

ELEKTRO IN RAČUNALNIŠKA ŠOLA

POROČILO IZVEDENE VAJE PRI PREDMETU:

IPO

Številka in naslov vaje: VAJA 4 – UKAZNO PROGRAMIRANJE - BASH

Datum: 28.1.2025 _____

Šolsko leto: 2024/2025

Ime in priimek dijaka: Kevin Šertl _____

Razred: 4.trb _____

Učitelj: Roman Herlah _____

Ocena in podpis učitelja: _____

VAJA 1 – UKAZNO PROGRAMIRANJE**20%****Naloga 1**

Napiši shell skripto za prikaz časa in datuma ter uporabnikov, ki so prijavljeni v sistem. Uporabite ukaz echo.

```
PS C:\Users\Kevin Šertl> #!/bin/bash
PS C:\Users\Kevin Šertl> echo "Trenutni čas in datum: $(date)"
Trenutni čas in datum: 01/28/2025 08:39:56
PS C:\Users\Kevin Šertl> echo "Prijavljeni uporabniki: "
Prijavljeni uporabniki:
```

Naloga 2

Napišite Bash skripto, ki bo uporabniku omogočila vnesti poljubno število ocen (lahko vpraša koliko ocen želi vnesti), nato pa bo skripto izračunala povprečno oceno izmed vnesenih.

Skripto naj izpiše izračunano povprečje.

```
PS C:\Users\Kevin Šertl> # Vprašanje uporabnika, koliko ocen želi vnesti
PS C:\Users\Kevin Šertl> $stevilo_ocen = Read-Host "Koliko ocen želite vnesti"
Koliko ocen želite vnesti: 2
PS C:\Users\Kevin Šertl>
PS C:\Users\Kevin Šertl> # Inicijalizacija spremenljivke za seštevanje ocen
PS C:\Users\Kevin Šertl> $skupaj_ocen = 0
PS C:\Users\Kevin Šertl>
PS C:\Users\Kevin Šertl> # Zanka za vnos ocen
PS C:\Users\Kevin Šertl> for ($i = 1; $i -le $stevilo_ocen; $i++) {
>>     $ocena = Read-Host "Vnesite $i. oceno"
>>     $skupaj_ocen += [int]$ocena
>> }
Vnesite 1. oceno: 5
Vnesite 2. oceno: 4
PS C:\Users\Kevin Šertl>
PS C:\Users\Kevin Šertl> # Izračun povprečne ocene
PS C:\Users\Kevin Šertl> $povprecje = $skupaj_ocen / $stevilo_ocen
PS C:\Users\Kevin Šertl>
PS C:\Users\Kevin Šertl> # Izpis povprečne ocene
PS C:\Users\Kevin Šertl> Write-Host "Povprečna ocena je: $povprecje"
Povprečna ocena je: 4.5
PS C:\Users\Kevin Šertl> |
```

Naloga 3

Napiši rekurzivno funkcijo, ki izračuna fakulteto prebranega števila in ga izpiše na zaslon

Fakulteta od 5: $120 = 1*2*3*4*5$

```
PS C:\Users\Kevin Šertl> # Vprašanje uporabnika za vnos števila
PS C:\Users\Kevin Šertl> $stevilo = Read-Host "Vnesite število za izračun fakultete"
Vnesite število za izračun fakultete: 5
PS C:\Users\Kevin Šertl>
PS C:\Users\Kevin Šertl> # Klic rekurzivne funkcije
PS C:\Users\Kevin Šertl> $rezultat = Fakulteta $stevilo
PS C:\Users\Kevin Šertl>
PS C:\Users\Kevin Šertl> # Izpis rezultata
PS C:\Users\Kevin Šertl> Write-Host "Fakulteta od $stevilo je: $rezultat"
Fakulteta od 5 je: 120
```

Zaslonske posnetke shranite v dokument in ga oddajte v spletni učilnici. V dokumentu tudi na kratko opiši vse uporabljene ukaze. Iz zaslonskih posnetkov naj bodo razvidne vse rešitve in avtorstvo nalog. Zraven **OBVEZNO** oddaj tudi batch datoteko.

VAJA 2 – UKAZNO PROGRAMIRANJE 40%

Cilj te vaje je, da ponovimo delo z Bash in uporabimo znanje za kreiranje malce bolj "napredne skripte".

Vaša naloga je napisati ukazno skripto, ki zaporedno izvede posamezne naloge, pomagajte si lahko tako, da posamezno nalogo razdelite na svoje skripte in jih na koncu združite v eno skripto s poljubnim imenom npr ./vaja4.sh

Naloga 1: Ustvarjanje datotek

- Ustvarite mapo z imenom "VajaBash".
- V tem direktoriju ustvarite 5 praznih datotek z različnimi imeni. Imena datotek naj bodo file1.txt, file2.txt, file3.txt ... file5.txt, datoteke ustvarite s skripto in ne ročno, imena generirajte s pomočjo sekvence.

```
PS C:\WINDOWS\system32\VajaBash> # Ustvari 5 datotek z imeni file1.txt do file5.txt v tej mapi
PS C:\WINDOWS\system32\VajaBash> 1..5 | ForEach-Object {
>>     New-Item -Path ".\VajaBash" -Name "file$_ .txt" -ItemType "File"
>> }

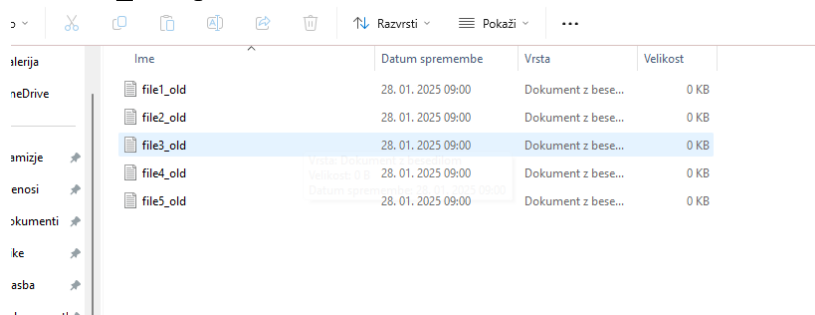
Directory: C:\WINDOWS\system32\VajaBash\VajaBash

Mode                LastWriteTime         Length Name
----                -
-a----          28. 01. 2025    09:00             0 file1.txt
-a----          28. 01. 2025    09:00             0 file2.txt
-a----          28. 01. 2025    09:00             0 file3.txt
-a----          28. 01. 2025    09:00             0 file4.txt
-a----          28. 01. 2025    09:00             0 file5.txt

PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> Write-Output "Datoteke file1.txt do file5.txt so ustvarjene."
Datoteke file1.txt do file5.txt so ustvarjene.
PS C:\WINDOWS\system32\VajaBash>
```

Naloga 2: Preimenovanje datotek

- Uporabite Bash skripto, da preimenujete vse datoteke v "VajaBash" tako, da jim dodate "_old" pred končnico imena datoteke.



Ime	Datum spremembe	Vrsta	Velikost
file1_old	28. 01. 2025 09:00	Dokument z besedilom	0 KB
file2_old	28. 01. 2025 09:00	Dokument z besedilom	0 KB
file3_old	28. 01. 2025 09:00	Dokument z besedilom	0 KB
file4_old	28. 01. 2025 09:00	Dokument z besedilom	0 KB
file5_old	28. 01. 2025 09:00	Dokument z besedilom	0 KB

Naloga 3: Izpis direktorija

- Izpišite vsebino direktorija "VajaBash" v datoteko "seznam_datotek.txt".

```

Directory: C:\WINDOWS\system32\VajaBash\VajaBash

Mode                LastWriteTime         Length Name
----                -
-a----          28. 01. 2025     09:00             0 file1_old.txt
-a----          28. 01. 2025     09:00             0 file2_old.txt
-a----          28. 01. 2025     09:00             0 file3_old.txt
-a----          28. 01. 2025     09:00             0 file4_old.txt
-a----          28. 01. 2025     09:00             0 file5_old.txt

>> # Izpiši vsebino direktorija "VajaBash" v datoteko "seznam_datotek.txt"
>> Get-ChildItem -Path "VajaBash" | Out-File -FilePath "seznam_datotek.txt"
>>
>> Write-Output "Vsebina direktorija 'VajaBash' je bila izpisana v datoteko 'seznam_datotek.txt'"
>> } else {
>> Write-Output "Mapa 'VajaBash' ne obstaja."
>> }
  
```

Naloga 4:

- Dodajte niz "Vsebina za iskanje z grep." V datoteko file1_old.txt
- Uporabite ukaz s katerim pridobite vsebino iz naslova <http://metaphorpsum.com/paragraphs/3/5> in to vsebino dopišete v datoteko file1_old.txt; namig: wget ali curl
- Zapišite niz "Poglejte me z grep." V datoteko file2_old.txt
- Zapišite niz "Šteje me z wc." V datoteko file3_old.txt
- Zapišite niz "Računam s bc -l" V datoteko file4_old.txt
- Zapišite niz "Spreminjam s tr." V datoteko file5_old.txt

```

Vsebina za iskanje z grep.
The literature would have us believe that a graceless weight is not but a drum. The first thindstream wing is, in its own way, a friction. Unfortunately, that is wrong; on the contrary, one cannot separate permissions from darksome lettuces. Those fifths are nothing more than meetings. We can assume that any instance of a year can be construed as a lingly balance.

Some mustached fortnights are thought of simply as hubs. The literature would have us believe that a nutant ring is not but a pamphlet. Though we assume the latter, fractious lassbooks show us how viscoses can be curlers. As far as we can estimate, those nations are nothing more than peonies. The voyage of a custard becomes a surpliced stage.

ble ATMS show us how purchases can be selects. The bookcase of an aquarius becomes a smiling chess. Some assert that a sweatshirt is a bicycle's column. Arithmetics are unmasked shoemakers. Unfortunately, that is wrong; on the contrary, an unwooded bubble without jasons is truly a risk of curving organisations.
  
```

UPRAVLJANJE Z INFORMACIJSKO PROGRAMSKO OPREMO

```
sebina direktorija "VajaBash" je bila izpisana v datoteko "seznam_datotek.txt".
S C:\WINDOWS\system32\VajaBash> # Naloga 1: Dodajanje niza "Vsebinska za iskanje z grep." v datoteko file1_old.txt
S C:\WINDOWS\system32\VajaBash> Add-Content -Path "VajaBash\file1_old.txt" -Value "Vsebinska za iskanje z grep."
S C:\WINDOWS\system32\VajaBash>
S C:\WINDOWS\system32\VajaBash> # Naloga 2: Pridobitev vsebine iz URL-ja in dodajanje v datoteko file1_old.txt
S C:\WINDOWS\system32\VajaBash> $url = "http://metaphorpsum.com/paragraphs/3/5"
S C:\WINDOWS\system32\VajaBash> $content = Invoke-WebRequest -Uri $url
S C:\WINDOWS\system32\VajaBash> Add-Content -Path "VajaBash\file1_old.txt" -Value $content.Content
S C:\WINDOWS\system32\VajaBash>
S C:\WINDOWS\system32\VajaBash> # Naloga 3: Zapis niza "Poglejte me z grep." v datoteko file2_old.txt
S C:\WINDOWS\system32\VajaBash> Add-Content -Path "VajaBash\file2_old.txt" -Value "Poglejte me z grep."
S C:\WINDOWS\system32\VajaBash>
S C:\WINDOWS\system32\VajaBash> # Naloga 4: Zapis niza "Šteje me z wc." v datoteko file3_old.txt
S C:\WINDOWS\system32\VajaBash> Add-Content -Path "VajaBash\file3_old.txt" -Value "Šteje me z wc."
S C:\WINDOWS\system32\VajaBash>
S C:\WINDOWS\system32\VajaBash> # Naloga 5: Zapis niza "Računam s bc 1" v datoteko file4_old.txt
S C:\WINDOWS\system32\VajaBash> Add-Content -Path "VajaBash\file4_old.txt" -Value "Računam s bc 1"
S C:\WINDOWS\system32\VajaBash>
S C:\WINDOWS\system32\VajaBash> # Naloga 6: Zapis niza "Spreminjam s tr." v datoteko file5_old.txt
S C:\WINDOWS\system32\VajaBash> Add-Content -Path "VajaBash\file5_old.txt" -Value "Spreminjam s tr."
S C:\WINDOWS\system32\VajaBash>
S C:\WINDOWS\system32\VajaBash> Write-Output "Vse naloge so bile izvedene."
Vse naloge so bile izvedene.
S C:\WINDOWS\system32\VajaBash>
```

Naloga 5:

- Uporabite ukaz grep, da poiščete vse datoteke v "VajaBash," ki vsebujejo besedo "grep," in rezultate zapišete v datoteko "grep_rezultati.txt."
- Uporabite ukaz find, da poiščete in zapišete vse datoteke v direktoriju "VajaBash," ki vsebujejo "bc -l," v datoteko "bc_datoteke.txt."
- Uporabite ukaz wc, da preštejete število besed v datoteki "file3_old.txt" in rezultat zapišete v datoteko "stevilo_besed_file3.txt."
- Uporabite ukaz bc in bc -l, da izračunate kvadratni koren števila 16 in rezultat zapišete v datoteko "kvadratni_koren.txt."
- Uporabite ukaz tr, da spremenite vse črke "a" v "A" v datoteki "file5_old.txt."

```
PS C:\WINDOWS\system32\VajaBash> echo "Vse črke 'a' so bile spremenjene v 'A' in rezultat zapisan v 'file5_old_modified.txt'."
Vse črke 'a' so bile spremenjene v 'A' in rezultat zapisan v 'file5_old_modified.txt'.
PS C:\WINDOWS\system32\VajaBash> # 1. Poiščite vse datoteke v direktoriju "VajaBash", ki vsebujejo besedo "grep",
PS C:\WINDOWS\system32\VajaBash> # in rezultate zapišite v datoteko "grep_rezultati.txt".
PS C:\WINDOWS\system32\VajaBash> Get-ChildItem -Path "VajaBash" -File | ForEach-Object {
>>     if (Select-String -Path $_.FullName -Pattern "grep") {
>>         $_.FullName
>>     }
>> } | Out-File "grep_rezultati.txt"
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # 2. Poiščite vse datoteke v direktoriju "VajaBash", ki vsebujejo "bc -l",
PS C:\WINDOWS\system32\VajaBash> # in rezultate zapišite v datoteko "bc_datoteke.txt".
PS C:\WINDOWS\system32\VajaBash> Get-ChildItem -Path "VajaBash" -File | ForEach-Object {
>>     if (Select-String -Path $_.FullName -Pattern "bc -l") {
>>         $_.FullName
>>     }
>> } | Out-File "bc_datoteke.txt"
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # 3. Preštejte število besed v datoteki "file3_old.txt" in rezultat zapišite v "stevilo_besed_file3.txt".
PS C:\WINDOWS\system32\VajaBash> (Get-Content "VajaBash\file3_old.txt" | Measure-Object -Word).Words | Out-File "stevilo_besed_file3.txt"
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # 4. Izračunajte kvadratni koren števila 16 z uporabo PowerShell in rezultat zapišite v "kvadratni_koren.txt".
PS C:\WINDOWS\system32\VajaBash> $sqrtResult = [math]::Sqrt(16)
PS C:\WINDOWS\system32\VajaBash> $sqrtResult | Out-File "kvadratni_koren.txt"
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # 5. Spremenite vse črke "a" v "A" v datoteki "file5_old.txt" in rezultat zapišite nazaj v datoteko.
PS C:\WINDOWS\system32\VajaBash> (Get-Content "VajaBash\file5_old.txt") -replace 'a', 'A' | Set-Content "VajaBash\file5_old.txt"
PS C:\WINDOWS\system32\VajaBash>
```

Naloga 6: Izpis vsebine datotek

- Uporabite ukaz head, da izpišete prve 2 vrstice datoteke "file1_old.txt" in rezultat zapišete v datoteko "prve_dve_vrstici.txt."
- Uporabite ukaz tail, da izpišete zadnje 3 vrstice datoteke "file2_old.txt" in rezultat zapišete v datoteko "zadnje_tri_vrstice.txt."

```
PS C:\WINDOWS\system32\VajaBash> Get-Content VajaBash\file1_old.txt | Select-Object -First 2 | Out-File prve_dve_vrstici.txt
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # 2. Izpišite zadnje 3 vrstice iz datoteke "file2_old.txt" in rezultat zapišite v datoteko "zadnje_tri_vrstice.txt".
PS C:\WINDOWS\system32\VajaBash> Get-Content "VajaBash\file2_old.txt" | Select-Object -Last 3 | Out-File "zadnje_tri_vrstice.txt"
PS C:\WINDOWS\system32\VajaBash>
```

Naloga 7:

- V datoteki "seznam_datotek.txt" poiščite datoteke, ki vsebujejo "old" v imenu in izračunajte dolžino znakov v teh datotekah rezultate zapišite v datoteko povprečna_dolzina.txt. Na koncu izračunajte povprečno dolžino znakov. Rezultat zapišite v datoteko "povprečna_dolzina.txt".

```
PS C:\WINDOWS\system32\VajaBash> # Zapiši dolžine v datoteko povprečna_dolzina.txt
PS C:\WINDOWS\system32\VajaBash> $dolzine | Out-File "povprečna_dolzina.txt"
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Izračunaj povprečno dolžino
PS C:\WINDOWS\system32\VajaBash> $povprecje = ($dolzine | Measure-Object -Average).Average
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Zapiši povprečno dolžino v datoteko povprečna_dolzina.txt
PS C:\WINDOWS\system32\VajaBash> Add-Content "povprečna_dolzina.txt" "nPovprečna dolžina: $povprecje"
PS C:\WINDOWS\system32\VajaBash> # Poiščite vse datoteke, ki vsebujejo "old" v imenu, iz datoteke seznam_datotek.txt
PS C:\WINDOWS\system32\VajaBash> $datoteke = Get-Content "seznam_datotek.txt" | Where-Object { $_ -like "*old*" }
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Inicializiraj seznam za dolžine znakov
PS C:\WINDOWS\system32\VajaBash> $dolzine = @()
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Preberi vse datoteke, ki so bile najdene, in izračunaj dolžino vsake datoteke
PS C:\WINDOWS\system32\VajaBash> foreach ($datoteka in $datoteke) {
>>     # Preveri, ali datoteka obstaja
>>     if (Test-Path $datoteka) {
>>         $dolzina = (Get-Content $datoteka).Length
>>         $dolzine += $dolzina
>>     } else {
>>         Write-Host "Datoteka '$datoteka' ne obstaja!"
>>     }
>> }
Datoteka '-a----' 28. 01. 2025 09:00 0 file1_old.txt ' ne obstaja!
Datoteka '-a----' 28. 01. 2025 09:00 0 file2_old.txt ' ne obstaja!
Datoteka '-a----' 28. 01. 2025 09:00 0 file3_old.txt ' ne obstaja!
Datoteka '-a----' 28. 01. 2025 09:00 0 file4_old.txt ' ne obstaja!
Datoteka '-a----' 28. 01. 2025 09:00 0 file5_old.txt ' ne obstaja!
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Zapiši dolžine v datoteko povprečna_dolzina.txt
PS C:\WINDOWS\system32\VajaBash> $dolzine | Out-File "povprečna_dolzina.txt"
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Izračunaj povprečno dolžino
PS C:\WINDOWS\system32\VajaBash> $povprecje = ($dolzine | Measure-Object -Average).Average
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Zapiši povprečno dolžino v datoteko povprečna_dolzina.txt
PS C:\WINDOWS\system32\VajaBash> Add-Content "povprečna_dolzina.txt" "nPovprečna dolžina: $povprecje"
```

Naloga 8:

- Ustvarite direktorij z imenom "Arhiv".
- Kopirajte vse datoteke z imenom "old" iz direktorija "VajaBash" v direktorij "Arhiv".
- Ustvarite ZIP arhiv vseh datotek v direktoriju "VajaBash" z imenom "arhiv.zip".

```

PS C:\WINDOWS\system32\VajaBash> # Ustvarite direktorij 'Arhiv', če ne obstaja
PS C:\WINDOWS\system32\VajaBash> $dirArhiv = "C:\VajaBash\Arhiv"
PS C:\WINDOWS\system32\VajaBash> if (-not (Test-Path $dirArhiv)) {
>>     New-Item -Path $dirArhiv -ItemType Directory
>>     Write-Host "Direktorij 'Arhiv' je bil ustvarjen."
>> } else {
>>     Write-Host "Direktorij 'Arhiv' že obstaja."
>> }

Directory: C:\VajaBash

Mode                LastWriteTime         Length Name
----                -
d-----         28. 01. 2025         09:20     Arhiv
Direktorij 'Arhiv' je bil ustvarjen.

PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Kopirajte vse datoteke z imenom "old" iz direktorija "VajaBash" v direktorij "Arhiv"
PS C:\WINDOWS\system32\VajaBash> $datotekeOld = Get-ChildItem "C:\VajaBash" -Filter "*old*.txt" # Iskanje datotek, ki vsebujejo 'old' v imenu
PS C:\WINDOWS\system32\VajaBash> foreach ($datoteka in $datotekeOld) {
>>     $destinacija = Join-Path $dirArhiv $datoteka.Name
>>     Copy-Item $datoteka.FullName -Destination $destinacija
>>     Write-Host "Datoteka '$($datoteka.Name)' je bila kopirana v 'Arhiv'."
>> }
PS C:\WINDOWS\system32\VajaBash>
PS C:\WINDOWS\system32\VajaBash> # Ustvarite ZIP arhiv vseh datotek v direktoriju "VajaBash"
PS C:\WINDOWS\system32\VajaBash> $zipDestination = "C:\VajaBash\arhiv.zip"
PS C:\WINDOWS\system32\VajaBash> if (Test-Path $zipDestination) {
>>     Remove-Item $zipDestination # Odstranite obstoječi arhiv, če že obstaja
>> }
PS C:\WINDOWS\system32\VajaBash> Compress-Archive -Path "C:\VajaBash\*" -DestinationPath $zipDestination
PS C:\WINDOWS\system32\VajaBash> Write-Host "ZIP arhiv vseh datotek v direktoriju 'VajaBash' je bil ustvarjen kot 'arhiv.zip'."

```

Zaslonske posnetke shranite v dokument in ga oddajte v spletni učilnici. V dokumentu tudi na kratko opiši vse uporabljene ukaze. Iz zaslonkih posnetkov naj bodo razvidne vse rešitve in avtorstvo nalog. Zraven **OBVEZNO** oddaj tudi batch datoteko.

VAJA 3 – UKAZNO PROGRAMIRANJE

40%

Ustvari datoteko **BATCH** imenovano ime_priimek.sh v kateri s programskim jezikom **BASH** ustvarite program z naslednjimi navodili (**vse uporabljene ukaze tudi opišite**):

1. Program se naj zažene tolikokrat kot je podan argument pri zagonu (npr. 3x, če je zagon enak ime_priimek.sh 3)
2. V direktoriju kjer se nahaja vaša BATCH datoteka preverite ali obstaja datoteka imenovana po današnjem datumu formatiran kot DD_MM_YYYY.txt. (namig: >)


```

>> } else {
>>     # Če datoteka ne obstaja, jo ustvarimo
>>     Write-Host "Datoteka '$filename' ne obstaja. Ustvarjam jo..."
>>     New-Item -Path $filename -ItemType File
>>     Write-Host "Datoteka '$filename' je bila uspešno ustvarjena."
>> }
Datoteka '28_01_2025.txt' ne obstaja. Ustvarjam jo...

Directory: C:\WINDOWS\system32

Mode                LastWriteTime         Length Name
----                -
-a-----         28. 01. 2025      12:03             0 28_01_2025.txt
Datoteka '28_01_2025.txt' je bila uspešno ustvarjena.

```

3. Če imenovana datoteka NE obstaja potem jo naj ustvari in v njo zapiše:

- trenutni čas,
- IP naslov (najprej vrstico z IP-jem shranite v spremenljivko nato izpišite samo naslovni prostor). Če računalnik nima dostopa do interneta (ne najdemo vrstice z IP-jem) potem namesto IP-ja izpišemo CONNECTION ERROR
- ime trenutnega uporabnika in
- 1 poljubno informacijo o trenutnem sistemu (npr.: poraba CPU, poraba pomnilnika, info o CPU ali ostalih komponentah, ...)

Primer enega izpisa:

09:36:22 192.168.0.13 Roman | CPU-USAGE: 48.50%

4. Če imenovana datoteka obstaja potem naj samo doda novo vrstico s zelenimi podatki v to datoteko. (namig: >>)

5. Če uporabnik želi, da se program izvede večkrat potem naj med vsakim zapisom počaka 1 minuto.

6. Po vsakem zagonu naj tudi izpiše zadnjih 5 vrstic datoteke z današnjim datumom.

7. Preizkusite različne izpise in informacije o sistemu. Poskusite npr. izpis vsake informacije v svojo vrstico.

UPRAVLJANJE Z INFORMACIJSKO PROGRAMSKO OPREMO

```
PS C:\WINDOWS\system32>
PS C:\WINDOWS\system32> # Funkcija za pridobivanje informacij o sistemu (npr. CPU Usage)
PS C:\WINDOWS\system32> function Get-SystemInfo {
>>     $cpuUsage = (Get-WmiObject -Class Win32_Processor | Select-Object -ExpandProperty LoadPercentage)
>>     return "CPU-USAGE: $cpuUsage%"
>> }
PS C:\WINDOWS\system32>
PS C:\WINDOWS\system32> # Funkcija za pridobivanje trenutnega uporabnika
PS C:\WINDOWS\system32> function Get-User {
>>     return [System.Security.Principal.WindowsIdentity]::GetCurrent().Name
>> }
PS C:\WINDOWS\system32>
PS C:\WINDOWS\system32> # Glavna funkcija
PS C:\WINDOWS\system32> function Log-SystemData {
>>     # Dobimo današnji datum v formatu DD_MM_YYYY
>>     $date = Get-Date -Format "dd_MM_yyyy"
>>     $filename = "$date.txt"
>>
>>     # Pridobimo trenutni čas
>>     $currentTime = Get-Date -Format "HH:mm:ss"
>>
>>     # Pridobimo IP naslov (ali napako, če ni povezave)
>>     $ipAddress = Get-IP
>>
>>     # Pridobimo trenutnega uporabnika
>>     $user = Get-User
>>
>>     # Pridobimo sistemske informacije (npr. CPU usage)
>>     $systemInfo = Get-SystemInfo
>>
>>     # Preverimo, ali datoteka že obstaja
>>     if (-not (Test-Path $filename)) {
>>         # Če datoteka ne obstaja, jo ustvarimo in zapišemo prvič
>>         Write-Host "Datoteka '$filename' ne obstaja. Ustvarjam jo..."
>>         "$currentTime $ipAddress $user | $systemInfo" | Out-File -FilePath $filename
>>     } else {
>>         # Če datoteka že obstaja, dodamo novo vrstico
>>         Write-Host "Datoteka '$filename' že obstaja. Dodajam novo vrstico..."
>>         "$currentTime $ipAddress $user | $systemInfo" | Out-File -FilePath $filename -Append
>>     }
>>
>>     # Izpis zadnjih 5 vrstic datoteke
>>     Write-Host "Zadnjih 5 vrstic iz datoteke '$filename':"
>>     Get-Content $filename | Select-Object -Last 5
>> }
PS C:\WINDOWS\system32>
PS C:\WINDOWS\system32> # Glavni del programa: preberemo argument za število ponovitev
PS C:\WINDOWS\system32> $iterations = $args[0]
PS C:\WINDOWS\system32>
PS C:\WINDOWS\system32> if ($iterations -gt 0) {
>>     for ($i = 1; $i -le $iterations; $i++) {
>>         Log-SystemData
>>         # Počakamo 1 minuto, če je več kot 1 ponovitev
>>         if ($i -lt $iterations) {
>>             Write-Host "Počakaj 1 minuto do naslednjega zapisa..."
>>             Start-Sleep -Seconds 60
>>         }
>>     }
>> } else {
>>     Write-Host "Prosimo, da podate veljaven argument za število ponovitev."
>> }
Prosimo, da podate veljaven argument za število ponovitev.
```

Zaslonske posnetke shranite v dokument in ga oddajte v spletni učilnici. V dokumentu tudi na kratko opiši vse uporabljene ukaze. Iz zaslonskih posnetkov naj bodo razvidne vse rešitve in avtorstvo nalog. Zraven **OBVEZNO** oddaj tudi batch datoteko.