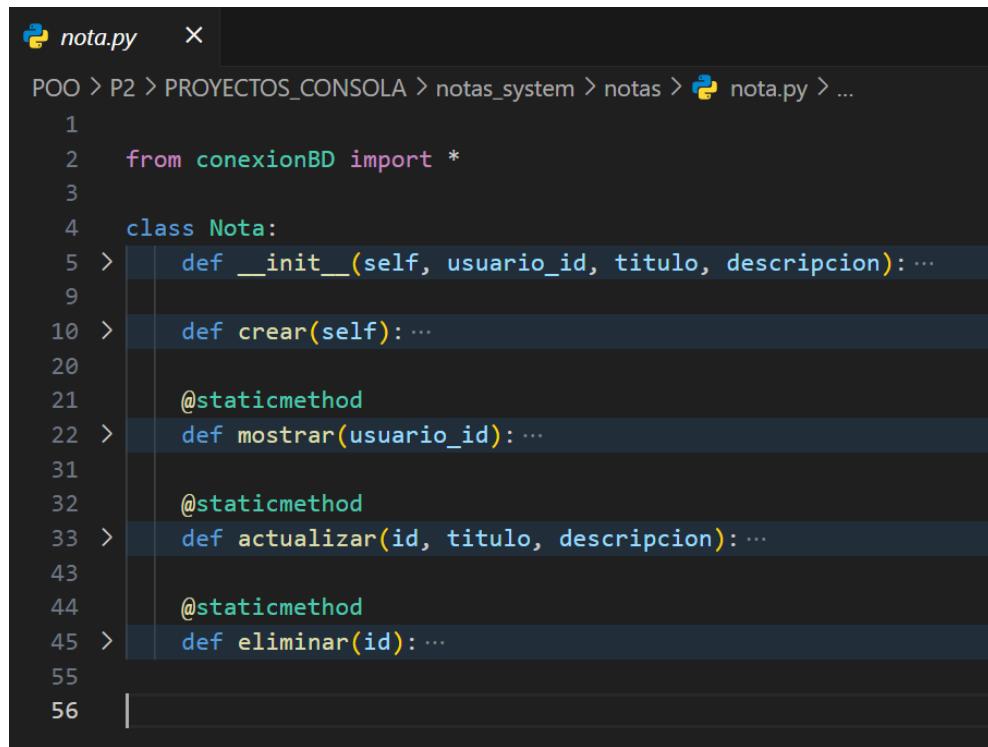
main.py  X
POO > P2 > PROYECTOS_CONSOLA > coches_system > main.py > ...
1   #Instanciar los objetos para posterior implementarlos
2   from model import coches,cochesBD
3   import os
4
5   class app:
6       def __init__(self): ...
8
9       def borrarPantalla(self): ...
11
12      def esperarTecla(self): ...
14
15      def datos_autos(self,tipo): ...
25
26      def imprimir_datos_vehiculo(self,marca,color,modelo,velocidad,potencia,plazas): ...
29
30      def respuesta_sql(self,respuesta): ...
37
38      def autos(self): ...
45
46      def camionetas(self): ...
58
59      def camiones(self): ...
67
68      def menu_acciones(self,tipo): ...
73
74      def menu_autos(self): ...
112
113     def menu_camionetas(self): ...
150
151     def menu_camiones(self): ...
189
190     def main(self): ...
211
212
213     if __name__=="__main__":
214         app=app()
215

```

Ilustración 3 Evidencia del main.py de carros

En la imagen anterior se muestra el main de los carros, donde se muestra la clase aplicación y dentro de ella los métodos para realizar un sistema básico, donde se necesitan los datos de los autos e imprimir lo que se va ingresando para después poder hacer una consulta dependiendo de lo que el usuario desee, como agregar, eliminar, consultar, etc.

Actividad 4. nota.py

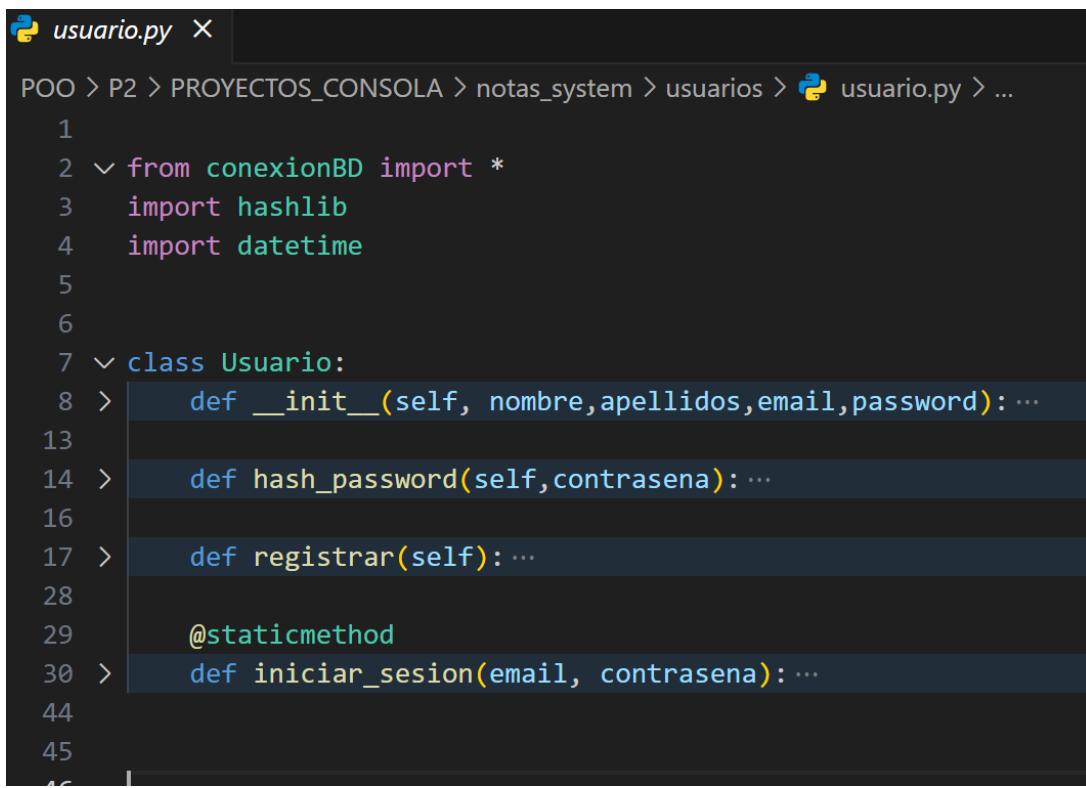


```
nota.py  X
POO > P2 > PROYECTOS_CONSOLA > notas_system > notas > nota.py > ...
1
2     from conexionBD import *
3
4     class Nota:
5     >         def __init__(self, usuario_id, titulo, descripcion): ...
9
10    >        def crear(self): ...
20
21    >        @staticmethod
22    >        def mostrar(usuario_id): ...
31
32    >        @staticmethod
33    >        def actualizar(id, titulo, descripcion): ...
43
44    >        @staticmethod
45    >        def eliminar(id): ...
55
56 |
```

Ilustración 4 Evidencia de nota.py

En la imagen anterior se muestra el modulo de nota.py, donde se creo la clase de nota, la cual muestra sus atributos y los métodos que esta realiza, como lo es crear, mostrar, actualizar y eliminar y como también lo que se necesita para poder llevar a cabo, como el id, el título, descripción o el id del usuario.

Actividad 5. usuario.py

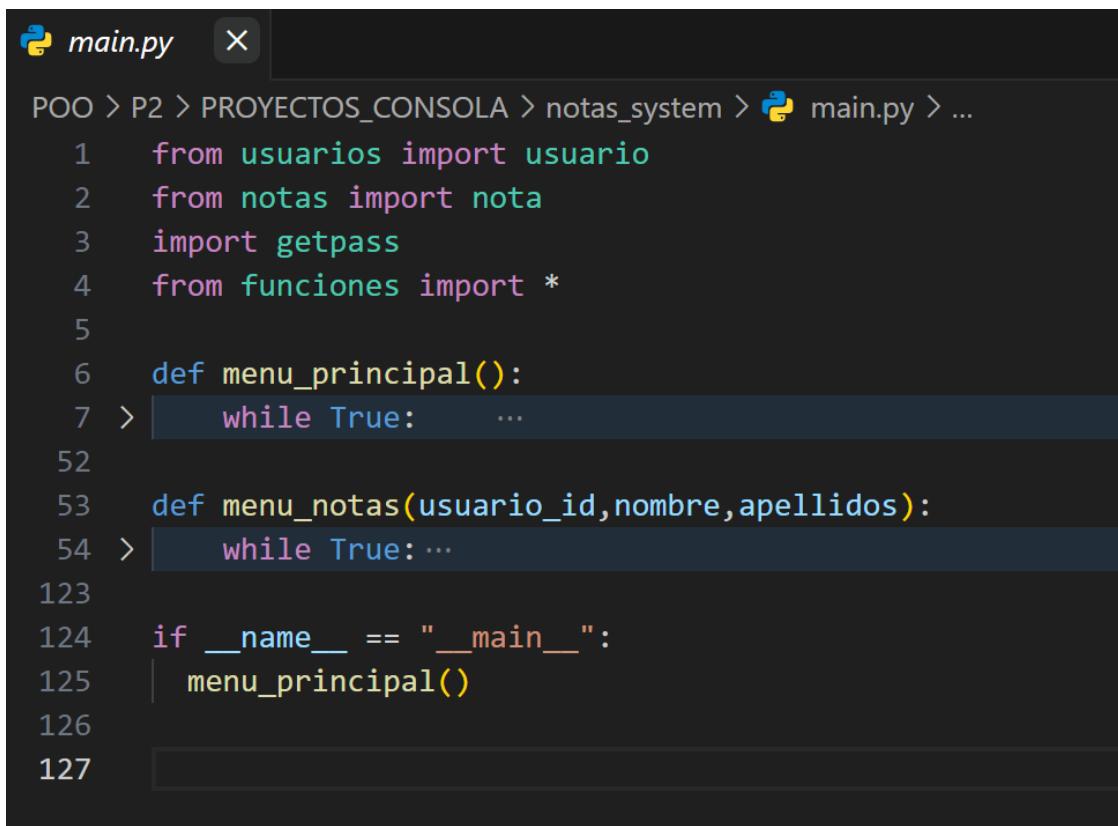


```
usuario.py X
POO > P2 > PROYECTOS_CONSOLA > notas_system > usuarios > usuario.py > ...
1
2  ✓ from conexionBD import *
3    import hashlib
4    import datetime
5
6
7  ✓ class Usuario:
8    >     def __init__(self, nombre, apellidos, email, password): ...
13
14    >     def hash_password(self, contrasena): ...
16
17    >     def registrar(self): ...
28
29        @staticmethod
30    >     def iniciar_sesion(email, contrasena): ...
44
45
46
```

Ilustración 5 Evidencia de usuario.py

En la imagen anterior se muestra el modulo de usuario.py donde se creo la clase de usuario, el cual muestra los atributos de nombre, apellidos, email y contraseña, así como sus métodos los cuales son registrar e iniciar sesión, además de esconder y encriptar las contraseñas que se ingresen por los usuarios.

Actividad 6. main.py de notas



The screenshot shows a code editor window titled "main.py". The code is a Python script for a notes system. It imports modules for users, notes, and functions. It defines two main menus: "menu_principal" which loops until a user exits, and "menu_notas" which also loops. The script ends with a check if it's run directly, calling the main menu. The code is color-coded with syntax highlighting.

```
POO > P2 > PROYECTOS_CONSOLA > notas_system > main.py > ...
1  from usuarios import usuario
2  from notas import nota
3  import getpass
4  from funciones import *
5
6  def menu_principal():
7      while True:
52
53      def menu_notas(usuario_id, nombre, apellidos):
54          while True:
123
124      if __name__ == "__main__":
125          menu_principal()
126
127
```

Ilustración 6 Evidencia del main.py de notas

En la imagen anterior se muestra el main.py de las notas, donde se muestra el menú principal el cual se muestra al inicio del sistema donde se inicia sesión o se crea una cuenta y el menú de notas donde se muestra todo el proceso que se hace para realizar consultas en el sistema de notas.

Retroalimentación

Durante el segundo parcial de la materia de programación orientada a objetos, se realizaron dos proyectos, de los cuales se utilizaron bases de datos y la POO, además no fueron tan complicados, ya que el profesor de la materia había puesto este tipo de proyectos anteriormente durante la clase de programación estructurada, así que incluso el proyecto de notas fue cambiado de estructurado a orientado a objetos.

Gracias a la realización de estos proyectos, se logró implementar de una mejor manera la POO en la vida real y con mejores ejemplos, ya que al ser complejo se logró comprender más a fondo para que sirve cada elemento de la POO y usarlo en su mayor esplendor, como aprovechar de una mejor manera la herencia la cual fue usada en el proyecto de coches.