

```
from abc import ABC, abstractmethod
```

```
# Clase abstracta que define la interfaz para los manejadores.
```

```
class SupportHandler(ABC):
```

```
    def __init__(self):
```

```
        self._next_handler = None
```

```
    def set_next(self, handler):
```

```
        """Establece el siguiente manejador en la cadena"""
```

```
        self._next_handler = handler
```

```
        return handler # Permite encadenamiento
```

```
    @abstractmethod
```

```
    def handle(self, request):
```

```
        """Método abstracto que debe implementar cada manejador concreto"""
```

```
        pass
```

```
# Manejador de soporte básico.
```

```
# Resuelve problemas simples como contraseñas y acceso básico.
```

```
class BasicSupport(SupportHandler):
```

```
    def handle(self, request):
```

```
        if request.priority in ['low', 'medium']:
```

```
            print(f"BasicSupport resolviendo: {request.description}")
```

```
            request.resolved = True
```

```
            request.handler_name = "BasicSupport"
```

```
            return request
```

```
        else:
```

```
            if self._next_handler:
```

```
    print(f"BasicSupport escalando a siguiente nivel...")
    return self._next_handler.handle(request)
else:
    print(f"No hay manejador disponible para: {request}")
    return request
```

# Manejador de soporte técnico especializado.

# Resuelve problemas técnicos complejos.

```
class SpecializedTechnician(SupportHandler):
```

```
    def handle(self, request):
```

```
        if request.priority in ['medium', 'high']:
```

```
            print(f"SpecializedTechnician resolviendo: {request.description}")
```

```
            request.resolved = True
```

```
            request.handler_name = "SpecializedTechnician"
```

```
            return request
```

```
        else:
```

```
            if self._next_handler:
```

```
                print(f"SpecializedTechnician escalando a siguiente nivel...")
```

```
                return self._next_handler.handle(request)
```

```
            else:
```

```
                print(f"No hay manejador disponible para: {request}")
```

```
            return request
```

# Manejador de gerente de soporte.

# Resuelve problemas críticos o no resueltos por otros niveles.

```
class SupportManager(SupportHandler):
```

```
    def handle(self, request):
```

```
        print(f"SupportManager resolviendo: {request.description}")
```

```
request.resolved = True

request.handler_name = "SupportManager"

if self._next_handler:

    return self._next_handler.handle(request)

return request
```

# Representa una solicitud de soporte técnico.

# Contiene información sobre el problema a resolver.

class SupportRequest:

```
def __init__(self, request_id, description, priority):

    self.request_id = request_id

    self.description = description

    self.priority = priority # Prioridad: 'low', 'medium', 'high', 'critical'

    self.resolved = False

    self.handler_name = None
```

```
def __str__(self):

    return (f"ID: {self.request_id} | "

            f"Prioridad: {self.priority} | "

            f"Descripción: {self.description} | "

            f"Resuelto: {self.resolved}")
```

# Función principal que demuestra el uso del patrón Chain of Responsibility.

# Crea la cadena de manejadores y procesa solicitudes.

```
def main():

    # Crear los manejadores

    basic_support = BasicSupport()

    specialized_tech = SpecializedTechnician()
```

```
support_manager = SupportManager()
```

```
# Construir la cadena: BasicSupport -> SpecializedTechnician -> SupportManager
```

```
basic_support.set_next(specialized_tech).set_next(support_manager)
```

```
# Crear solicitudes de prueba
```

```
requests = [
```

```
    SupportRequest(1, "Olvidé mi contraseña", "low"),
```

```
    SupportRequest(2, "Error en la configuración del servidor", "high"),
```

```
    SupportRequest(3, "Fallo crítico del sistema", "critical"),
```

```
]
```

```
# Procesar cada solicitud
```

```
for request in requests:
```

```
    print(f"\n--- Procesando {request} ---")
```

```
    result = basic_support.handle(request)
```

```
    print(f"Resultado: {result}\n")
```

```
if __name__ == "__main__":
```

```
    main()
```

```
PROBLEMAS_PRACTICOS.PY X
ETAPA 1 > PROBLEMAS_PRACTICOS.PY > SpecializedTechnician > handle
20 class BasicSupport(SupportHandler):
21     def handle(self, request):
22         # Escalando a siguiente nivel...
23         return self.next_handler.handle(request)
24     else:
25         print(f"No hay manejador disponible para: {request}")
26         return request
27
28 # Manejador de soporte técnico especializado.
29 # Resuelve problemas técnicos complejos.
30 class SpecializedTechnician(SupportHandler):
31     def handle(self, request):
32         if request.priority in ["medium", "high"]:
33             # Resolviendo problema...
34             return True
35         else:
36             # Escalando a siguiente nivel...
37             return self.next_handler.handle(request)
38
39 # Soporte Manager
40 class SupportManager:
41     def __init__(self):
42         self.handlers = []
43         self.current_handler = None
44
45     def add_handler(self, handler):
46         self.handlers.append(handler)
47
48     def get_handler(self, request):
49         for handler in self.handlers:
50             if handler.can_handle(request):
51                 return handler
52         return None
53
54     def handle_request(self, request):
55         handler = self.get_handler(request)
56         if handler:
57             return handler.handle(request)
58         return False
59
60 # Ejecución
61 if __name__ == "__main__":
62     manager = SupportManager()
63     manager.add_handler(BasicSupport(manager))
64     manager.add_handler(SpecializedTechnician(manager))
65
66     # Simulando solicitudes
67     requests = [
68         {"id": 1, "description": "Olvidé mi contraseña", "priority": "low"},
69         {"id": 2, "description": "Error en la configuración del servidor", "priority": "high"},
70         {"id": 3, "description": "Fallo crítico del sistema", "priority": "critical"}
71     ]
72
73     for request in requests:
74         result = manager.handle_request(request)
75         print(f"Resultado: ID: {request['id']} | Prioridad: {request['priority']} | Descripción: {request['description']} | Resuelto: {result}")
76
77     # Limpieza
78     del manager.handlers
79     del manager.current_handler
80     del requests
81
82     # Finalización
83     print("Programa finalizado exitosamente.")
84
85 # Fin del código
```

PS C:\Users\poolg\OneDrive - INTEC\Escritorio\BOB\PYTHON> & C:/Users/poolg/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/poolg/OneDrive - INTEC/Escritorio/BOB/PYTHON/ETAPA 1/PROBLEMAS\_PRACTICOS.PY"

Price of AAPL changed from 155 to 160  
John received alert: AAPL is now \$160  
Alice received alert: AAPL is now \$160

PS C:\Users\poolg\OneDrive - INTEC\Escritorio\BOB\PYTHON> & C:/Users/poolg/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/poolg/OneDrive - INTEC/Escritorio/BOB/PYTHON/ETAPA 1/PROBLEMAS\_PRACTICOS.PY"

--- Procesando ID: 1 | Prioridad: low | Descripción: Olvidé mi contraseña | Resuelto: False ---  
BasicSupport resolviendo: Olvidé mi contraseña  
Resultado: ID: 1 | Prioridad: low | Descripción: Olvidé mi contraseña | Resuelto: True

--- Procesando ID: 2 | Prioridad: high | Descripción: Error en la configuración del servidor | Resuelto: False ---  
BasicSupport escalando a siguiente nivel...  
SpecializedTechnician resolviendo: Error en la configuración del servidor  
Resultado: ID: 2 | Prioridad: high | Descripción: Error en la configuración del servidor | Resuelto: True

--- Procesando ID: 3 | Prioridad: critical | Descripción: Fallo crítico del sistema | Resuelto: False ---  
BasicSupport escalando a siguiente nivel...  
SpecializedTechnician escalando a siguiente nivel...  
SupportManager resolviendo: Fallo crítico del sistema  
Resultado: ID: 3 | Prioridad: critical | Descripción: Fallo crítico del sistema | Resuelto: True

PS C:\Users\poolg\OneDrive - INTEC\Escritorio\BOB\PYTHON>