# dac\_simple.py

class DAC:

def \_\_init\_\_(self): # <-- Fixed: double underscores

self.acl = {

'alice': {'file1.txt': 'rw'},

'bob': {'file1.txt': 'r'},

}

def check\_access(self, user, file, operation):

if user not in self.acl:

print("Access Denied: Unknown user")

return

permissions = self.acl[user].get(file, '')

if operation in permissions:

print(f"Access Granted: {user} can {operation} {file}")

try:

with open(file, 'r') as f:

print(f.read())

except FileNotFoundError:

print("Access Granted, but file does not exist.")

else:

print("Access Denied: Permission not granted")

# Example Usage

dac = DAC()

dac.check\_access('alice', 'file1.txt', 'r')

dac.check\_access('bob', 'file1.txt', 'w')

# mac\_simple.py

class MAC:

def \_\_init\_\_(self): # <-- Fixed constructor

self.clearance = {'alice': 3, 'bob': 1} # 1 = Confidential, 2 = Secret, 3 = Top Secret

self.file\_labels = {'file1.txt': 2, 'file2.txt': 3}

def check\_access(self, user, file):

if user not in self.clearance:

print("Access Denied: Unknown user")

return

user\_level = self.clearance[user]

file\_level = self.file\_labels.get(file)

if file\_level is None:

print("Access Denied: Unknown file")

return

if user\_level >= file\_level:

print(f"Access Granted: {user} has clearance level {user\_level}")

try:

with open(file, 'r') as f:

print(f.read())

except FileNotFoundError:

print("Access Granted, but file does not exist.")

else:

print("Access Denied: Clearance too low")

# Run MAC

mac = MAC()

mac.check\_access('alice', 'file2.txt') # Allowed

mac.check\_access('bob', 'file2.txt') # Denied