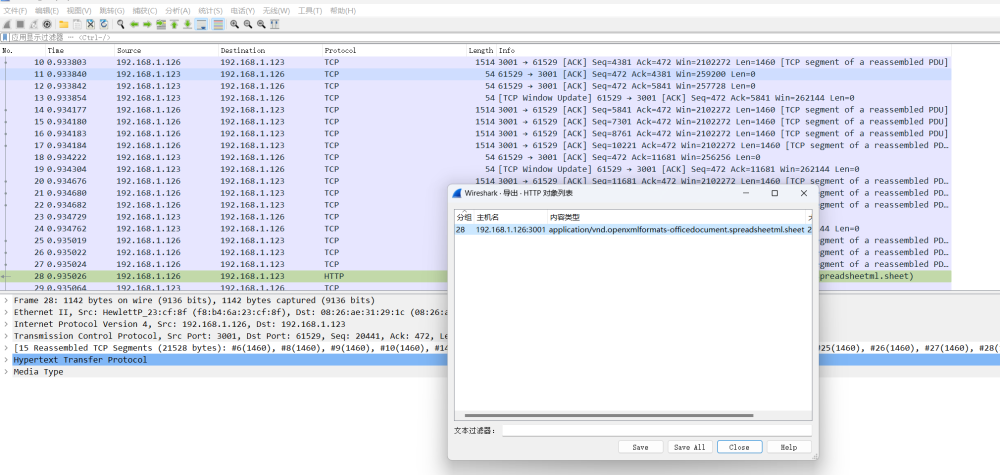
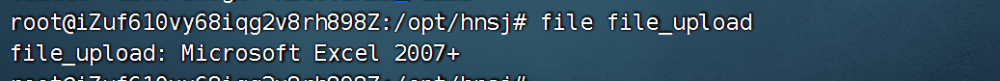
# transfer\_info流量分析题目

流量导出，http导出http对象

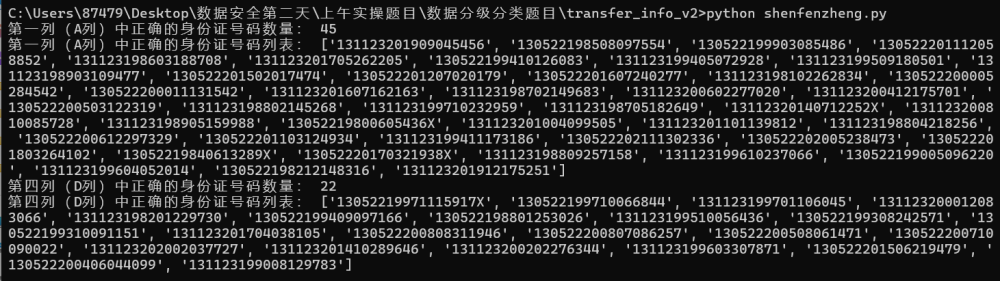
文件识别

file file\_upload

修改文件后缀为xlsx

## 身份证代码识别C1

from openpyxl import load\_workbook # 用于处理 Excel 文件的库。  
import re  
  
# 定义一个函数用于校验身份证号码的有效性  
def is\_valid\_idcard(idcard):  
 # 验证长度和格式  
 if len(idcard) != 18 or not re.match(r"^\d{17}[\dX]$", idcard):  
 return False  
  
 # 校验码计算  
 nums = [int(x) for x in idcard[:-1]] # 将身份证号码的前17位转换为整数列表  
 weights = (7, 9, 10, 5, 8, 4, 2, 1, 6, 3, 7, 9, 10, 5, 8, 4, 2) # 加权因子  
 checksum = sum(a \* b for a, b in zip(weights, nums)) # 计算加权和  
 charset = '10X98765432' # 校验码字符集  
 return charset[checksum % 11] == idcard[-1].upper() # 校验码验证  
  
# 加载Excel工作簿并选择第一个工作表  
wb = load\_workbook('test.xlsx') # 读取名为'test.xlsx'的Excel文件  
sheet = wb['Sheet1'] # 选择工作簿中的第一个工作表'Sheet1'  
  
correct\_idcards\_col1 = [] # 创建一个空列表用于存储第一列（A列）中正确的身份证号码  
correct\_idcards\_col4 = [] # 创建一个空列表用于存储第四列（D列）中正确的身份证号码  
  
# 从工作表中提取A1-A51和D1-D51列的数据并验证身份证号码  
for i in range(2, 52): # 遍历每列的行，从第2行到第51行  
 idcard\_col1 = sheet.cell(i, 1).value # 提取A列（第1列）的数据  
 idcard\_col4 = sheet.cell(i, 4).value # 提取D列（第4列）的数据  
   
 # 验证A列的身份证号码  
 if idcard\_col1:  
 # 验证前6位是否在指定范围内  
 if re.match(r"^(131123|130522)", idcard\_col1):  
 # 提取出生日期并验证范围  
 try:  
 year = int(idcard\_col1[6:10])  
 month = int(idcard\_col1[10:12])  
 day = int(idcard\_col1[12:14])  
 except ValueError:  
 continue  
   
 if (1930 <= year <= 2024) and (1 <= month <= 12) and (1 <= day <= 31) and is\_valid\_idcard(idcard\_col1):  
 correct\_idcards\_col1.append(idcard\_col1) # 将正确的身份证号码添加到列表中  
   
 # 验证D列的身份证号码  
 if idcard\_col4:  
 # 验证前6位是否在指定范围内  
 if re.match(r"^(131123|130522)", idcard\_col4):  
 # 提取出生日期并验证范围  
 try:  
 year = int(idcard\_col4[6:10])  
 month = int(idcard\_col4[10:12])  
 day = int(idcard\_col4[12:14])  
 except ValueError:  
 continue  
   
 if (1930 <= year <= 2024) and (1 <= month <= 12) and (1 <= day <= 31) and is\_valid\_idcard(idcard\_col4):  
 correct\_idcards\_col4.append(idcard\_col4) # 将正确的身份证号码添加到列表中  
  
# 统计正确的身份证号码数量  
correct\_count\_col1 = len(correct\_idcards\_col1)  
correct\_count\_col4 = len(correct\_idcards\_col4)  
  
print("第一列（A列）中正确的身份证号码数量：", correct\_count\_col1)  
print("第一列（A列）中正确的身份证号码列表：", correct\_idcards\_col1)  
  
print("第四列（D列）中正确的身份证号码数量：", correct\_count\_col4)  
print("第四列（D列）中正确的身份证号码列表：", correct\_idcards\_col4)

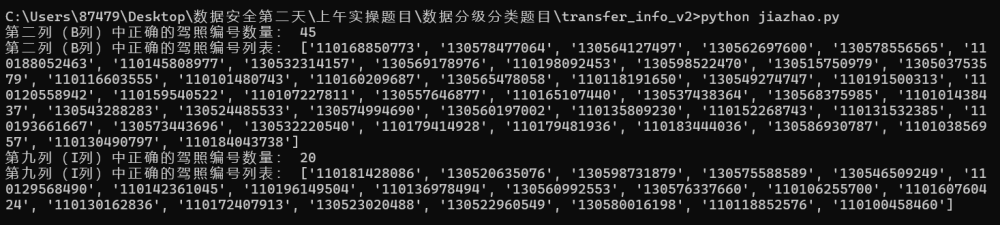
符合身份证的A1列，也就是C1

## 驾照信息识别C2

from openpyxl import load\_workbook  
import re  
  
# 定义一个函数用于校验驾照编号的有效性

#这个函数用于验证驾照编号的有效性。

#如果编号不是 12 位数字或者前 4 位不以 1101 或 1305 开头，函数会返回 False，表示编号无效，否则返回 True。  
def is\_valid\_drivelicense(drivelicense):  
 # 验证长度和格式  
 if len(drivelicense) != 12 or not drivelicense.isdigit():  
 return False  
  
 # 验证前4位是否在指定范围内  
 if not re.match(r"^(1101|1305)", drivelicense):  
 return False  
  
 return True  
  
# 加载Excel工作簿并选择第一个工作表  
wb = load\_workbook('test.xlsx') # 读取名为'test.xlsx'的Excel文件  
sheet = wb['Sheet1'] # 选择工作簿中的第一个工作表'Sheet1'  
  
correct\_drivelicenses\_col2 = [] # 创建一个空列表用于存储第二列（B列）中正确的驾照编号  
correct\_drivelicenses\_col9 = [] # 创建一个空列表用于存储第九列（I列）中正确的驾照编号  
  
# 从工作表中提取B1-B51和I1-I51列的数据并验证驾照编号  
for i in range(2, 52): # 遍历每列的行，从第2行到第51行  
 drivelicense\_col2 = sheet.cell(i, 2).value # 提取B列（第2列）的数据  
 drivelicense\_col9 = sheet.cell(i, 9).value # 提取I列（第9列）的数据  
   
 # 验证B列的驾照编号  
 if drivelicense\_col2 and is\_valid\_drivelicense(drivelicense\_col2):  
 correct\_drivelicenses\_col2.append(drivelicense\_col2) # 将正确的驾照编号添加到列表中  
   
 # 验证I列的驾照编号  
 if drivelicense\_col9 and is\_valid\_drivelicense(drivelicense\_col9):  
 correct\_drivelicenses\_col9.append(drivelicense\_col9) # 将正确的驾照编号添加到列表中  
  
# 统计正确的驾照编号数量  
correct\_count\_col2 = len(correct\_drivelicenses\_col2)  
correct\_count\_col9 = len(correct\_drivelicenses\_col9)  
  
print("第二列（B列）中正确的驾照编号数量：", correct\_count\_col2)  
print("第二列（B列）中正确的驾照编号列表：", correct\_drivelicenses\_col2)  
  
print("第九列（I列）中正确的驾照编号数量：", correct\_count\_col9)  
print("第九列（I列）中正确的驾照编号列表：", correct\_drivelicenses\_col9)

符合身份证的B1列，也就是C2

## 银行卡号识别C3

from openpyxl import load\_workbook  
import re  
  
# 定义一个函数用于使用Luhn算法校验银行卡号的有效性  
def luhn\_checksum(card\_number):  
 def digits\_of(n):  
 return [int(d) for d in str(n)]  
 digits = digits\_of(card\_number)  
 odd\_digits = digits[-1::-2]  
 even\_digits = digits[-2::-2]  
 checksum = sum(odd\_digits)  
 for d in even\_digits:  
 checksum += sum(digits\_of(d \* 2))  
 return checksum % 10  
  
def is\_valid\_bankcard(bankcard):  
 # 验证长度和格式  
 if len(bankcard) != 19 or not bankcard.isdigit():  
 return False  
  
 # 验证BIN码是否在指定范围内  
 if not re.match(r"^(622481|622513|622250)", bankcard):  
 return False  
  
 # 使用Luhn算法验证校验位  
 return luhn\_checksum(bankcard) == 0  
  
# 加载Excel工作簿并选择第一个工作表  
wb = load\_workbook('test.xlsx') # 读取名为'test.xlsx'的Excel文件  
sheet = wb['Sheet1'] # 选择工作簿中的第一个工作表'Sheet1'  
  
correct\_bankcards\_col3 = [] # 创建一个空列表用于存储第三列（C列）中正确的银行卡号  
correct\_bankcards\_col8 = [] # 创建一个空列表用于存储第八列（H列）中正确的银行卡号  
  
# 从工作表中提取C1-C51和H1-H51列的数据并验证银行卡号  
for i in range(2, 52): # 遍历每列的行，从第2行到第51行  
 bankcard\_col3 = sheet.cell(i, 3).value # 提取C列（第3列）的数据  
 bankcard\_col8 = sheet.cell(i, 8).value # 提取H列（第8列）的数据  
   
 # 验证C列的银行卡号  
 if bankcard\_col3 and is\_valid\_bankcard(bankcard\_col3):  
 correct\_bankcards\_col3.append(bankcard\_col3) # 将正确的银行卡号添加到列表中  
   
 # 验证H列的银行卡号  
 if bankcard\_col8 and is\_valid\_bankcard(bankcard\_col8):  
 correct\_bankcards\_col8.append(bankcard\_col8) # 将正确的银行卡号添加到列表中  
  
# 统计正确的银行卡号数量  
correct\_count\_col3 = len(correct\_bankcards\_col3)  
correct\_count\_col8 = len(correct\_bankcards\_col8)  
  
print("第三列（C列）中正确的银行卡号数量：", correct\_count\_col3)  
print("第三列（C列）中正确的银行卡号列表：", correct\_bankcards\_col3)  
  
print("第八列（H列）中正确的银行卡号数量：", correct\_count\_col8)  
print("第八列（H列）中正确的银行卡号列表：", correct\_bankcards\_col8)

符合身份证的C1列，也就是C3

## mac地址识别C5

from openpyxl import load\_workbook  
import re  
  
# 定义一个函数用于校验MAC地址的有效性  
def is\_valid\_mac(mac):  
 # 验证长度和格式  
 if not re.match(r"^([0-9A-F]{2}:){5}[0-9A-F]{2}$", mac):  
 return False  
  
 # 验证前6个十六进制数对是否在指定范围内  
 if not re.match(r"^(AA:AD:7C|00:2D:4B|00:1A:5F)", mac):  
 return False  
  
 return True  
  
# 加载Excel工作簿并选择第一个工作表  
wb = load\_workbook('test.xlsx') # 读取名为'test.xlsx'的Excel文件  
sheet = wb['Sheet1'] # 选择工作簿中的第一个工作表'Sheet1'  
  
correct\_macs\_col5 = [] # 创建一个空列表用于存储第五列（E列）中正确的MAC地址  
correct\_macs\_col10 = [] # 创建一个空列表用于存储第十列（J列）中正确的MAC地址  
  
# 从工作表中提取E1-E51和J1-J51列的数据并验证MAC地址  
for i in range(2, 52): # 遍历每列的行，从第2行到第51行  
 mac\_col5 = sheet.cell(i, 5).value # 提取E列（第5列）的数据  
 mac\_col10 = sheet.cell(i, 10).value # 提取J列（第10列）的数据  
   
 # 验证E列的MAC地址  
 if mac\_col5 and is\_valid\_mac(mac\_col5):  
 correct\_macs\_col5.append(mac\_col5) # 将正确的MAC地址添加到列表中  
   
 # 验证J列的MAC地址  
 if mac\_col10 and is\_valid\_mac(mac\_col10):  
 correct\_macs\_col10.append(mac\_col10) # 将正确的MAC地址添加到列表中  
  
# 统计正确的MAC地址数量  
correct\_count\_col5 = len(correct\_macs\_col5)  
correct\_count\_col10 = len(correct\_macs\_col10)  
  
print("第五列（E列）中正确的MAC地址数量：", correct\_count\_col5)  
print("第五列（E列）中正确的MAC地址列表：", correct\_macs\_col5)  
  
print("第十列（J列）中正确的MAC地址数量：", correct\_count\_col10)  
print("第十列（J列）中正确的MAC地址列表：", correct\_macs\_col10)

符合MAC地址的E1列，也就是C5

## 邮箱识别C7

from openpyxl import load\_workbook  
import re  
  
# 定义一个函数用于校验邮箱地址的有效性  
def is\_valid\_email(email):  
 pattern = r"^[a-zA-Z][\da-zA-Z]\*\w+@(163\.com|qq\.com|hotmail\.com)$"  
 return re.match(pattern, email) is not None  
  
# 加载Excel工作簿并选择第一个工作表  
wb = load\_workbook('test.xlsx') # 替换为你的文件路径  
sheet = wb['Sheet1'] # 选择工作簿中的第一个工作表'Sheet1'  
  
correct\_emails\_col6 = [] # 创建一个空列表用于存储第六列（F列）中正确的邮箱地址  
correct\_emails\_col7 = [] # 创建一个空列表用于存储第七列（G列）中正确的邮箱地址  
  
# 从工作表中提取F1-F51和G1-G51列的数据并验证邮箱地址  
for i in range(2, 52): # 遍历每列的行，从第2行到第51行  
 email\_col6 = sheet.cell(i, 6).value # 提取F列（第6列）的数据  
 email\_col7 = sheet.cell(i, 7).value # 提取G列（第7列）的数据  
   
 # 验证F列的邮箱地址  
 if email\_col6 and is\_valid\_email(str(email\_col6).strip()):  
 correct\_emails\_col6.append(email\_col6.strip())  
   
 # 验证G列的邮箱地址  
 if email\_col7 and is\_valid\_email(str(email\_col7).strip()):  
 correct\_emails\_col7.append(email\_col7.strip())  
  
# 统计正确的邮箱地址数量  
correct\_count\_col6 = len(correct\_emails\_col6)  
correct\_count\_col7 = len(correct\_emails\_col7)  
  
print("第六列（F列）中正确的邮箱地址数量：", correct\_count\_col6)  
print("第六列（F列）中正确的邮箱地址列表：", correct\_emails\_col6)  
  
print("第七列（G列）中正确的邮箱地址数量：", correct\_count\_col7)  
print("第七列（G列）中正确的邮箱地址列表：", correct\_emails\_col7)

pattern = r"^[a-zA-Z][\da-zA-Z]\*\w+@(163\.com|qq\.com|hotmail\.com)$"

^:  
  
表示字符串的开始。确保匹配从字符串的开头开始。  
[a-zA-Z]:  
  
匹配任意一个字母（大小写均可）。这确保邮箱的用户名以一个字母开头。  
[\da-zA-Z]\*:  
  
\d匹配任意一个数字（0-9）。  
a-zA-Z匹配任意一个字母（大小写均可）。  
\*表示前面的模式（数字或字母）可以出现零次或多次。这确保在用户名的开头字母之后可以有任意数量的字母或数字。  
\w+:  
  
\w匹配任意一个字母、数字或下划线。  
+表示前面的模式（字母、数字或下划线）必须至少出现一次。这确保用户名包含至少一个字母、数字或下划线。  
@:  
  
匹配@符号，标识电子邮件的用户名和域名的分隔符。  
(163\.com|qq\.com|hotmail\.com)$:  
  
|表示或的关系，匹配163.com、qq.com或hotmail.com中的任意一个。  
\.匹配实际的点号.。  
$表示字符串的结束。确保匹配到字符串的末尾。

符合MAC地址的G列，也就是C7

## flag按照格式拼接

身份证c1

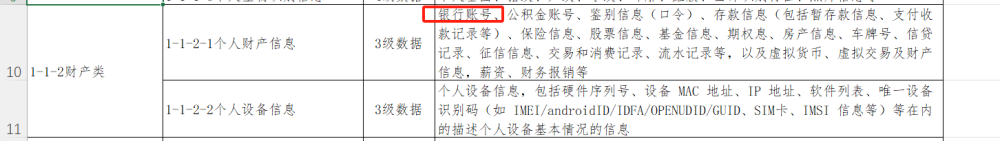
c1\_1-1-5-1\_3;

驾照c2

c2\_1-1-5-1\_3;

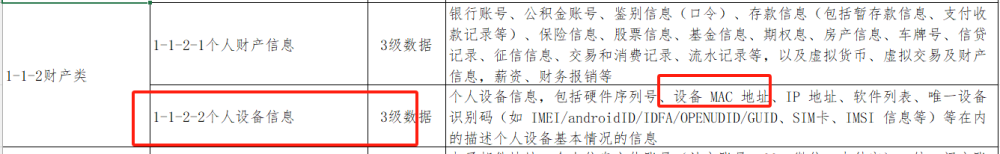
银行卡c3

c3\_1-1-2-1\_3;



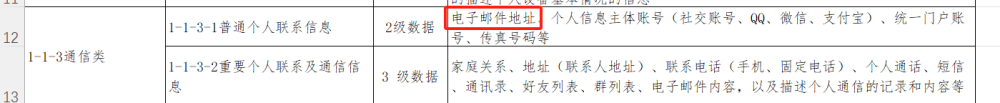
mac地址c5

c5\_1-1-2-2\_3;



邮箱c7

c7\_1-1-3-1\_2



c1\_1-1-5-1\_3;c2\_1-1-5-1\_3;c3\_1-1-2-1\_3;c5\_1-1-2-2\_3;c7\_1-1-3-1\_2;

<https://www.sojson.com/md5/>

最后的flag为

bd8c4ee3e25545b16eb3622378b2a95a