


Ejercicios de aplicación

PROGRAMACIÓN ORIENTADA A OBJETOS EN PYTHON

Es hora de que pongas en práctica todo lo aprendido. 

Este apartado tiene el objetivo de ayudarte a seguir potenciando tus habilidades, por lo que a continuación encontrarás diferentes desafíos que podrás resolver de forma independiente y a tu ritmo.

Más adelante conseguirás las resoluciones para que valides tus respuestas y puedas monitorear tu progreso. 

¡Manos a la obra!


Desafío

Crear una clase `Persona` que incluya los atributos `nombre`, `edad`, y `correo_electronico`. Además, implementar métodos que permitan:

- Mostrar los datos de la persona en un formato amigable.
 - Validar que el correo electrónico sea válido utilizando expresiones regulares.
 - Actualizar los datos de la persona.
 - Verificar si la persona es mayor de edad.
-

¿Dónde se lleva a cabo? 


Visual Studio Code o cualquier editor de Python de tu preferencia.

Tiempo de dedicación 

1:30 horas

Recursos 

Documentación sobre manejo de clases y métodos en Python, video explicativo sobre validación de correos electrónicos con expresiones regulares (opcional), tutorial sobre manejo de expresiones regulares en Python.

Plus 

Agrega una función para comparar la edad de dos objetos `Persona` y devolver quién es mayor.

Resolución del ejercicio:

Código Completo y Explicación

```
import re # Necesario para la validación de correos electrónicos
```

```
class Persona:
```

```
    def __init__(self, nombre, edad, correo_electronico):
```

```
        self.nombre = nombre
```

```
        self.edad = edad
```

```
        self.correo_electronico = correo_electronico
```

```
# Método para mostrar los datos de la persona
```

```
def mostrar_datos(self):
```

```
    print(f"Nombre: {self.nombre}, Edad: {self.edad}, Email: {self.correo_electronico}")
```

```
# Método para validar si el correo electrónico es válido
```

```
def validar_correo(self):
```

```
    patron = r'^[\w\.-]+@[\w\.-]+\.\w+$'
```

```
    if re.match(patron, self.correo_electronico):
```

```
        return True
```

```
else:
```

```
    return False
```

```
# Método para actualizar los datos de la persona
```

```
def actualizar_datos(self, nombre=None, edad=None, correo_electronico=None):
```

```
    if nombre:
```

```
        self.nombre = nombre
```

```
    if edad:
```

```
        self.edad = edad
```

```
    if correo_electronico:
```

```
        self.correo_electronico = correo_electronico
```

```
# Método para verificar si la persona es mayor de edad
```

```
def es_mayor_de_edad(self):
```

```
    return self.edad >= 18
```

1. Definir la clase `Persona`: La clase tiene un constructor que inicializa `nombre`, `edad`, y `correo_electronico`.
2. Implementar `mostrar_datos()`: Este método muestra los datos de la persona de forma amigable en la consola.
3. Crear `validar_correo()`: Verifica si el correo tiene un formato válido utilizando expresiones regulares.
4. `actualizar_datos()`: Permite modificar el `nombre`, `edad`, o `correo_electronico` de la persona.
5. `es_mayor_de_edad()`: Retorna `True` si la persona tiene 18 años o más, y `False` en caso contrario.