# ERRONKA:

**Er1-UD1-Whois**

# Enuntziatua:

Aeronautika eta Espazioaren Euskal Agentziak (AEEA) urte asko daramatza lanean ezkutuan. 6 hilabete barru bere aurkezpen publikoa egingo du, eta erronka aitzindari askori ekingo die: robot batek Neptuno barnetik arakatzea, Marteren lurra zapaltzeko espedizioa bidali...  
  
Izango duen esposizio handiaz jabetuta, agentziak bere sistema informatiko osoaren auditoria zehatza egin nahi du. Zure enpresa horretarako kontratatutu dute, eta zure lehen eginkizuna kontzientziazio-kanpaina bat zuzentzea da. Informatika mintegiak gaur bertan txosten bat bidaltzeko eskatu dizu; dokumentuaren helburua zera da: erakustea zuzendaritza taldeko kideei zein erraz lortu daitekeen enpresaren informazioa. Eskarmentuko hackerra izanik, lasai hartu duzu kontua. Hondartzara eramangarria eraman, eta bertatik egingo duzu lan. Han hemenka ibili ostean, ordenagailua piztu, eta ordu erdirako bateria baino ez daukazu!!! Eta kargadorea etxean dago. 30 minutu dituzu datu esanguratsuak topatu eta zerbait txukuna bidaltzeko. Txostena ez da egoitza nagusiko sareari buruzkoa, beste nonbaiteko domeinu sekretu bati buruzkoa baizik.   
Gai izango al zara informazio interesgarria topatzeko?

# Mugarriak:

**Er1-UD1-Mug1: 5 puntu**

Zein da scanme.org domeinuaren 2. domeinu-izena (2 zifra daukana)?  
  
EMAITZAREN FORMATUA: xxxxxx.xxxx.xxxxx  
Adibidez: tknika2.hezk.ziber

# Pistak:

**Er1-UD1-Mug1-Pista1: -1 puntu**

"consulta" aldagaia objektu bat da; besteak beste, honako atributu hauek ditu:   
 -name  
 -name\_servers   
 -registrar  
 -creation\_date  
 -expiration\_date  
 -last\_updated

**Er1-UD1-Mug1-Pista2: -2 puntu**

for ns in consulta.name\_servers:  
 print (?????????)

# ERRONKA:

**Er2-UD1-DNS**

# Enuntziatua:

DNS zerbitzuaren gaineko azpiauditoretza bat egin behar duzu orain. Zure eramangarria egoitza nagusiko sarean dago, domeinu nagusian. Baina, dakizunez, beste egoitza eta domeinu batzuk ere baditu erakundeak; hortaz, azpi-auditoretza kutxa zuri zein kutxa beltzekoa izango da. Domeinu nagusia barnetik aztertuko duzu (kutxa zuria) eta, besteak, kanpotik (kutxa beltza.

# Mugarriak:

**Er2-UD1-Mug1: 5 puntu**

ER2\_UD1\_Kontsulta\_DNS.py scripta erabili. Zein da MX erregistroaren balioa scanme.org domeinuan?  
  
EMAITZAREN FORMATUA: xxxx.xxxxx.xxx  
Adibidez: mail.ziber.eus

**Er2-UD1-Mug2: 5 puntu**

ER2\_UD1\_Transferentzia\_DNS.py scripta erabili.   
Zein da zonetransfer.me domeinuko (81.4.108.41) Pipparen telefonoa?  
  
EMAITZAREN FORMATUA: +NNNNNNNNNNN.  
Zuriunerik gabe!!  
Adibidez: +34567890123

**Er2-UD1-Mug3: 5 puntu**

ER2\_UD1\_Transferentzia\_DNS.py scripta erabili.   
Eskatu zonalde transferentzia bat zibertknika.eus domeinuko DNS zerbitzariari (192.168.1.150) . Zein da jasotako erantzunaren azken hitza?

**Er2-UD1-Mug4: 15 puntu**

Lehenengo eta behin, hartu zibertknika.eus domeinuko DNS zerbitzariaren (192.168.1.150) kontrola (root/python).   
Aldatu konfigurazio fitxategi bat Kali makinatik egindako zonalde-transferentzia eskaerak onartuak izan daitezen.   
Ondoren, ER2\_UD1\_Transferentzia\_DNS.py scripta erabili.   
Zein da dnssecondary izeneko makinaren IPa?

# Pistak:

**Er2-UD1-Mug1-Pista 1: -1 puntu**

print(....)

**Er2-UD1-Mug1-Pista 2: -2 puntu**

Ikusi zelan inprimatzen diren beste erregistroak.

**Er2-UD1-Mug2-Pista 1: -1 puntu**

zone = dns.zone.from\_xfr(...)

**Er2-UD1-Mug2-Pista 2: -2 puntu**

zone = dns.zone.from\_xfr(dns.query.xfr(IP\_helbidea, DOMEINUa))

**Er2-UD1-Mug3-Pista1: -1 puntu**

Zer saretan zaude?

**Er2-UD1-Mug3-Pista 2: -2 puntu**

Zein DNS zerbitzari daukazu konfiguratuta?  
nslookup (adibidez) erabili DNS makinaren kontra domeinua jakiteko...

**Er2-UD1-Mug4-Pista1: -2 puntu**

Konektatu DNS makinara ssh bidez. Kredentzialak enuntizatuan daude.   
Birpasatu apunteak jakiteko zer ikutu bertan.

**Er2-UD1-Mug4-Pista 2: -6 puntu**

Moldatu beharreko fitxategia /etc/bind/named.conf.local da.   
Gehitu allow-transfer lerroa... Kali-ren IPa jarriz.   
Ez ahaztu bind9 zerbitzua berrabiatzea.

# ERRONKA:

**Er3-UD1-Geolokalizazioa**

# Enuntziatua:

Lan nekeza da erakundeak Munduan zehar dituen aktibo guztiak miatzea. Fase honetan, garrantzitsua da zenbait gailuren kokapen fisikoa ezagutzea. Beraz, prozesua arintzeko, script pare bat sortzea erabaki duzu. Lehenengoak, koordenada batzukk jasota, haien hiria ematen dizu; bigarrenak, aldiz, kontrako prozesua egiten du.

# Mugarriak:

**Er3-UD1-Mug1: 15 puntu**

Koordenada batzuk topatu dituzu ezkutuko fitxategi batean:   
 Latitudea: 38.80  
 Longitudea: -9.37  
  
Aukera egokia izan daiteke zeure lehenengo scripta probatzeko. Zein hiriri dagozkio koordenada horiek?   
  
ER3\_UD1\_Geolokalizazioa\_Bat.py scripta erabili.

**Er3-UD1-Mug2: 10 puntu**

Topatutako informazioaren artean helbide hau ikusi duzu:   
 Mineros 14, Río Turbio, Santa Cruz, Argentina  
  
Berriz ere aukera aproposa izan daiteke zeure script bat probatzeko. Zein koordenada dagozkio helbide horri?   
  
ER3\_UD1\_Geolokalizazioa\_Bi.py scripta erabili.   
  
EMAITZAREN FORMATUA: [Latitudea];[Longitudea] (2 dezimalekin, puntua erabilita)  
Adibidez: 45.67;-83.21

# Pistak:

**Er3-UD1-Mug1-Pista 1: -2 puntu**

Funtzio biek True edo False itzuli behar dute.

**Er3-UD1-Mug1-Pista 2: -7 puntu**

def latitud\_correcta(lat):   
 if not (-90 <= lat <= 90):  
 return False  
 else:  
 return True  
  
if obtener\_ciudad(latitud, longitud) is not None:  
.......

**Er3-UD1-Mug2-Pista 1: -2 puntu**

"localizacion" aldagaia objektu bat da. Bere atributuen artean honako hauek daude:   
 address  
 longitude  
 point  
 raw  
 latitude

**Er3-UD1-Mug2-Pista 2: -4 puntu**

if localizacion is not None:  
 latitud = localizacion.latitude  
........

# ERRONKA:

**Er4-UD1-Metadatuak**

# Enuntziatua:

Bi argazki aurkitu dituzu zerbitzari batean. Itxuraz, ez daukate ezer berezirik, baina zibersegurtasunean edozein fitxategi inozok eman ahal dizu informazio baliagarria. Gai izango al zara irudi horietatik zerbait mamitsua lortzeko?

# Mugarriak:

**Er4-UD1-Mug1: 12 puntu**

Zer kamera eredurekin izan zen hartua "er4\_irudia\_bat.jpg" argazkia?  
Erabili Er4\_UD1\_Metadatuak\_Bat.py scripta.

**Er4-UD1-Mug2: 28 puntu**

Zer hiritan izan zen hartua "er4\_irudia\_bi.jpg" argazkia?  
Erabili Er4\_UD1\_Metadatuak\_Bi.py scripta.

# Pistak:

**Er4-UD1-Mug1-Pista 1: -2 puntu**

Ikusteko ea fitxategia existitzen den os.path.isfile erabili dezakezu.   
Beste lanak egiteko, erabili definitutako funtzioak ordena egokian.

**Er4-UD1-Mug1-Pista 2: -5 puntu**

while not os.path.isfile (archivo):  
 archivo= input ("El archivo no existe en la ruta especificada. Introduce la ruta nuevamente:")  
 metadatos = obtener\_datos\_exif(archivo)  
 modelo\_camara = obtener\_modelo\_camara(metadatos)

**Er4-UD1-Mug2-Pista 1: -5 puntu**

from ER3\_UD1\_Geolokalizazioa\_Bat import obtener\_ciudad, latitud\_correcta, longitud\_correcta

**Er4-UD1-Mug2-Pista 2: -10 puntu**

.....................  
   
 archivo= input (.......)  
 ??????? = obtener\_datos\_exif(??????)  
 ??????? = obtener\_geolocalizacion(???????)  
 latitud, longitud = obtener\_coordenadas(????????)  
 if latitud\_correcta(????????) and longitud\_correcta(???????):  
 ciudad = obtener\_ciudad(??????, ???????)   
  
..........

**Er4-UD1-Mug2-Pista 3: -24 puntu**

#El módulo PIL proporciona funciones y clases para manipular imágenes. Importamos su clase Image  
from PIL import Image  
#Importamos las constantes TAGS y GPSTAGS del módulo ExifTags en el paquete PIL. Estas constantes contienen los nombres de las  
#etiquetas EXIF estándar y etiquetas GPS respectivamente.  
from PIL.ExifTags import TAGS, GPSTAGS  
#Importamos este módulo para interactuar con el sistema operativo. En este ejercicio en concreto, para comprobar si el archivo  
#introducido por el usuario existe  
import os  
#(◕‿−)OPCIONAL: Importamos las funciones obtener\_ciudad, latitud\_correcta y longitud\_correcta del script ER3\_UD1\_Geolokalizazioa\_Bat  
from ER3\_UD1\_Geolokalizazioa\_Bat\_SOLUZIOA import obtener\_ciudad, latitud\_correcta, longitud\_correcta  
  
#Función que obtiene los datos EXIF de una imagen, devolviéndolos en una tupla  
def obtener\_datos\_exif(imagen):  
 diccionario\_exif = {}  
 try:  
 img = Image.open(imagen)  
 diccionario\_exif = img.\_getexif()  
 except:  
 print ("Error inesperado al obtener los datos de la imagen proporcionada")  
 return diccionario\_exif  
  
# Función que obtiene los datos de geolocalización a partir de los datos EXIF de una imagen y los devuelve en una tupla  
def obtener\_geolocalizacion(datos\_exif):  
 if not datos\_exif:  
 raise ValueError("No se encontraron metadatos EXIF")  
 geolocalizacion = {}  
 clave\_gps\_info = None  
 #Estamos iterando sobre las claves y valores del diccionario TAGS. En cada iteración, key representa la clave actual y  
 #value representa el valor correspondiente. clave\_gps\_info almacenará el valor del código de la etiqueta GPSInfo. Podríamos  
 #haber asumido como clave la que tiene en la actualizad (34853), pero, ¿y si varía?  
 for clave, valor in TAGS.items():  
 if valor == 'GPSInfo':  
 clave\_gps\_info = clave  
 #Si hay datos de geolocalización y la clave encontrada está dentro del diccionario recibido, formamos una tupla ("geolocalizacion")  
 #sólo con la información que nos interesa.   
 if clave\_gps\_info is not None and clave\_gps\_info in datos\_exif:  
 gps\_info = datos\_exif[clave\_gps\_info]  
 for clave\_gps, valor\_gps in gps\_info.items():  
 if clave\_gps in GPSTAGS:  
 geolocalizacion[GPSTAGS[clave\_gps]] = valor\_gps  
 #El contenido de la tupla "geolocalizacion" será algo así como:  
 #{'GPSVersionID': b'\x02\x02\x00\x00', 'GPSLatitudeRef': 'N', 'GPSLatitude': (40.0, 41.0, 21.12),  
 # 'GPSLongitudeRef': 'W', 'GPSLongitude': (74.0, 2.0, 40.2), 'GPSAltitudeRef': b'\x00', 'GPSAltitude': 10.0,  
 # 'GPSTimeStamp': (0.0, 0.0, 0.0), 'GPSProcessingMethod': '', 'GPSDateStamp': '00:00:00'}  
 return geolocalizacion  
  
# Función que obtiene las coordenadas GPS de latitud y longitud a partir de los datos de geolocalización y los devuelve en  
# dos variables  
def obtener\_coordenadas(geolocalizacion):  
 latit = None  
 longit = None  
 #Dentro de la tupla, iteramos sólo sobre las claves que nos interesan para obtener sus valores.   
 #Convertimos el valor de esas claves a coordenada decimal. Por ejemplo, GPSLatitude lo pasaremos de (40.0, 41.0, 21.12)  
 #a 40.6892  
 for key in ['GPSLatitude', 'GPSLongitude', 'GPSLatitudeRef', 'GPSLongitudeRef']:  
 if key in geolocalizacion:  
 if key == 'GPSLatitude':  
 latit = convertir\_a\_decimal(geolocalizacion[key])  
 if key == 'GPSLongitude':  
 longit = convertir\_a\_decimal(geolocalizacion[key])  
 if key == 'GPSLatitudeRef' and geolocalizacion[key] == 'S':  
 latit = 0 - latit  
 if key == 'GPSLongitudeRef' and geolocalizacion[key] == 'W':  
 longit = 0 - longit  
 return latit, longit  
  
# Función que convierte los valores GPS de grados, minutos y segundos a valores decimales  
def convertir\_a\_decimal(coordenada):  
 grados = float(coordenada[0])  
 minutos = float(coordenada[1])  
 segundos = float(coordenada[2])  
 return grados + minutos / 60 + segundos / 3600  
  
#(◕‿−) Escribir el cuerpo del script  
if \_\_name\_\_ == '\_\_main\_\_':  
 #Pedimos al usuario que introduzca la ruta de un archivo. Mientras el archivo introducido no exista, se lo volveremos a  
 #solicitar.   
 archivo = input ("Introduce la ruta completa del archivo: ")  
 while not os.path.isfile (archivo):  
 archivo= input ("El archivo no existe en la ruta especificada. Introduce la ruta nuevamente:")  
 #Obtenemos los metadatos del archivo  
 datos\_exif = obtener\_datos\_exif(archivo)  
 #Creamos una tupla sólo con los metadatos relativos a la geolocalización  
 geolocalizacion = obtener\_geolocalizacion(datos\_exif)  
 #Creamos una tupla sólo con los metadatos relativos a la geolocalización  
 latitud, longitud = obtener\_coordenadas(geolocalizacion)  
 #Comprobamos si la latitud y la longitud tienen valores crrectos. Si los tienen, obtenemos la ciudad. Para ello,  
 #podemos utilizar funciones del script ER3\_UD1\_Geolokalizazioa\_Bat  
 if latitud\_correcta(latitud) and longitud\_correcta(longitud):  
 ciudad = obtener\_ciudad(latitud, longitud)  
 if ciudad is not None:  
 print(f"La foto se ha tomado en la ciudad de {ciudad}")  
 else:  
 print("No se ha podido obtener la ciudad en la que se ha tomado la foto")  
 else:  
 print ("El valor de la latitud o de la longitud no es correcto")

#############################################################