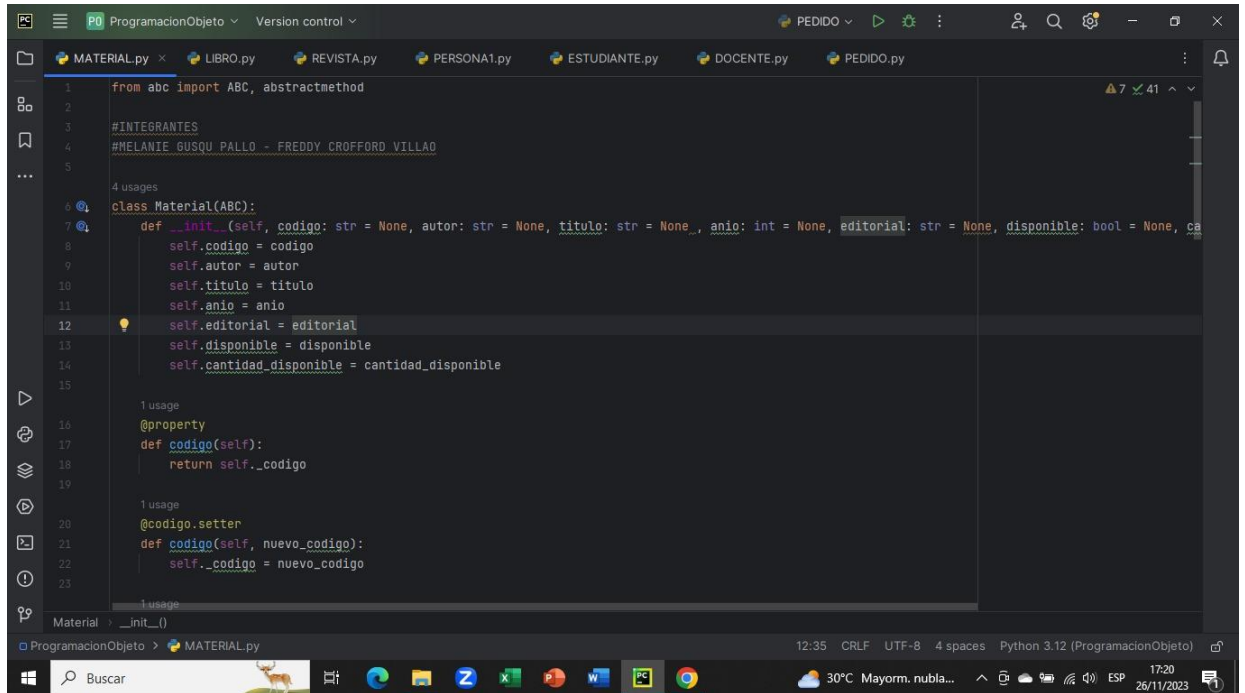


Nombre: Melanie Gusqui Pallo / Freddy Crofford Villao

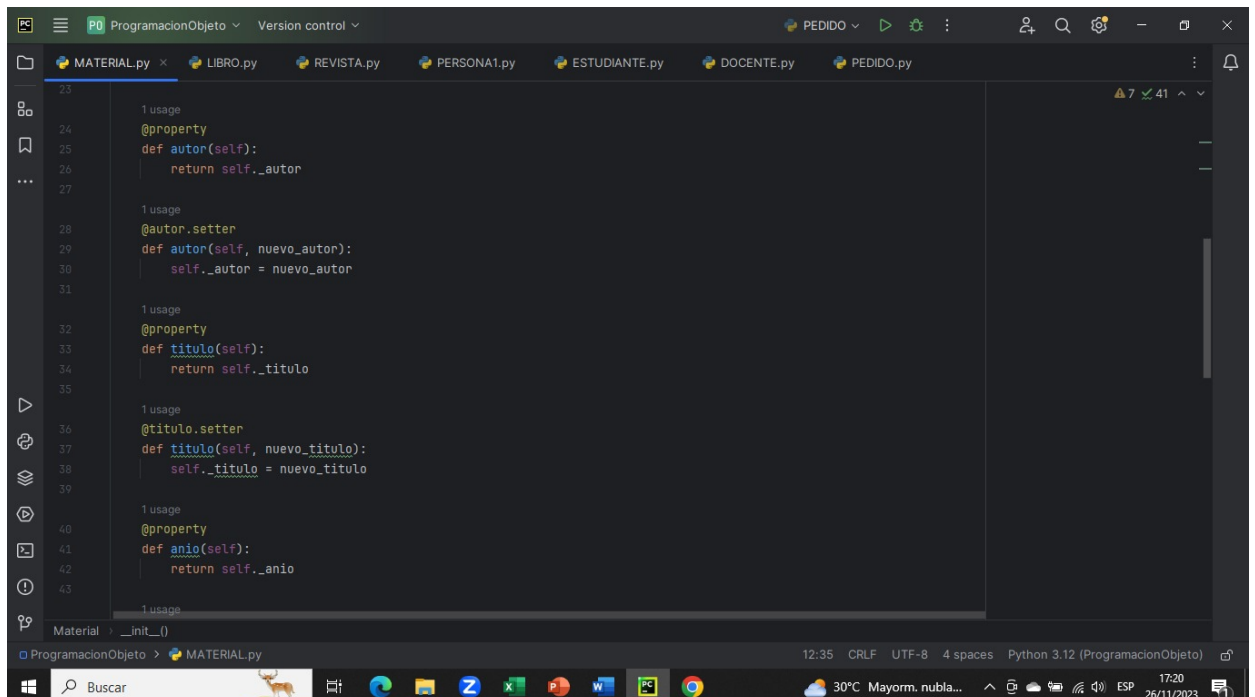
Fecha: 26/11/2023

PROYECTO DE PROGRAMACION ORIENTADA A OBJETOS



The screenshot shows a Python IDE with a dark theme. The file explorer on the left lists several files: MATERIAL.py, LIBRO.py, REVISTA.py, PERSONA1.py, ESTUDIANTE.py, DOCENTE.py, and PEDIDO.py. The main editor window displays the code for 'MATERIAL.py'. The code starts with a docstring in Spanish, followed by the definition of the 'Material' class. The class has an 'ABC' base class and several attributes: 'codigo', 'autor', 'titulo', 'anio', 'editorial', 'disponible', and 'cantidad_disponible'. The '__init__' method initializes these attributes. There are also property decorators for 'codigo' and 'codigo.setter'.

```
1 from abc import ABC, abstractmethod
2
3 #INTEGRANTES
4 #MELANIE GUSQUI PALLO - FREDDY CROFFORD VILLAO
5
6 class Material(ABC):
7     def __init__(self, codigo: str = None, autor: str = None, titulo: str = None, anio: int = None, editorial: str = None, disponible: bool = None, cantidad_disponible: int = None):
8         self.codigo = codigo
9         self.autor = autor
10        self.titulo = titulo
11        self.anio = anio
12        self.editorial = editorial
13        self.disponible = disponible
14        self.cantidad_disponible = cantidad_disponible
15
16    @property
17    def codigo(self):
18        return self._codigo
19
20    @codigo.setter
21    def codigo(self, nuevo_codigo):
22        self._codigo = nuevo_codigo
23
24    1 usage
```



The screenshot shows the same Python IDE as the first image, but the code in 'MATERIAL.py' has been scrolled down. It shows the continuation of the 'Material' class with property decorators for 'autor', 'titulo', and 'anio'. The 'autor' property has a getter and a setter. The 'titulo' property has a getter and a setter. The 'anio' property has a getter.

```
23
24    1 usage
25    @property
26    def autor(self):
27        return self._autor
28
29    1 usage
30    @autor.setter
31    def autor(self, nuevo_autor):
32        self._autor = nuevo_autor
33
34    1 usage
35    @property
36    def titulo(self):
37        return self._titulo
38
39    1 usage
40    @titulo.setter
41    def titulo(self, nuevo_titulo):
42        self._titulo = nuevo_titulo
43
44    1 usage
45    @property
46    def anio(self):
47        return self._anio
48
49    1 usage
```

```
ProgramacionObjeto Version control
PEDIDO
MATERIAL.py LIBRO.py REVISTA.py PERSONA1.py ESTUDIANTE.py DOCENTE.py PEDIDO.py
1 usage
44 @anio.setter
45 def anio(self, nuevo_anio):
46     self._anio = nuevo_anio
47
1 usage
48 @property
49 def editorial(self):
50     return self._editorial
51
1 usage
52 @editorial.setter
53 def editorial(self, nuevo_editorial):
54     self._editorial = nuevo_editorial
55
1 usage
56 @property
57 def disponible(self):
58     return self._disponible
59
1 usage
60 @disponible.setter
61 def disponible(self, nuevo_estado):
62     self._disponible = nuevo_estado
63
1 usage
64 @property
65 def cantidad_disponible(self):
66     return self._cantidad_disponible
67
1 usage
68 @cantidad_disponible.setter
69 def cantidad_disponible(self, nueva_cantidad):
70     self._cantidad_disponible = nueva_cantidad
71
72
73 @abstractmethod
74 def actualizar_disponibilidad(self, disponible):
75     pass
76
77
Material > _init_()
```

```
ProgramacionObjeto Version control
PEDIDO
MATERIAL.py LIBRO.py REVISTA.py PERSONA1.py ESTUDIANTE.py DOCENTE.py PEDIDO.py
1 usage
60 @disponible.setter
61 def disponible(self, nuevo_estado):
62     self._disponible = nuevo_estado
63
1 usage
64 @property
65 def cantidad_disponible(self):
66     return self._cantidad_disponible
67
1 usage
68 @cantidad_disponible.setter
69 def cantidad_disponible(self, nueva_cantidad):
70     self._cantidad_disponible = nueva_cantidad
71
72
73 @abstractmethod
74 def actualizar_disponibilidad(self, disponible):
75     pass
76
77
Material > _init_()
```

LIBRO

```
from MATERIAL import Material

#INTEGRANTES
#MELANIE GUSQU PALLO - FREDDY CROFFORD VILLAO

5 usages
class Libro(Material):
    _contador_libro = 0

    def __init__(self, codigo: str = None, autor: str = None, titulo: str = None, anio: int = None, editorial: str = None, disponible: bool = None, cantidad_disponible: int = None):
        super().__init__(codigo, autor, titulo, anio, editorial, disponible, cantidad_disponible)
        self.id = id
        self._tipo_pasta = tipo_pasta
        Libro._contador_libro += 1

    1 usage
    @property
    def contador_libro(self):
        return Libro._contador_libro

    2 usages
    @property
    def id(self):
        return self._id

    @id.setter
    def id(self, nuevo_id):
```

```
@id.setter
def id(self, nuevo_id):
    self._id = nuevo_id

2 usages
@property
def tipo_pasta(self):
    return self._tipo_pasta

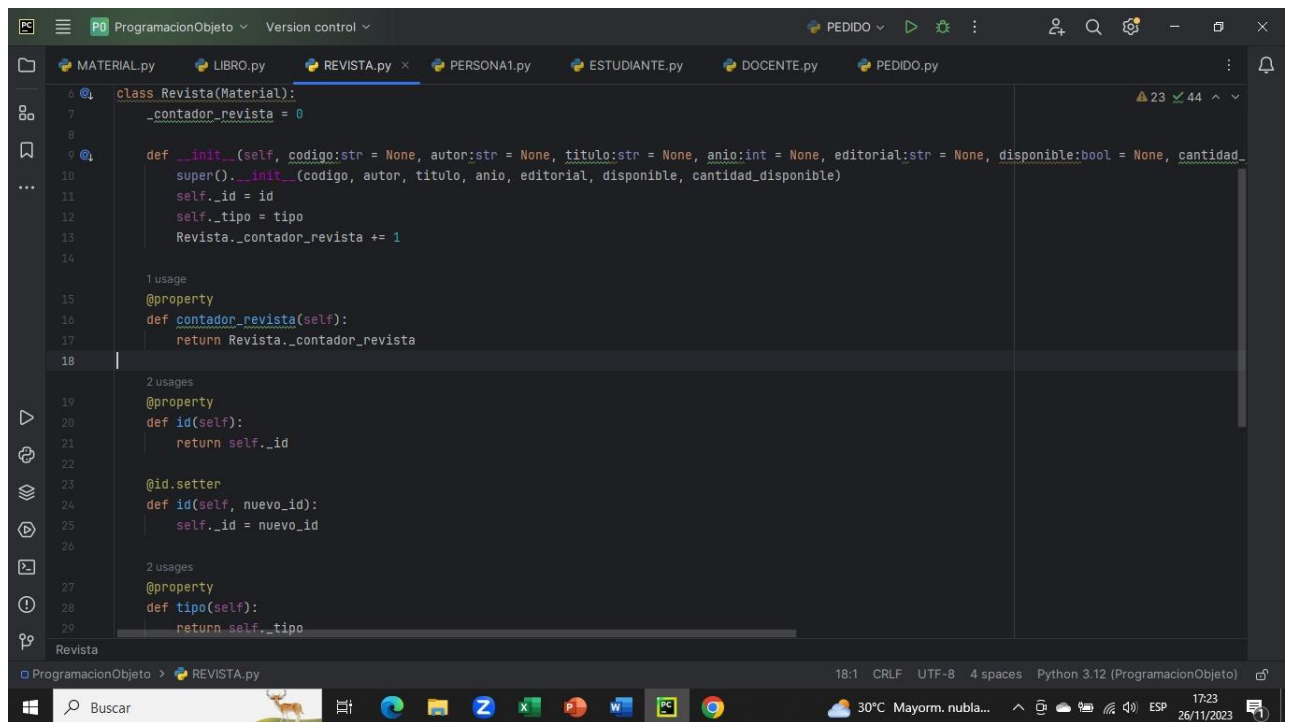
@tipo_pasta.setter
def tipo_pasta(self, nueva_tipo_pasta):
    self._tipo_pasta = nueva_tipo_pasta

def actualizar_disponibilidad(self, disponible):
    self.disponible = disponible

def __str__(self):
    return f'Libro(id={self.id}, codigo={self.codigo}, titulo={self.titulo}, autor={self.autor}, disponible={self.disponible}, cantidad_disponible={self.cantidad_disponible})'

if __name__ == '__main__':
    #libro1 = Libro(id=10, codigo='CP73L', titulo='La Sombra del Viento', autor='Carlos Ruiz Zafón', disponible=True, cantidad_disponible=7, tipo_pasta='Pasta dura')
    #print(libro1)
    #libro2 = Libro(id=102, codigo='D13PS', titulo='Asesino de Brujas', autor='Shelby Mahurin', disponible=True, cantidad_disponible=19, tipo_pasta='Pasta blanda')
    #print(libro2)
```

REVISTA



```
class Revista(Material):
    _contador_revista = 0

    def __init__(self, codigo:str = None, autor:str = None, titulo:str = None, anio:int = None, editorial:str = None, disponible:bool = None, cantidad_disponible:int = None):
        super().__init__(codigo, autor, titulo, anio, editorial, disponible, cantidad_disponible)
        self._id = id
        self._tipo = tipo
        Revista._contador_revista += 1

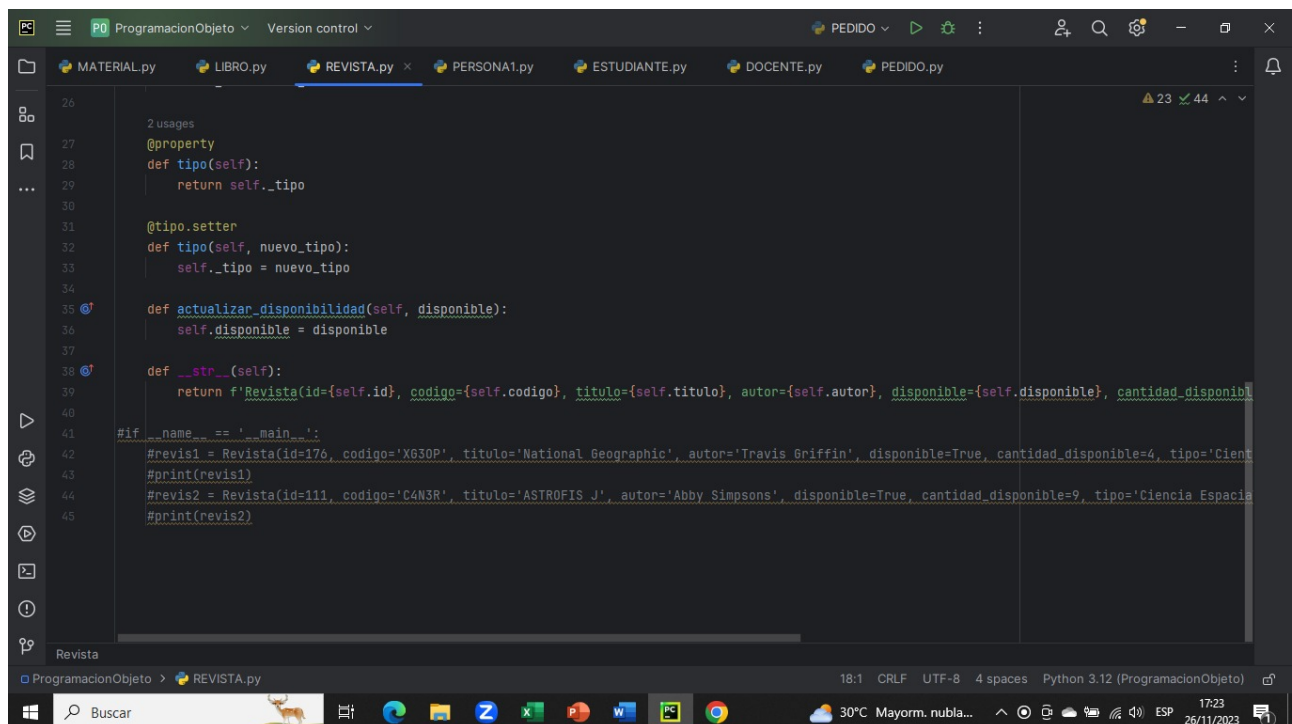
    @property
    def contador_revista(self):
        return Revista._contador_revista

    @property
    def id(self):
        return self._id

    @id.setter
    def id(self, nuevo_id):
        self._id = nuevo_id

    @property
    def tipo(self):
        return self._tipo
```

ProgramacionObjeto > REVISTA.py 18:1 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto) 17:23 26/11/2023



```
    @property
    def tipo(self):
        return self._tipo

    @tipo.setter
    def tipo(self, nuevo_tipo):
        self._tipo = nuevo_tipo

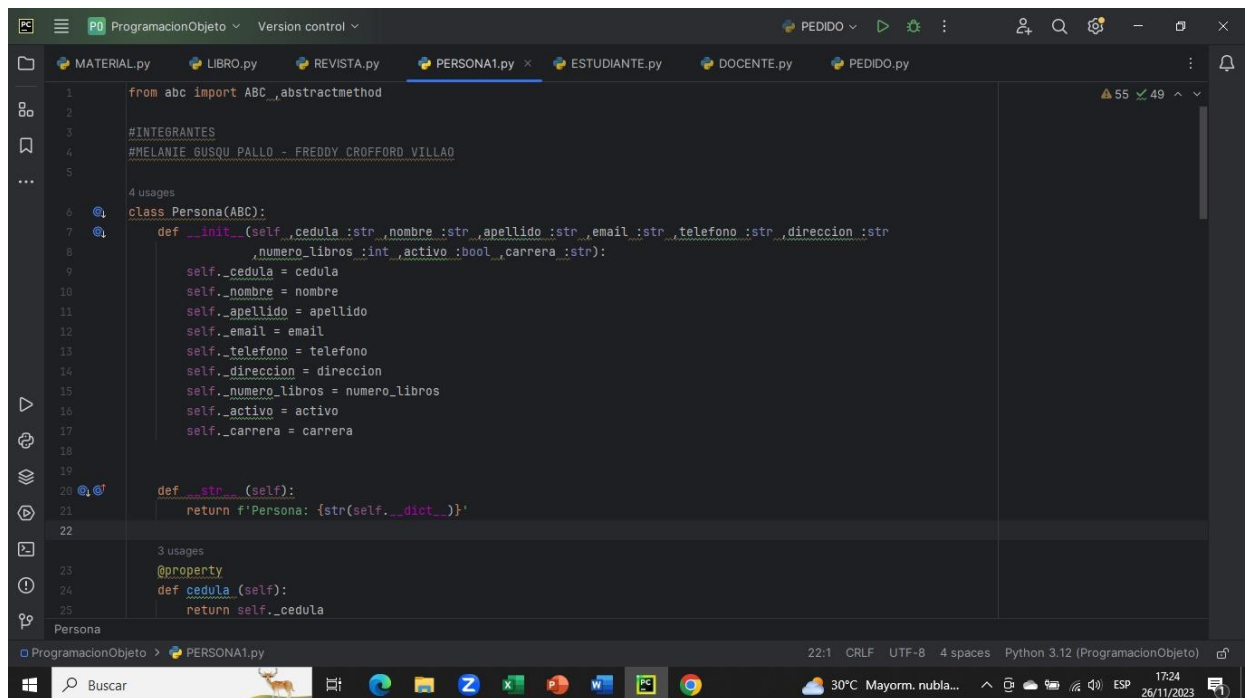
    def actualizar_disponibilidad(self, disponible):
        self.disponible = disponible

    def __str__(self):
        return f'Revista(id={self.id}, codigo={self.codigo}, titulo={self.titulo}, autor={self.autor}, disponible={self.disponible}, cantidad_disponible={self.cantidad_disponible})'

if __name__ == '__main__':
    #revis1 = Revista(id=176, codigo='X630P', titulo='National Geographic', autor='Travis Griffin', disponible=True, cantidad_disponible=4, tipo='Ciencia Espacial')
    #print(revis1)
    #revis2 = Revista(id=111, codigo='C4N3R', titulo='ASTROFIS J', autor='Abby Simpsons', disponible=True, cantidad_disponible=9, tipo='Ciencia Espacial')
    #print(revis2)
```

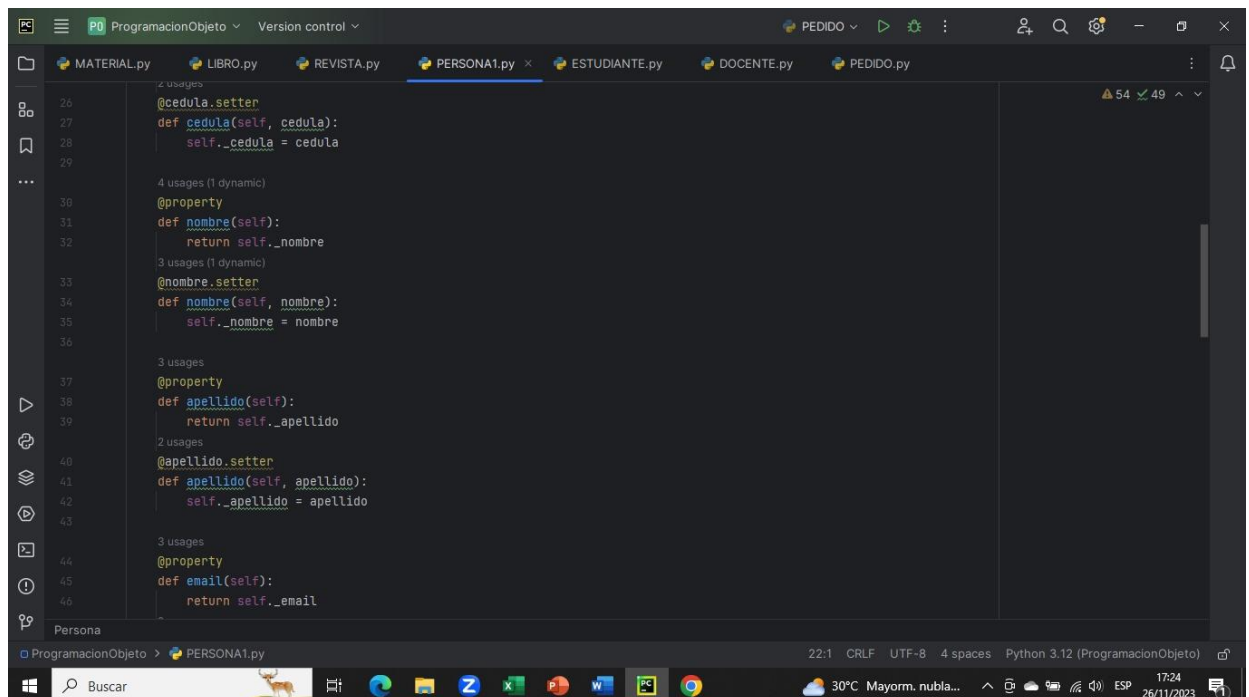
ProgramacionObjeto > REVISTA.py 18:1 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto) 17:23 26/11/2023

PERSONA



The screenshot shows a code editor with the file 'PERSONA1.py' open. The code defines a class 'Persona' with an 'ABC' abstract method. The class has an 'init' method that initializes attributes: 'cedula', 'nombre', 'apellido', 'email', 'telefono', 'direccion', 'numero_libros', 'activo', and 'carrera'. It also has a 'str' method that returns a string representation of the object. The editor shows line numbers 1 through 25. The status bar at the bottom indicates '22:1 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)'.

```
1 from abc import ABC, abstractmethod
2
3 #INTEGRANTES
4 #MELANIE GUSQU PALLO - FREDDY CROFFORD VILLAO
5
6 4 usages
7 class Persona(ABC):
8     def __init__(self, cedula: str, nombre: str, apellido: str, email: str, telefono: str, direccion: str,
9         numero_libros: int, activo: bool, carrera: str):
10         self._cedula = cedula
11         self._nombre = nombre
12         self._apellido = apellido
13         self._email = email
14         self._telefono = telefono
15         self._direccion = direccion
16         self._numero_libros = numero_libros
17         self._activo = activo
18         self._carrera = carrera
19
20 2 usages
21     def __str__(self):
22         return f'Persona: {str(self.__dict__)}'
23
24 3 usages
25     @property
26     def cedula(self):
27         return self._cedula
```



The screenshot shows the same code editor with the file 'PERSONA1.py' open, displaying the continuation of the 'Persona' class. It includes setter methods for 'cedula', 'nombre', 'apellido', and 'email'. The editor shows line numbers 26 through 46. The status bar at the bottom indicates '22:1 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)'.

```
26 2 usages
27     @cedula.setter
28     def cedula(self, cedula):
29         self._cedula = cedula
30
31 4 usages (1 dynamic)
32     @property
33     def nombre(self):
34         return self._nombre
35
36 3 usages (1 dynamic)
37     @nombre.setter
38     def nombre(self, nombre):
39         self._nombre = nombre
40
41 3 usages
42     @property
43     def apellido(self):
44         return self._apellido
45
46 2 usages
47     @apellido.setter
48     def apellido(self, apellido):
49         self._apellido = apellido
50
51 3 usages
52     @property
53     def email(self):
54         return self._email
```

The screenshot shows a code editor with the file `PERSONA1.py` open. The code defines the `Persona` class with the following attributes and methods:

```
47 @email.setter
48 def email(self, email):
49     self._email = email
50
51 3 usages
52 @property
53 def telefono(self):
54     return self._telefono
55
56 2 usages
57 @telefono.setter
58 def telefono(self, telefono):
59     self._telefono = telefono
60
61 3 usages
62 @property
63 def direccion(self):
64     return self._direccion
65
66 2 usages
67 @direccion.setter
68 def direccion(self, direccion):
69     self._direccion = direccion
70
71 3 usages
72 @property
73 def numero_libros(self):
74     return self._numero_libros
75
76 2 usages
```

The status bar at the bottom indicates the file is `PERSONA1.py` in the `ProgramacionObjeto` directory, using Python 3.12 with CRLF line endings, UTF-8 encoding, and 4 spaces for indentation. The system tray shows the date and time as 26/11/2023 at 17:25.

The screenshot shows the same code editor with the `PERSONA1.py` file. The code continues with the implementation of the `numero_libros`, `activo`, `carrera`, and `pedir_libro` attributes and methods:

```
68 2 usages
69 @numero_libros.setter
70 def numero_libros(self, numero_libros):
71     self._numero_libros = numero_libros
72
73 3 usages
74 @property
75 def activo(self):
76     return self._activo
77
78 2 usages
79 @activo.setter
80 def activo(self, activo):
81     self._activo = activo
82
83 3 usages
84 @property
85 def carrera(self):
86     return self._carrera
87
88 2 usages
89 @carrera.setter
90 def carrera(self, carrera):
91     self._carrera = carrera
92
93 3 usages
94 @abstractmethod
95 def pedir_libro(self) -> bool:
96     pass
97
98 0 usages
```

The status bar at the bottom remains the same, showing the file `PERSONA1.py` in the `ProgramacionObjeto` directory, using Python 3.12 with CRLF line endings, UTF-8 encoding, and 4 spaces for indentation. The system tray shows the date and time as 26/11/2023 at 17:25.

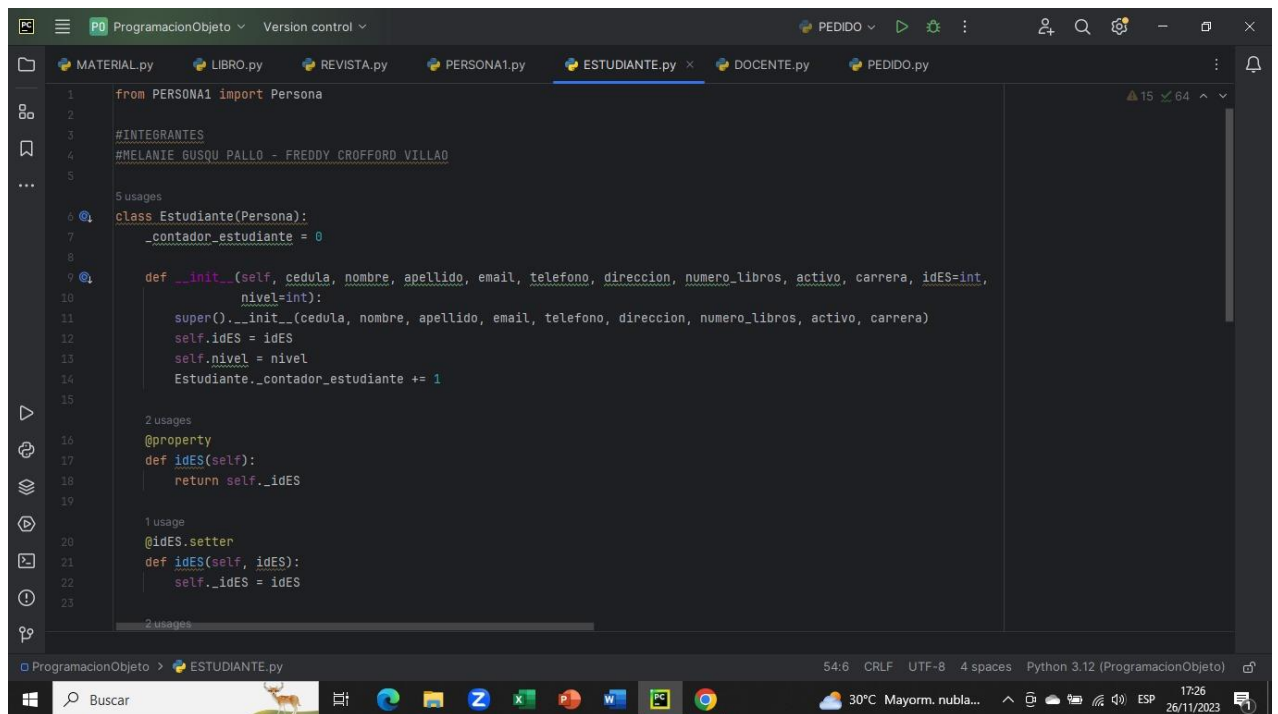

```
77     self._activo = activo
78
79     3 usages
80     @property
81     def carrera(self):
82         return self._carrera
83
84     2 usages
85     @carrera.setter
86     def carrera(self, carrera):
87         self._carrera = carrera
88
89
90     @abstractmethod
91     def pedir_libro(self) -> bool:
92         pass
93
94
95     @abstractmethod
96     def devolver_libro(self) -> bool:
97         pass
98
99
100 if __name__ == '__main__':
101     pass
```

Persona

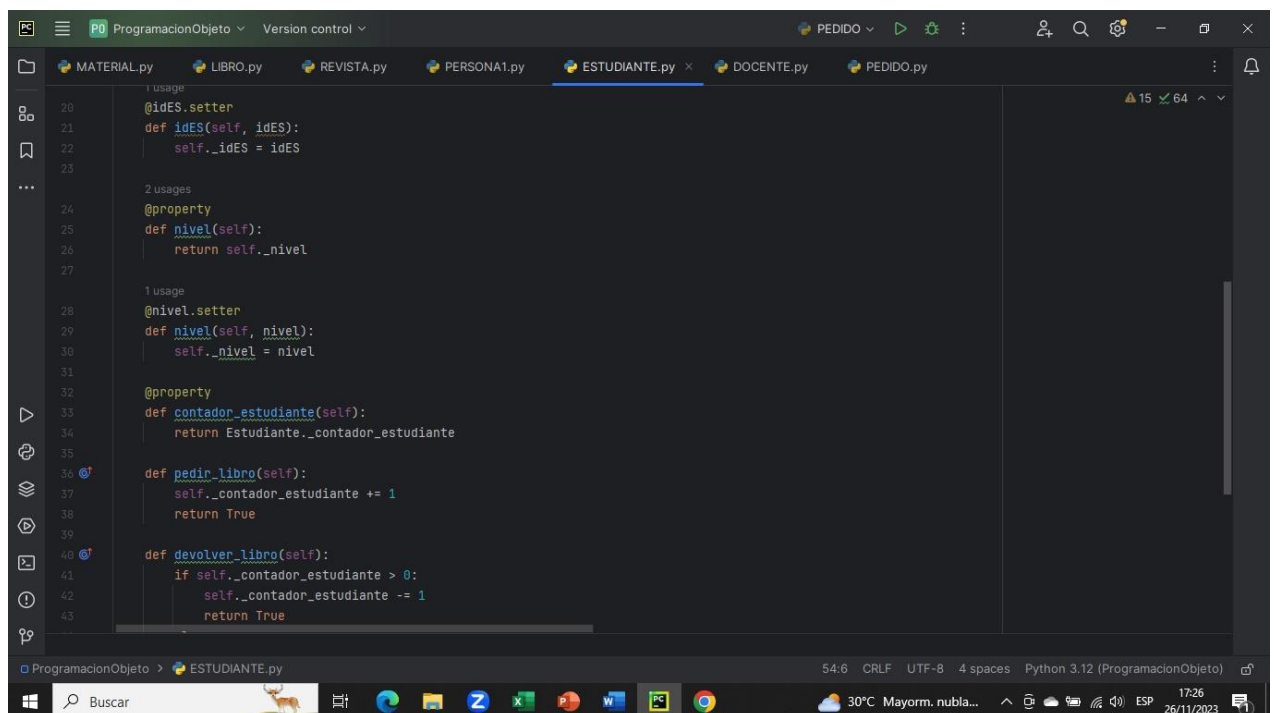
ProgramacionObjeto > PERSONA1.py 22:1 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)

Buscar 30°C Mayorm. nubla... 17:25 26/11/2023

ESTUDIANTE



```
1 from PERSONA1 import Persona
2
3 #INTEGRANTES
4 #MELANIE GUSQU PALLO - FREDDY CROFFORD VILLAO
5
6 5 usages
7 class Estudiante(Persona):
8     _contador_estudiante = 0
9
10     def __init__(self, cedula, nombre, apellido, email, telefono, direccion, numero_libros, activo, carrera, idES=int, nivel=int):
11         super().__init__(cedula, nombre, apellido, email, telefono, direccion, numero_libros, activo, carrera)
12         self.idES = idES
13         self.nivel = nivel
14         Estudiante._contador_estudiante += 1
15
16     2 usages
17     @property
18     def idES(self):
19         return self._idES
20
21     1 usage
22     @idES.setter
23     def idES(self, idES):
24         self._idES = idES
25
26     2 usages
```



```
20     1 usage
21     @idES.setter
22     def idES(self, idES):
23         self._idES = idES
24
25     2 usages
26     @property
27     def nivel(self):
28         return self._nivel
29
30     1 usage
31     @nivel.setter
32     def nivel(self, nivel):
33         self._nivel = nivel
34
35     @property
36     def contador_estudiante(self):
37         return Estudiante._contador_estudiante
38
39     36
40     def pedir_libro(self):
41         self._contador_estudiante += 1
42         return True
43
44     def devolver_libro(self):
45         if self._contador_estudiante > 0:
46             self._contador_estudiante -= 1
47             return True
```



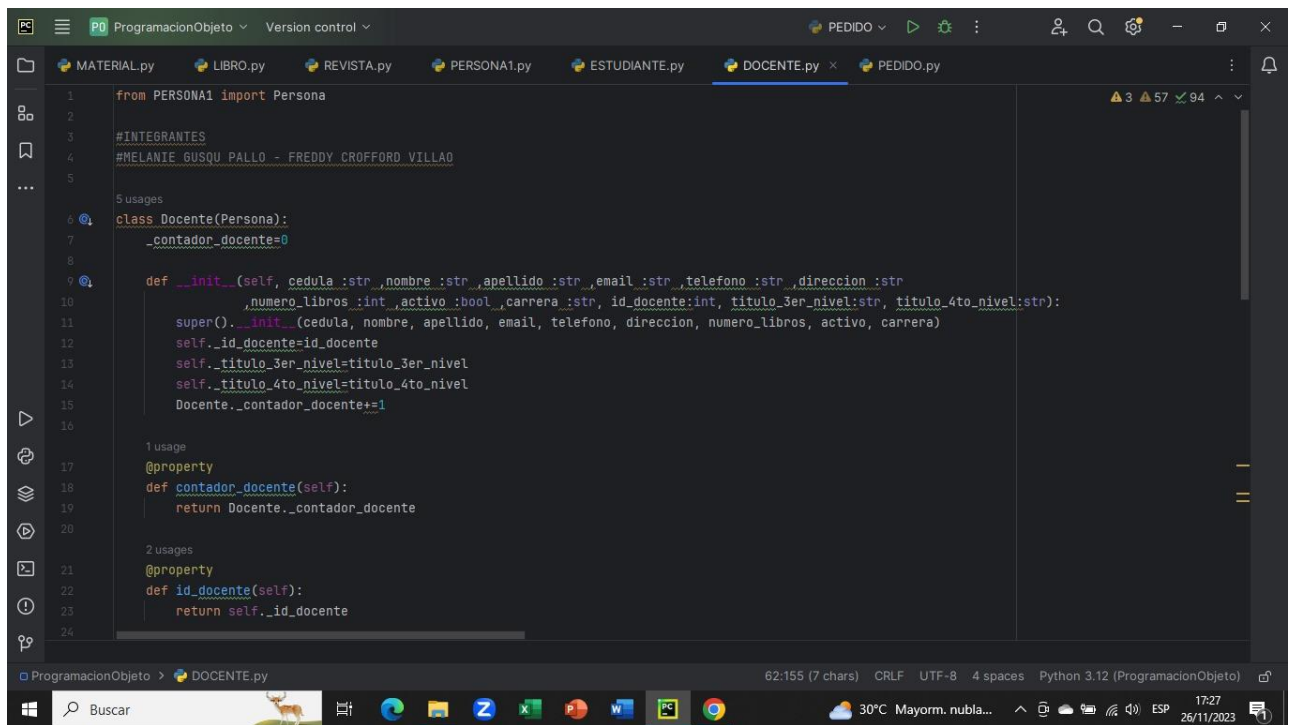
```
ProgramacionObjeto  Version control  PEDIDO  ESTUDIANTE.py  DOCENTE.py  PEDIDO.py

35
36 def pedir_libro(self):
37     self._contador_estudiante += 1
38     return True
39
40 def devolver_libro(self):
41     if self._contador_estudiante > 0:
42         self._contador_estudiante -= 1
43         return True
44     else:
45         return False
46
47 def __str__(self):
48     return f"Estudiante(cedula={self.cedula}, nombre={self.nombre}, apellido={self.apellido}, email={self.email}, telefono={self.telefono}, direccion={self.direccion})"
49
50 #if __name__ == "__main__":
51 #    estudiante1 = Estudiante(cedula="0931054233", nombre="Melanie", apellido="Gusqui", email="melaniegusqui@gmail.com", telefono="0997079658", direccion="Calle 123")
52 #    print(estudiante1)
53 #    estudiante1 = Estudiante(cedula="0957808181", nombre="Freddy", apellido="Crofford", email="crovillpapo@gmail.com", telefono="0978759219", direccion="Calle 456")
54 #    print(estudiante1)
55
```

ProgramacionObjeto > ESTUDIANTE.py 54:6 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)

Buscar 30°C Mayorm. nubla... 17:26 26/11/2023

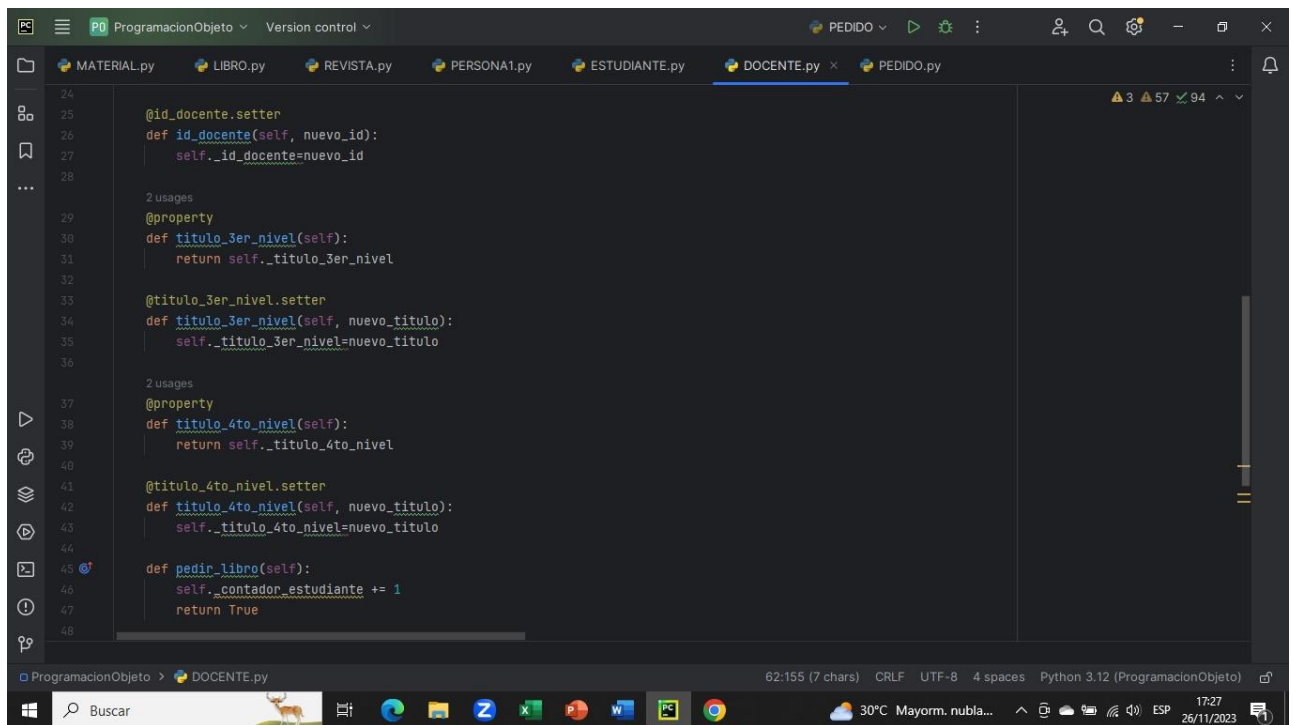
DOCENTE



The screenshot shows a code editor with the following Python code for the `Docente` class:

```
1 from PERSONA1 import Persona
2
3 #INTEGRANTES
4 #MELANIE GUSQU PALLO - FREDDY CROFFORD VILLAO
5
6 class Docente(Persona):
7     _contador_docente=0
8
9     def __init__(self, cedula :str, nombre :str, apellido :str, email :str, telefono :str, direccion :str,
10                 numero_libros :int, activo :bool, carrera :str, id_docente:int, titulo_3er_nivel:str, titulo_4to_nivel:str):
11         super().__init__(cedula, nombre, apellido, email, telefono, direccion, numero_libros, activo, carrera)
12         self._id_docente=id_docente
13         self._titulo_3er_nivel=titulo_3er_nivel
14         self._titulo_4to_nivel=titulo_4to_nivel
15         Docente._contador_docente+=1
16
17     @property
18     def contador_docente(self):
19         return Docente._contador_docente
20
21     @property
22     def id_docente(self):
23         return self._id_docente
24
```

The editor interface includes a sidebar with icons for Explorer, Search, and Run and Debug. The bottom status bar shows the file path `ProgramacionObjeto > DOCENTE.py`, character count `62:155 (7 chars)`, encoding `CRLF`, `UTF-8`, `4 spaces`, and Python version `Python 3.12 (ProgramacionObjeto)`. The Windows taskbar at the bottom shows the time `17:27` on `26/11/2023`.



The screenshot shows the continuation of the `Docente` class implementation in the code editor:

```
24
25 @id_docente.setter
26 def id_docente(self, nuevo_id):
27     self._id_docente=nuevo_id
28
29 @property
30 def titulo_3er_nivel(self):
31     return self._titulo_3er_nivel
32
33 @titulo_3er_nivel.setter
34 def titulo_3er_nivel(self, nuevo_titulo):
35     self._titulo_3er_nivel=nuevo_titulo
36
37 @property
38 def titulo_4to_nivel(self):
39     return self._titulo_4to_nivel
40
41 @titulo_4to_nivel.setter
42 def titulo_4to_nivel(self, nuevo_titulo):
43     self._titulo_4to_nivel=nuevo_titulo
44
45 def pedir_libro(self):
46     self._contador_estudiante += 1
47     return True
48
```

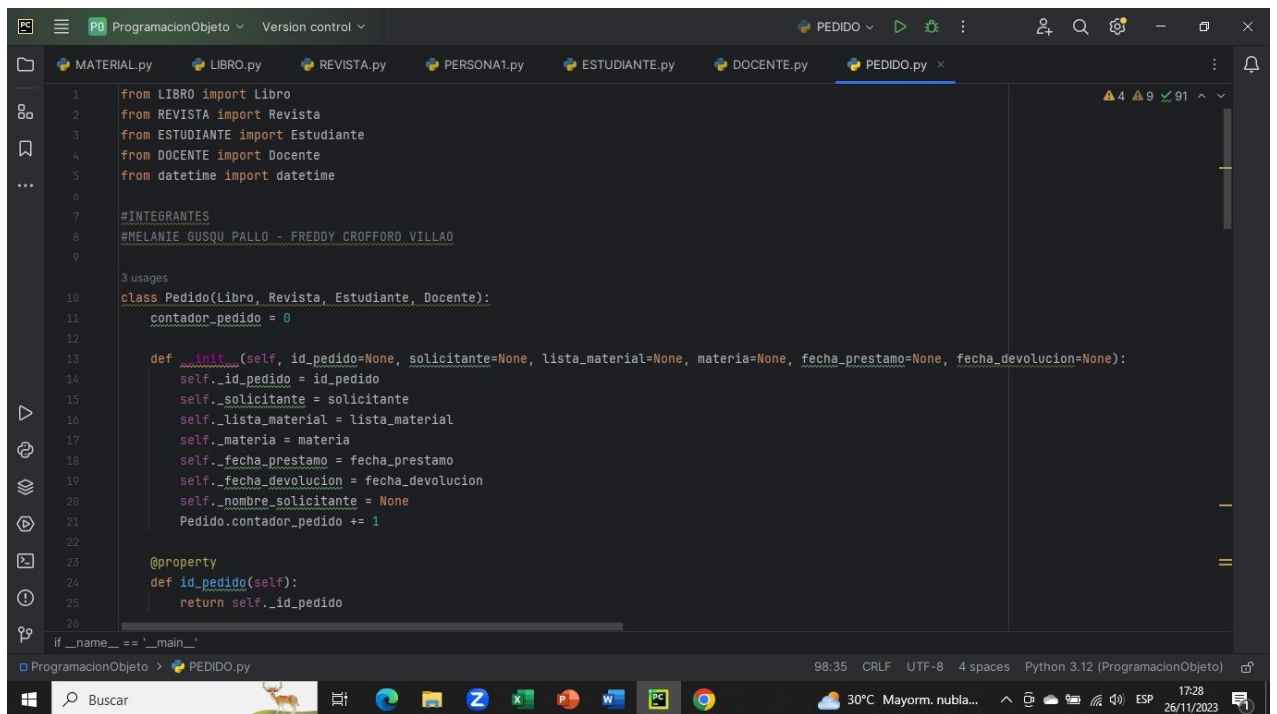
The editor interface is consistent with the previous screenshot, showing the same sidebar and status bar information.

```
PC  P0 ProgramacionObjeto  Version control  PEDIDO  PEDIDO.py  3 57 94  ^ v
MATERIAL.py  LIBRO.py  REVISTA.py  PERSONA1.py  ESTUDIANTE.py  DOCENTE.py  PEDIDO.py
44
45 def pedir_libro(self):
46     self._contador_estudiante += 1
47     return True
48
49 def devolver_libro(self):
50     if self._contador_estudiante > 0:
51         self._contador_estudiante -= 1
52         return True
53     else:
54         return False
55
56 def __str__(self):
57     return f'Docente(cedula={self.cedula}, nombre={self.nombre}, apellido={self.apellido}, email={self.email}, telefono={self.telefono}, direccion={self.direccion})'
58
59 #if __name__ == '__main__':
60 #docen1 = Docente(cedula="0690024187", nombre="Javier", apellido="Macias", email="javimacias@gmail.com", telefono="0995321105", direccion="Alborada")
61 #print(docen1)
62 #docen2 = Docente(cedula="0974321421", nombre="Mercedes", apellido="Mendoza", email="mercemend@hotmail.com", telefono="0987235117", direccion="Acac")
63 #print(docen2)
64
```

ProgramacionObjeto > DOCENTE.py 64:1 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)

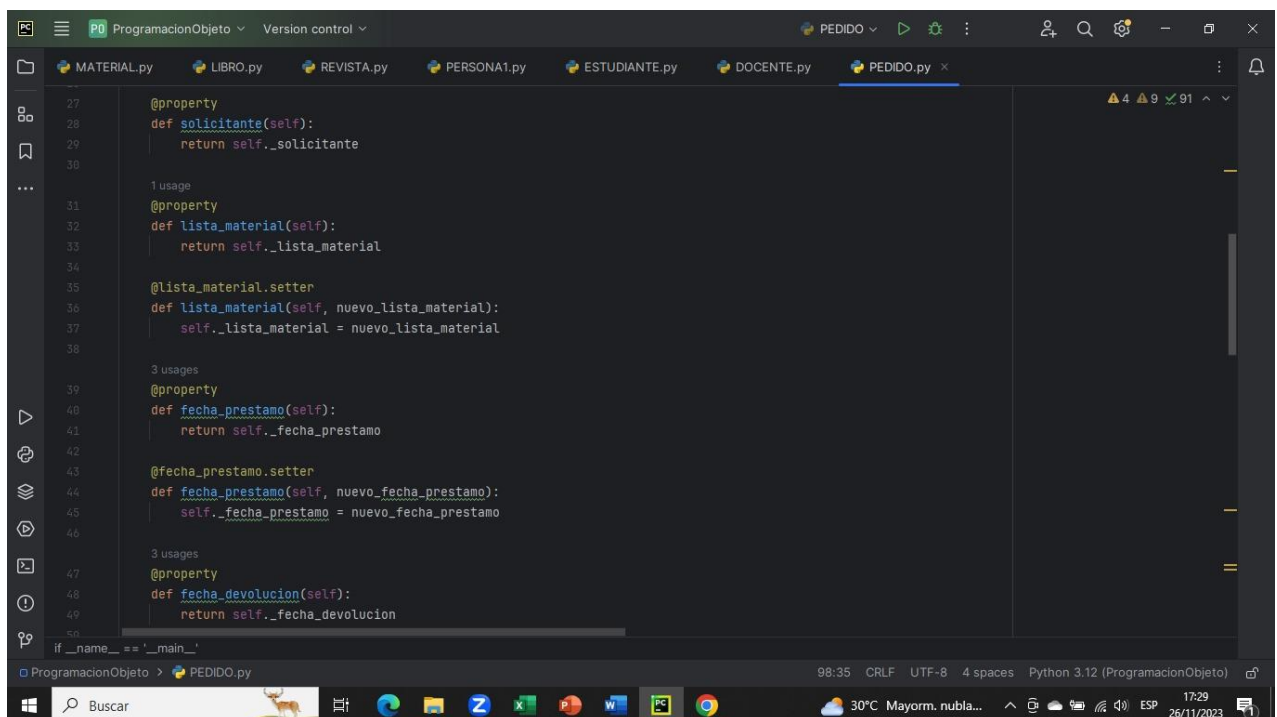
Buscar 30°C Mayorm. nubla... 17:28 26/11/2023

PEDIDO



```
1 from LIBRO import Libro
2 from REVISTA import Revista
3 from ESTUDIANTE import Estudiante
4 from DOCENTE import Docente
5 from datetime import datetime
6
7 #INTEGRANTES
8 #MELANIE GUSQU PALLO - FREDDY CROFFORD VILLAO
9
10 3 usages
11 class Pedido(Libro, Revista, Estudiante, Docente):
12     contador_pedido = 0
13
14     def __init__(self, id_pedido=None, solicitante=None, lista_material=None, materia=None, fecha_prestamo=None, fecha_devolucion=None):
15         self._id_pedido = id_pedido
16         self._solicitante = solicitante
17         self._lista_material = lista_material
18         self._materia = materia
19         self._fecha_prestamo = fecha_prestamo
20         self._fecha_devolucion = fecha_devolucion
21         self._nombre_solicitante = None
22         Pedido.contador_pedido += 1
23
24     @property
25     def id_pedido(self):
26         return self._id_pedido
27
28 if __name__ == '__main__':
```

ProgramacionObjeto > PEDIDO.py 98:35 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)



```
27 @property
28 def solicitante(self):
29     return self._solicitante
30
31 1 usage
32 @property
33 def lista_material(self):
34     return self._lista_material
35
36 @lista_material.setter
37 def lista_material(self, nuevo_lista_material):
38     self._lista_material = nuevo_lista_material
39
40 3 usages
41 @property
42 def fecha_prestamo(self):
43     return self._fecha_prestamo
44
45 @fecha_prestamo.setter
46 def fecha_prestamo(self, nuevo_fecha_prestamo):
47     self._fecha_prestamo = nuevo_fecha_prestamo
48
49 3 usages
50 @property
51 def fecha_devolucion(self):
52     return self._fecha_devolucion
53
54 if __name__ == '__main__':
```

ProgramacionObjeto > PEDIDO.py 98:35 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)

```
PC  P0 ProgramacionObjeto  Version control  PEDIDO  PEDIDO.py x
MATERIAL.py LIBRO.py REVISTA.py PERSONA1.py ESTUDIANTE.py DOCENTE.py PEDIDO.py x
51 @fecha_devolucion.setter
52 def fecha_devolucion(self, nuevo_fecha_devolucion):
53     self._fecha_devolucion = nuevo_fecha_devolucion
54
55 1 usage
56 @property
57 def nombre_solicitante(self):
58     return self._nombre_solicitante
59
60 @nombre_solicitante.setter
61 def nombre_solicitante(self, nuevo_nombre):
62     self._nombre_solicitante = nuevo_nombre
63
64 2 usages
65 def pedir_material(self, material):
66     self._lista_material = material
67     self._fecha_prestamo = datetime.now()
68     self._fecha_devolucion = None
69     if self._solicitante:
70         self._nombre_solicitante = self._solicitante.nombre
71         print(f"Se ha realizado el préstamo del material {self._lista_material} a {self._nombre_solicitante}.")
72
73 2 usages
74 def devolver_material(self):
75     if self._solicitante:
76         self._fecha_devolucion = datetime.now()
77
78 if __name__ == '__main__':
79
98:35 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)
Buscar 30°C Mayorm. nubla... 17:29 26/11/2023
```

```
PC  P0 ProgramacionObjeto  Version control  PEDIDO  PEDIDO.py x
MATERIAL.py LIBRO.py REVISTA.py PERSONA1.py ESTUDIANTE.py DOCENTE.py PEDIDO.py x
73 self._fecha_devolucion = datetime.now()
74 print(f"Se ha devuelto el material {self._lista_material} de {self._nombre_solicitante}.")
75 else:
76     print("No hay material para devolver o no se ha registrado un solicitante.")
77
78
79 if __name__ == '__main__':
80     # Caso con Revista y Estudiante
81     revista1 = Revista(codigo=211, autor='Travis Griffin', titulo='National Geographic', anio=2021, editorial='Edelvives', disponible=True, cantidad_disponible=19)
82     estudiante = Estudiante(cedula="0931054233", nombre="Melanie", apellido="Gusqui", email="melaniegusqui@gmail.com", telefono="0997079658", direccion="Acac")
83
84     pedido1 = Pedido(solicitante=estudiante)
85
86     pedido1.pedir_material(revista1)
87     print(f"Fecha de préstamo: {pedido1.fecha_prestamo}")
88
89     pedido1.devolver_material()
90     print(f"Fecha de devolución: {pedido1.fecha_devolucion}")
91
92     # Caso con Libro y Docente
93     libro1 = Libro(codigo=101, autor='Shelby Mahurin', titulo='Asesino de Brujas', anio=2018, editorial='Puck', disponible=True, cantidad_disponible=19)
94     docente = Docente(cedula="0974321421", nombre="Mercedes", apellido="Mendoza", email="mercemend@hotmail.com", telefono="0987235117", direccion="Acac")
95
96     pedido2 = Pedido(solicitante=docente)
97
98     pedido2.pedir_material(libro1)
99     print(f"Fecha de préstamo: {pedido2.fecha_prestamo}")
100
101 if __name__ == '__main__':
102
98:35 CRLF UTF-8 4 spaces Python 3.12 (ProgramacionObjeto)
Buscar 30°C Mayorm. nubla... 17:29 26/11/2023
```

