KONTAKT
Adrian Florek
adrian.florek@pwr.edu.pl

Część Praktyczna

1. **Napisz skrypt PowerShell, który monitoruje określony folder i automatycznie przenosi nowo dodane pliki .txt do innej lokalizacji.**

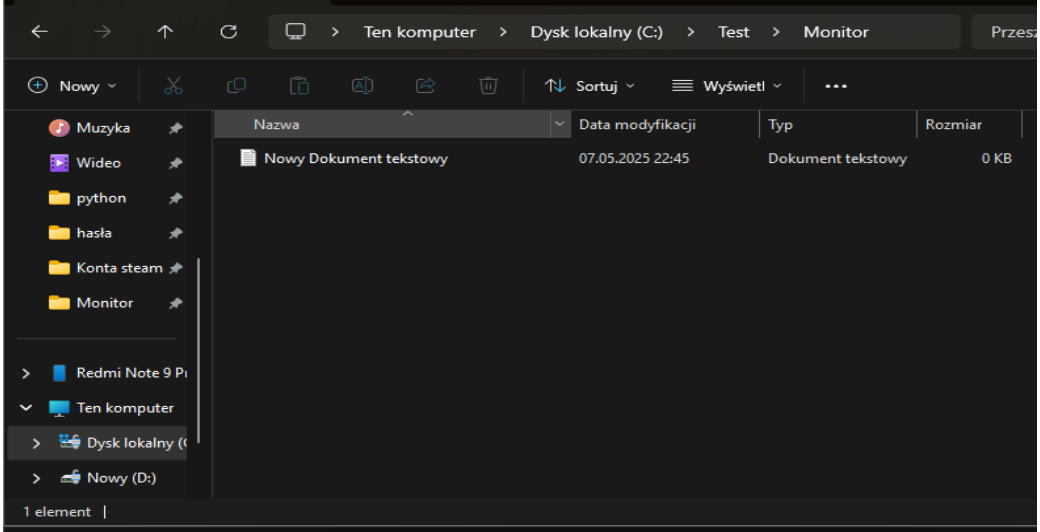
   a) Jeżeli folder docelowy nie istnieje skrypt musi go utworzyć.
   b) Skrypt powinien działać ciągle aż do jego ręcznego wyłączenia.

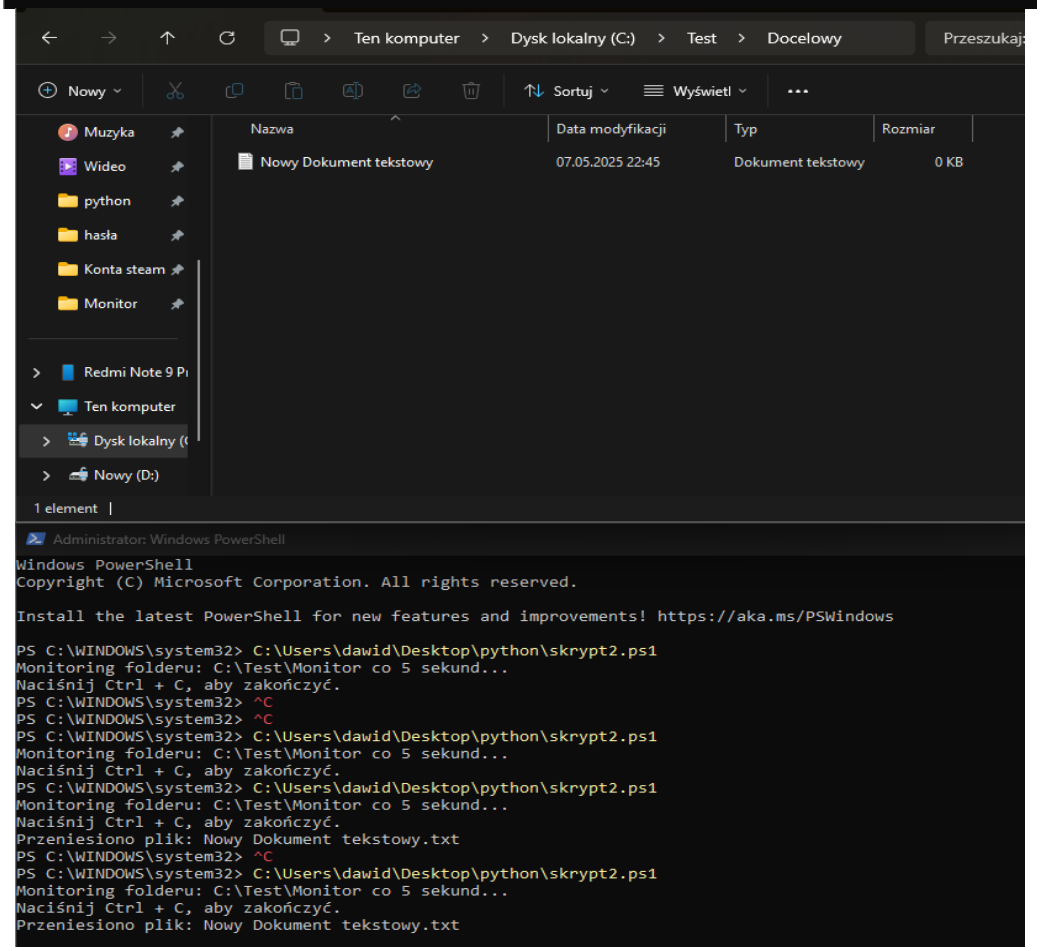
skrypt2.ps1

```powershell
$sourceFolder = "C:\Test\Monitor"
$destinationFolder = "C:\Test\Docelowy"
# Tworzymy folder docelowy, jeśli nie istnieje
if (-not (Test-Path -Path $destinationFolder)) {
    New-Item -ItemType Directory -Path $destinationFolder | Out-Null
    Write-Host "Utworzono folder docelowy: $destinationFolder"
}
Write-Host "Monitoring folderu: $sourceFolder co 5 sekund..."
Write-Host "Naciśnij Ctrl + C, aby zakończyć."
while ($true) {
    # Pobieramy pliki .txt z folderu źródłowego
    $files = Get-ChildItem -Path $sourceFolder -Filter *.txt -File -ErrorAction SilentlyContinue
    foreach ($file in $files) {
        $sourcePath = $file.FullName
        $destinationPath = Join-Path -Path $destinationFolder -ChildPath $file.Name
        Move-Item -Path $sourcePath -Destination $destinationPath -Force
        Write-Host "Przeniesiono plik: $($file.Name)"
    }

    Start-Sleep -Seconds 5
}

```





2. **Napisz skrypt w PowerShell który:**

a) Obliczy sumę kontrolną pliku (MD5 lub SHA256).
b) Wyśle zapytanie do API VirusTotal
c) Zinterpretuje odpowiedź API i wyświetli informację czy plik jest bezpieczny, czy nie.

Sprawdzić, czy wyniki są zgodne z oczekiwaniami - pobrać plik EICAR oraz utworzyć nowy plik testowy. Każdy etap skryptu powinien być opatrzony komentarzem

Totalvirus.ps1

```powershell
$apiKey = "ea92c7d9db127314acfbd1a4865104c8535491cabab8c7ec5155ceb5dc79b4d5"
# 1. Funkcja obliczająca sumę kontrolną pliku (MD5 lub SHA256)
Function Get-FileHashValue {
    param (
        [string]$filePath,
        [string]$hashAlgorithm = "SHA256"  # Domyślnie używamy SHA256, ale można zmienić na "MD5"
    )
    $hash = Get-FileHash -Path $filePath -Algorithm $hashAlgorithm
    return $hash.Hash
}
# 2. Funkcja wysyłająca zapytanie do API VirusTotal i sprawdzająca bezpieczeństwo pliku
Function Check-FileSafety {
    param (
        [string]$filePath,
        [string]$apiKey
    )
    $hashValue = Get-FileHashValue -filePath $filePath -hashAlgorithm "SHA256"
    Write-Host "Suma kontrolna pliku ($filePath): $hashValue"
    $virusTotalUrl = "https://www.virustotal.com/api/v3/files/$hashValue"
    try {
        $response = Invoke-RestMethod -Uri $virusTotalUrl -Headers @{ "x-apikey" = $apiKey } -Method Get
        if ($response.data.attributes.last_analysis_stats.malicious -gt 0) {
            Write-Host "Plik jest niebezpieczny"
        } else {
            Write-Host "Plik jest bezpieczny"
        }
    } catch {
        Write-Host "Błąd w zapytaniu do VirusTotal: $_"
    }
}

$fileToCheck = Read-Host "Podaj pełną ścieżkę do pliku."
if (-Not (Test-Path $fileToCheck)) {
    Write-Host "Błąd: Plik nie istnieje."
} else {
    Check-FileSafety -filePath $fileToCheck -apiKey $apiKey
}
```

```
PS C:\WINDOWS\system32> C:\Users\dawid\Desktop\python\Totalvirus.ps1
Podaj pełną ścieżkę do pliku.: C:\Test\eicar.txt
Suma kontrolna pliku (C:\Test\eicar.txt): 275A021BBFB6489E54D471899F7DB9D1663FC695EC2FE2A2C4538AABF651FD0F
Plik jest niebezpieczny
PS C:\WINDOWS\system32> C:\Users\dawid\Desktop\python\Totalvirus.ps1
Podaj pełną ścieżkę do pliku.: C:\Test\Nowy Dokument tekstowy.txt
Suma kontrolna pliku (C:\Test\Nowy Dokument tekstowy.txt): E3B0C44298FC1C149AFBF4C8996FB92427AE41E4649B934CA495991B7852B855
Plik jest bezpieczny
```