



VILNIAUS UNIVERSITETAS  
MATEMATIKOS IR INFORMATIKOS FAKULTETAS  
INFORMACINIŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

**PD nr. 3. Susipažinimas su Linux OS Bash ir Windows OS  
PowerShell komandinėmis eilutėmis**

Praktinio darbo ataskaita

Atliko: Monika Mirbakaitė

VU el. p.: [monika.mirbakaite@mif.stud.vu.lt](mailto:monika.mirbakaite@mif.stud.vu.lt),

Vertino: Vyresnysis lektorius Aleksandr  
Igumenov

Vilnius

2024

# TURINYS

TURINYS.....	3
1. Naudojant komandą who nukreipkite jos įvykdymo rezultatą į failą „Jūsų_Vardas“. Paleiskite komandą more failo „Jūsų_Vardas“ peržiūrai.....	5
LINUX.....	5
WINDOWS .....	5
2. Panaudokite komandas date ir who vienu metu (vienoje eilutėje) taip, kad date išvedimas atsirastu Jūsų ekrane, o who nukreiptas į failą „Jūsų_Pavardė“. Patikrinkite failo „Jūsų_Pavardė“ turinį su komanda more. ....	5
LINUX.....	5
WINDOWS .....	5
3. Raskite informaciją apie komandą sed ir surašykite ją taip kad ji apkeistu kiekvienoje eilutėje pirmą ir antrą žodį parinktame faile. ....	6
LINUX.....	6
WINDOWS .....	6
4. Sukurkite dvi programas, vieną scenarijų ir pamatuokite jų veikimo laiką su komanda time:.....	6
4.1. Pirma programa – tai shell (PowerShell)scenarijus išvedantis į ekraną Jūsų vardą ir pavardę. .6	
LINUX.....	6
WINDOWS .....	7
4.2. Antra programa – tai C programa išvedantis į ekraną Jūsų vardą ir pavardę.....	7
LINUX.....	7
WINDOWS .....	8
4.3. Trečia programa – tai C programa su MPI arba OMP bibliotekos panaudojimu, apskaičiuokite visas galimas funkcijos reikšmes ir pateikite kaip rezultatą funkcijos. Išmatuokite rezultato apskaičiavimo greitį su 1, 2, 3, 4 ... MAX (Log_CPU_NR) procesoriaus branduoliais, sudarykite greitaiveikos kreivę. ....	9
LINUX.....	9
WINDOWS .....	9
5. Sukurkite scenarijų kuris priimtu komandinės eilutės argumentą ir pateiktu informaciją kas tai yra: failo vardas, direktorijos vardas, arba kažkas tai kito. ....	9
LINUX.....	9
WINDOWS .....	9
6. Sukurkite scenarijų kuriam perduodami dvejų ar daugiau failų vardai. Scenarijus turi pervardinti juos į tuos pačius, tik pavadinimuose turės būti visos didelės raidės (jeigu tokie dar neegzistuoja darbo direktorijoje).....	10
LINUX.....	10
WINDOWS .....	11
7. Sukurkite scenarijų kuris nustatytu kiek laiko naudotojas dirba sistemoje (naudotojo vardas perduodamas kaip parametras). ....	11
LINUX.....	12

WINDOWS .....	12
8. Sukurkite scenarijų kuris kaip parametą priimtu failo vardą, pradžios ir pabaigos eilučių numerius ir išveda į ekraną failo turinį tarp nurodytų eilučių. ....	13
LINUX.....	13
WINDOWS .....	14
9. Sukurkite scenarijų trinantį visus failus aplankale turinčius žodį perduotą jam kaip parametą. ....	14
LINUX.....	14
WINDOWS .....	15
10. Sukurkite scenarijų kuriam perduodamas tekstinio failo vardas. Scenarijus turi atlikti tokius veiksmus: Apjungti kas dvi eilutes į vieną. Apskaičiuoti naujų eilučių ilgį ir išvesti šią informaciją į ekraną.	
LINUX.....	17
WINDOWS .....	18

## 1. NAUDOJANT KOMANDĄ WHO NUKREIPKITE JOS ĮVYKDYMO REZULTATĄ Į FAILĄ „JŪSŲ\_VARDAS“. PALEISKITE KOMANDĄ MORE FAILO „JŪSŲ\_VARDAS“ PERŽIŪRAI.

1 pav. vaizduojamos who ir more komandos Linux operacinėje sistemoje, o 2 pav. vaizduojami who ir more komandos atitikmenys Windows operacinėje sistemoje.

### LINUX

```
[admin@openmandriva-x8664 ~]$ who > monika
[admin@openmandriva-x8664 ~]$ more monika
admin    tty1          2024-05-04 14:03
admin    pts/0         2024-05-04 14:03 (:0)
admin    pts/1         2024-05-04 14:05 (:0)
```

1 pav. Monika peržiūra (Linux OS)

### WINDOWS

```
C:\Windows\system32>query user > monika

C:\Windows\system32>more monika
USERNAME                SESSIONNAME              ID  STATE  IDLE TIME  LOGON TIME
>monika                  console                  1   Active  none       5/21/2024 4:39 PM
```

2 pav. Monika peržiūra (Windows OS)

## 2. PANAUDOKITE KOMANDAS DATE IR WHO VIENU METU (VIENOJE EILUTĖJE) TAIP, KAD DATE IŠVEDIMAS ATSIKRASTU JŪSŲ EKRANE, O WHO NUKREIPTAS Į FAILĄ „JŪSŲ\_PAVARDĖ“. PATIKRINKITE FAILO „JŪSŲ\_PAVARDĖ“ TURINĮ SU KOMANDA MORE.

3 pav. vaizduojamas date ir who komandų atlikimas Linux operacinėje sistemoje, o 4 pav. vaizduojami date ir who komandų atitikmenys Windows operacinėje sistemoje.

### LINUX

```
[admin@openmandriva-x8664 ~]$ (date && who) > mirbakaite
[admin@openmandriva-x8664 ~]$ more mirbakaite
Sat  4 May 14:16:30 BST 2024
admin    tty1          2024-05-04 14:03
admin    pts/0         2024-05-04 14:03 (:0)
admin    pts/1         2024-05-04 14:05 (:0)
```

3 pav. Mirbakaite peržiūra (Linux OS)

### WINDOWS

```
C:\Windows\system32>date /T & query user > mirbakaite
Tue 05/21/2024

C:\Windows\system32>more mirbakaite
USERNAME                SESSIONNAME              ID  STATE  IDLE TIME  LOGON TIME
>monika                  console                  1   Active  none       5/21/2024 4:39 PM
```

4 pav. Mirbakaite peržiūra (Windows OS)

### 3. RASKITE INFORMACIJĄ APIE KOMANDĄ SED IR SURAŠYKITE JĄ TAIP KAD JŲ APKEISTU KIEKVIENOJE EILUTĖJE PIRMĄ IR ANTRĄ ŽODĮ PARINKTAME FAILE.

5 pav. vaizduojama sed komanda Linux operacinėje sistemoje, o 6 pav. ir 7 pav. vaizduojamas sed komandos atitikmuo Windows operacinėje sistemoje.

#### LINUX

```
[admin@openmandriva-x8664 ~]$ cat test
word switch 1
word switch 2
word switch 3

[admin@openmandriva-x8664 ~]$ sed 's/\([^[:space:]]*\)[[:space:]]*\([^[:space:]]*\)/\2 \1/' test
switch word 1
switch word 2
switch word 3
```

5 pav. komanda sed (Linux OS)

#### WINDOWS

```
C:\Windows\system32>query user > test.txt

C:\Windows\system32>more test.txt
word switch 1
word switch 2
word switch 3
```

6 pav. sed atitikmuo (1) (Windows OS)

```
C:\Windows\system32>powershell -command "Get-Content test.txt | ForEach-Object { $_ -replace '^s*(\S+)\s+(\S+)', '$2 $1' } | Set-Content test1.txt"

C:\Windows\system32>more test1.txt
switch word 1
switch word 2
switch word 3
```

7 pav. sed atitikmuo (2) (Windows OS)

### 4. SUKURKITE DVI PROGRAMAS, VIENĄ SCENARIJŲ IR PAMATUOKITE JŲ VEIKIMO LAIKĄ SU KOMANDA TIME:

#### 4.1. Pirma programa – tai shell (PowerShell) scenarijus išvedantis į ekraną Jūsų vardą ir pavardę.

8 pav. vaizduojamas scenarijaus laiko matavimas Linux operacinėje sistemoje, o 10 pav. ir 11 pav. vaizduojamas scenarijaus laiko matavimas Windows operacinėje sistemoje. 9 pav. vaizduojamas Windows PowerShell paruošimas scenarijų rašymui.

#### LINUX

```
[monika@openmandriva-x8664 3LD]$ cat name.sh
echo Monika Mirbakaite
[monika@openmandriva-x8664 3LD]$ time bash name.sh
Monika Mirbakaite

real    0m0.002s
user    0m0.002s
sys     0m0.000s
```

8 pav. scenarijaus laikas (Linux OS)

## WINDOWS

```
PS C:\Windows\System32> Set-ExecutionPolicy RemoteSigned -Scope Process
```

9 pav. pasiruošimas scenarijų rašymui (Linux OS)

```
name.ps1 X
1 Write-Output "Monika Mirbakaite"
```

10 pav. scenarijaus laikas (1) (Windows OS)

```
PS C:\Windows\System32> Measure-Command {.\name.ps1}

Days           : 0
Hours          : 0
Minutes        : 0
Seconds        : 0
Milliseconds    : 53
Ticks          : 537146
TotalDays      : 6.21696759259259E-07
TotalHours     : 1.49207222222222E-05
TotalMinutes   : 0.000895243333333333
TotalSeconds   : 0.0537146
TotalMilliseconds : 53.7146

PS C:\Windows\System32> .\name.ps1
Monika Mirbakaite
```

11 pav. scenarijaus laikas (2) (Windows OS)

### 4.2. Antra programa – tai C programa išvedantis į ekraną Jūsų vardą ir pavardę.

Kadangi C kalba nepavyko įgyvendinti užduoties Linux operacinėje sistemoje, rinkausi alternatyvų sprendimą C++ kalba. 12 pav., 13 pav., vaizduojamas pasiruošimas C++ programos rašymui. 14 pav., 15 pav. vaizduojamas C++ programos laiko matavimas Linux operacinėje sistemoje, o 16 pav. ir 17 pav. vaizduojamas C programos laiko matavimas Windows operacinėje sistemoje.

## LINUX

```
[monika@openmandriva-x8664 3LD]$ sudo dnf install gcc-c++
Last metadata expiration check: 0:33:11 ago on Tue 14 May 2024 10:07:49 PM EEST.
Dependencies resolved.
=====
Package                                     Architecture
=====
Installing:
gcc-c++                                     x86_64

Transaction Summary
=====
Install 1 Package

Total download size: 13 M
Installed size: 36 M
Is this ok [y/N]: y
```

12 pav. pasiruošimas C++ programos rašymui (1) (Linux OS)

```
[monika@openmandriva-x86_64 3LD]$ sudo dnf install libstdc++-devel glibc-devel
Last metadata expiration check: 0:38:56 ago on Tue 14 May 2024 10:07:49 PM EEST.
Package lib64stdc++-devel-13.2.1-0.20240330.1.x86_64 is already installed.
Dependencies resolved.
=====
Package                                     Architecture
=====
Installing:
glibc-devel                                x86_64
Upgrading:
glibc                                      x86_64
Installing dependencies:
kernel-headers                            x86_64
lib64crypt-devel                           x86_64
lib64pkgconf                               x86_64
pkgconf                                    x86_64
=====
Transaction Summary
=====
Install 5 Packages
Upgrade 1 Package

Total download size: 5.9 M
Is this ok [y/N]: y
```

13 pav. pasiruošimas C++ programos rašymui (2) (Linux OS)

```
[monika@openmandriva-x86_64 3LD]$ cat code.cpp
#include <iostream>
#include <chrono>

int main() {
    std::cout << "Monika Mirbakaite" << std::endl;
    return 0;
}
```

14 pav. C++ programos laikas (1) (Linux OS)

```
[monika@openmandriva-x86_64 3LD]$ g++ code.cpp -o code
[monika@openmandriva-x86_64 3LD]$ time ./code
Monika Mirbakaite

real    0m0.002s
user    0m0.002s
sys     0m0.000s
```

15 pav. C++ programos laikas (2) (Linux OS)

## WINDOWS

```
#include <stdio.h>

int main () {
    printf("Monika Mirbakaite");
return 0;
}
```

16 pav. C programos laikas (1) (Windows OS)

```
PS C:\Users\Monika\Desktop> gcc test.c -o test.exe
PS C:\Users\Monika\Desktop> .\test.exe
Monika Mirbakaite
PS C:\Users\Monika\Desktop> Measure-Command {.\test.exe}

Days           : 0
Hours          : 0
Minutes        : 0
Seconds        : 0
Milliseconds   : 17
Ticks          : 173564
TotalDays      : 2.00884259259259E-07
TotalHours     : 4.82122222222222E-06
TotalMinutes   : 0.000289273333333333
TotalSeconds   : 0.0173564
TotalMilliseconds : 17.3564
```

17 pav. C programos laikas (2) (Windows OS)

**4.3. Trečia programa – tai C programa su MPI arba OMP bibliotekos panaudojimu, apskaičiuokite visas galimas funkcijos reikšmes ir pateikite kaip rezultatą funkcijos. Išmatuokite rezultato apskaičiavimo greitį su 1, 2, 3, 4 ... MAX (Log\_CPU\_NR) procesoriaus branduoliais, sudarykite greیتaveikos kreivę.**

**LINUX**

**WINDOWS**

## **5. SUKURKITE SCENARIJŲ KURIS PRIIMTU KOMANDINĖS EILUTĖS ARGUMENTĄ IR PATEIKTU INFORMACIJĄ KAS TAI YRA: FAILO VARDAS, DIREKTORIJOS VARDAS, ARBA KAŽKAS TAI KITO.**

18 pav. vaizduojamas argumento tipo nustatymas Linux operacinėje sistemoje, 19 pav. ir 20 pav. vaizduojamas argumento tipo nustatymas Windows operacinėje sistemoje.

**LINUX**

```
[admin@openmandriva-x8664 ~]$ cat check.sh
if [ -f $1 ]
then
echo "$1 yra failo pavadinimas"
exit 0
elif [ -d $1 ]
then
echo "$1 yra direktorijos pavadinimas"
exit 0
else
echo "$1 nera nei vienas is paminetu"
exit 1
fi

[admin@openmandriva-x8664 ~]$ bash check.sh /home/admin/
/home/admin/ yra direktorijos pavadinimas
[admin@openmandriva-x8664 ~]$ bash check.sh monika
monika yra failo pavadinimas
[admin@openmandriva-x8664 ~]$ bash check.sh "monika.txt"
monika.txt nera nei vienas is paminetu
[admin@openmandriva-x8664 ~]$
```

18 pav. argumento tipas (Linux OS)

**WINDOWS**

```
check.ps1 X
1 param (
2     [string]$Path
3 )
4
5 if (Test-Path $Path) {
6     if (Test-Path $Path -PathType Leaf) {
7         Write-Output "$Path yra failo pavadinimas."
8     } elseif (Test-Path $Path -PathType Container) {
9         Write-Output "$Path yra direktorijos pavadinimas."
10    } else {
11        Write-Output "$Path nera nei vienas is paminetu."
12    }
13 } else {
14     Write-Output "$Path neegzistuoja."
15 }
```

19 pav. argumento tipas (1) (Windows OS)



```
PS C:\Windows\System32> .\check.ps1 -Path "C:\Windows\System32\test.txt"
C:\Windows\System32\test.txt yra failo pavadinimas.

PS C:\Windows\System32> .\check.ps1 -Path "C:\Windows\System32"
C:\Windows\System32 yra direktorijos pavadinimas.

PS C:\Windows\System32> .\check.ps1 -Path "\aL"
\aL neegzistuoja.
```

20 pav. argumento tipas (2) (Windows OS)

## 6. SUKURKITE SCENARIJŲ KURIAM PERDUODAMI DVEJŲ AR DAUGIAU FAILŲ VARDAI. SCENARIJUS TURI PERVARDINTI JUOS Į TUOS PAČIUS, TIK PAVADINIMUOSE TURĖS BŪTI VISOS DIDELĖS RAIDĖS (JEIGU TOKIE DAR NEEGZISTUOJA DARBO DIREKTORIJOJE).

21 pav. ir 22 pav. vaizduojamas failų pavadinimų keitimas į didžiąsias raides Linux operacinėje sistemoje, o 23 pav., 24 pav., 25 pav., 26 pav. vaizduojamas failų pavadinimų keitimas į didžiąsias raides Windows operacinėje sistemoje.

## LINUX

```
[admin@openmandriva-x8664 ~]$ cat uppercase.sh
if [ $# -lt 2 ]
then
echo "pateikite per mazai failu"
exit 1
fi

for file in "$@"
do
if [ -f "$file" ]
then
filename=$(basename -- "$file")

if [[ "$filename" != "${filename^^}" ]]
then
mv "$file" "${dirname "$file"}/${filename^^}"
echo "$file pervadintas i ${filename^^}"
else
echo "$file yra jau pavadintas didziosiomis"
fi
else
echo "$file nera tinkamas"
fi
done
[admin@openmandriva-x8664 ~]$ bash uppercase.sh k
pateikite per mazai failu
[admin@openmandriva-x8664 ~]$ bash uppercase.sh k m
k nera tinkamas
m nera tinkamas
[admin@openmandriva-x8664 ~]$ bash uppercase.sh monika mirbakaite
monika pervadintas i MONIKA
mirbakaite pervadintas i MIRBAKAITE
```

21 pav. raidžių keitimas (1) (Linux OS)

```
[admin@openmandriva-x8664 ~]$ sudo ./check.sh MONIKA
MONIKA yra failo pavadinimas
[admin@openmandriva-x8664 ~]$ sudo ./check.sh MIRBAKAITE
MIRBAKAITE yra failo pavadinimas
```

22 pav. raidžių keitimas (2) (Linux OS)

## WINDOWS

```
uppercase.ps1 X
1 param (
2     [Parameter(Mandatory=$true, Position=0, ValueFromPipeline=$true)]
3     [ValidateNotNullOrEmpty()]
4     [string[]]$FileNames
5 )
6
7 if ($FileNames.Count -lt 2) {
8     Write-Host "Pateikėte per mažai failų."
9     exit
10 }
11
12 foreach ($FileName in $FileNames) {
13     try {
14         $file = Get-ChildItem -Path $FileName -ErrorAction Stop
15         $newName = $file.Name.ToUpper()
16         if ($file.Name -ceq $newName) {
17             Write-Host "$($file.Name) jau yra didžiosiomis raidėmis, todėl pervadinimo nereikia."
18         } else {
19             $oldName = $file.Name
20             if ($oldName -eq $oldName.ToUpper()) {
21                 Write-Host "$oldName jau yra didžiosiomis raidėmis, todėl pervadinimo nereikia."
22             } else {
23                 Rename-Item -Path $file.FullName -NewName $newName
24                 Write-Host "$oldName pervadintas į $newName"
25             }
26         }
27     } catch {
28         Write-Host "$FileName nėra tinkamas failo pavadinimas."
29     }
30 }
31 }
```

23 pav. raidžių keitimas (1) (Windows OS)

```
PS C:\Windows\System32> .\uppercase.ps1 -FileNames ("test.txt")
Pateikėte per mažai failų.
```

24 pav. raidžių keitimas (2) (Windows OS)

```
PS C:\Windows\System32> .\uppercase.ps1 -FileNames ("test.txt", "monika", "mirbakaite")
test.txt pervadintas į TEST.TXT
monika pervadintas į MONIKA
mirbakaite pervadintas į MIRBAKAITE
```

25 pav. raidžių keitimas (3) (Windows OS)

```
PS C:\Windows\System32> .\uppercase.ps1 -FileNames ("test.txt", "monika", "mirbakaite")
test.txt jau yra didžiosiomis raidėmis, todėl pervadinimo nereikia.
monika jau yra didžiosiomis raidėmis, todėl pervadinimo nereikia.
mirbakaite jau yra didžiosiomis raidėmis, todėl pervadinimo nereikia.
```

26 pav. raidžių keitimas (4) (Windows OS)

## 7. SUKURKITE SCENARIJŲ KURIS NUSTATYTU KIEK LAIKO NAUDOTOJAS DIRBA SISTEMOJE (NAUDOTOJO VARDAS PERDUODAMAS KAIP PARAMETRAS).

27 pav. ir 28 pav. vaizduojamas naudotojo prisijungimo laiko nustatymas Linux operacinėje sistemoje, o 29 pav., 30 pav. vaizduojamas naudotojo prisijungimo laiko nustatymas Windows operacinėje sistemoje.

# LINUX

```
[monika@openmandriva-x8664 3LD]$ cat user.sh

current_time=$(date "+%Y-%m-%d %H:%M:%S")
last_time=$(last | grep "$1" | grep -v "still logged in" | head -n 1 | awk '{print $5, $6, $7, $8}')

# Check if last_time is empty
if [ -z "$last_time" ]; then
    echo "$1 nebuvo prisijungęs"
    exit 1
fi

difference=$((date -d "$current_time" +%s) - $(date -d "$last_time" +%s))
hours=$((difference / 3600))
difference=$((difference % 3600))
minutes=$((difference / 60))
seconds=$((difference % 60))

hours=$(printf "%02d" $hours)
minutes=$(printf "%02d" $minutes)
seconds=$(printf "%02d" $seconds)

echo "$1 prisijungė: $last_time"
echo "Naudotojas pradirbo: $hours:$minutes:$seconds."
```

27 pav. naudotojo laikas sistemoje (1) (Linux OS)

```
[monika@openmandriva-x8664 3LD]$ sudo bash user.sh monika
monika prisijungė: May 14 22:32 -
Naudotojas pradirbo: 215:46:50.
[monika@openmandriva-x8664 3LD]$ sudo bash user.sh user1
user1 nebuvo prisijungęs
[monika@openmandriva-x8664 3LD]$ sudo bash user.sh user2
user2 prisijungė: May 11 21:56 -
Naudotojas pradirbo: 288:23:00.
[monika@openmandriva-x8664 3LD]$ sudo bash user.sh user3
user3 prisijungė: May 7 15:49 -
Naudotojas pradirbo: 390:30:04.
```

28 pav. naudotojo laikas sistemoje (2) (Linux OS)

# WINDOWS

```
users.ps1 X
1 param (
2     [string]$username
3 )
4 $current_time = Get-Date
5 $quser_output = quser
6 Write-Output "quser output:"
7 Write-Output $quser_output
8 $last_time_line = ($quser_output | Select-String -Pattern $username | Select-Object -First 1).Line
9 Write-Output "Matched line:"
10 Write-Output $last_time_line
11 if (-not $last_time_line) {
12     Write-Output "Naudotojas nerastas."
13     exit
14 }
15 $last_time = $last_time_line -replace '.*\s+(\d{1,2})/(\d{1,2})/(\d{4})\s+(\d{1,2}):(\d{2})\s+[AP]M.', '$1'
16 Write-Output $last_time
17 try {
18     $last_time_dt = [datetime]::ParseExact($last_time, "M/d/yyyy h:mm tt", $null)
19     Write-Output $last_time_dt
20 } catch {
21     Write-Output "Nepavyko gauti duomenų."
22     exit
23 }
24 $difference = ($current_time - $last_time_dt).TotalSeconds
25 $hours = [math]::Floor($difference / 3600)
26 $difference = $difference % 3600
27 $minutes = [math]::Floor($difference / 60)
28 $seconds = $difference % 60
29 $hours = "{0:D2}" -f [int]$hours
30 $minutes = "{0:D2}" -f [int]$minutes
31 $seconds = "{0:D2}" -f [int]$seconds
32 Write-Output "$username buvo prisijungęs: $last_time"
33 Write-Output "Naudotojas dirbo: ${hours}:${minutes}:${seconds}."
34
```

29 pav. naudotojo laikas sistemoje (1) (Windows OS)

```
PS C:\Windows\System32> .\users.ps1 -username "Monika"
quser output:
USERNAME                SESSIONNAME             ID  STATE  IDLE TIME  LOGON TIME
>monika                  console                 1   Active  none       5/23/2024 12:36 PM
Matched line:
>monika                  console                 1   Active  none       5/23/2024 12:36 PM
5/23/2024 12:36 PM

Thursday, May 23, 2024 12:36:00 PM
Monika buvo prisijungęs: 5/23/2024 12:36 PM
Naudotojas dirbo: 01:21:03.
```

30 pav. naudotojo laikas sistemoje (2) (Windows OS)

## 8. SUKURKITE SCENARIJŲ KURIS KAIP PARAMETRĄ PRIIMTU FAILO VARDĄ, PRADŽIOS IR PABAIGOS EILUČIŲ NUMERIUS IR IŠVEDA Į EKRANĄ FAILO TURINĮ TARP NURODYTŲ EILUČIŲ.

31 pav., 32 pav. ir 33 pav. vaizduojamas pasirinktų eilučių rodymas nustatymas Linux operacinėje sistemoje, o 34 pav., 35 pav. ir 36 pav. vaizduojamas pasirinktų eilučių rodymas nustatymas Windows operacinėje sistemoje.

### LINUX

```
[monika@openmandriva-x8664 3LD]$ cat lines.sh
if [ $# -ne 3 ]; then
    echo "Naudojimas: $0 <failo_vardas> <pradžios_eilutė> <pabaigos_eilutė>"
    exit 1
fi

# Patikriname, ar failas egzistuoja
if [ ! -f "$1" ]; then
    echo "Klaida: Failas '$1' neegzistuoja."
    exit 1
fi

if ! [[ $2 =~ ^[0-9]+$ ]]; then
    echo "Klaida: Pradžios eilutė turi būti sveikasis skaičius."
    exit 1
fi

if ! [[ $3 =~ ^[0-9]+$ ]]; then
    echo "Klaida: Pabaigos eilutė turi būti sveikasis skaičius."
    exit 1
fi

sed -n "${2},${3}p" "$1"
```

31 pav. eilučių rodymas (1) (Linux OS)

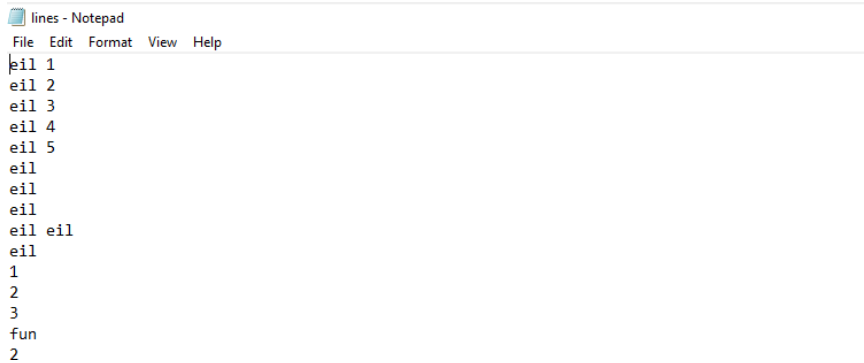
```
[monika@openmandriva-x8664 3LD]$ cat linija.txt
eil 1
eil 2
eil 3
eil 4
eil 5
eil
eil
eil
eil eil
eil
1
2
3
fun
2
```

32 pav. eilučių rodymas (2) (Linux OS)

```
[monika@openmandriva-x8664 3LD]$ sudo bash lines.sh linija.txt 1 6
eil 1
eil 2
eil 3
eil 4
eil 5
eil
```

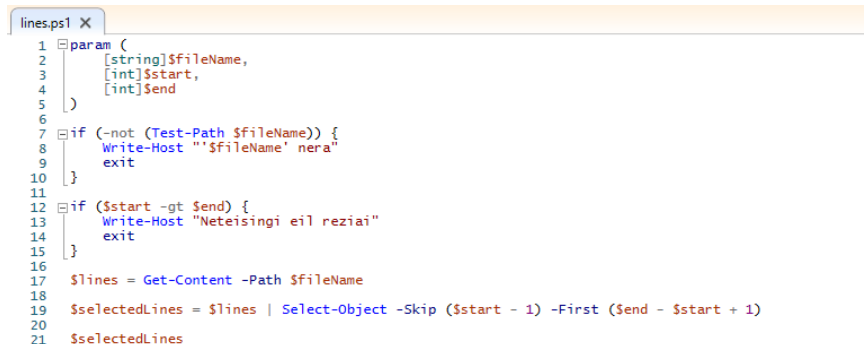
33 pav. eilučių rodymas (3) (Linux OS)

## WINDOWS



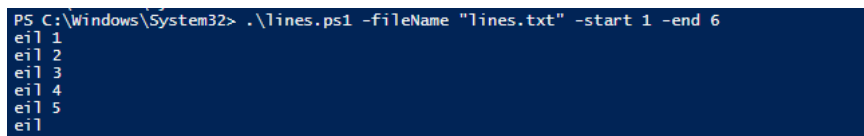
```
lines - Notepad
File Edit Format View Help
eil 1
eil 2
eil 3
eil 4
eil 5
eil
eil
eil eil
eil
1
2
3
fun
2
```

34 pav. eilučių rodymas (1) (Windows OS)



```
lines.ps1 X
1 param (
2     [string]$FileName,
3     [int]$start,
4     [int]$end
5 )
6
7 if (-not (Test-Path $FileName)) {
8     Write-Host "'$FileName' nėra"
9     exit
10 }
11
12 if ($start -gt $end) {
13     Write-Host "Neteisingi eil reziai"
14     exit
15 }
16
17 $lines = Get-Content -Path $FileName
18 $selectedLines = $lines | Select-Object -Skip ($start - 1) -First ($end - $start + 1)
19
20 $selectedLines
```

35 pav. eilučių rodymas (2) (Windows OS)



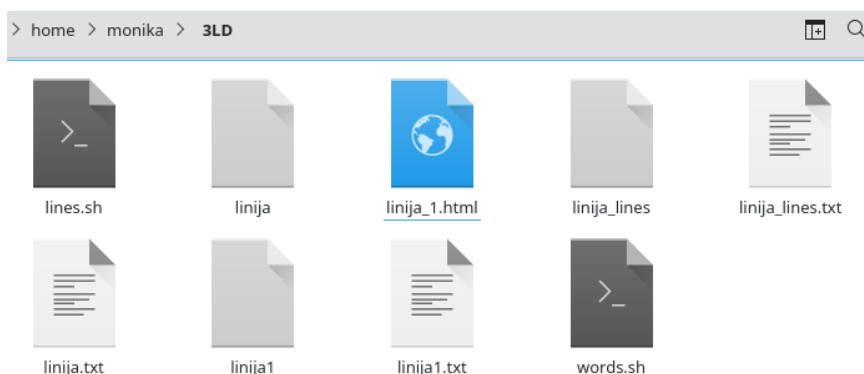
```
PS C:\Windows\System32> .\lines.ps1 -fileName "lines.txt" -start 1 -end 6
eil 1
eil 2
eil 3
eil 4
eil 5
eil
```

36 pav. eilučių rodymas (3) (Windows OS)

## 9. SUKURKITE SCENARIJŲ TRINANTĮ VISUS FAILUS APLANKALE TURINČIUS ŽODĮ PERDUOTĄ JAM KAIP PARAMETRĄ.

37 pav., 38 pav., 39 pav. ir 40 pav. vaizduojamas failų trynimas, kurių pavadinimuose egzistuoja argumento pavadinimas Linux operacinėje sistemoje, o 41 pav., 42 pav., 43 pav. ir 44 pav. vaizduojamas failų trynimas, kurių pavadinimuose egzistuoja argumento pavadinimas Windows operacinėje sistemoje.

## LINUX



37 pav. failų trynimas (1) (Linux OS)

```
[monika@openmandriva-x8664 3LD]$ cat words.sh
if [ "$#" -ne 1 ]; then
    echo "Naudojimas: $0 <zodis>"
    exit 1
fi

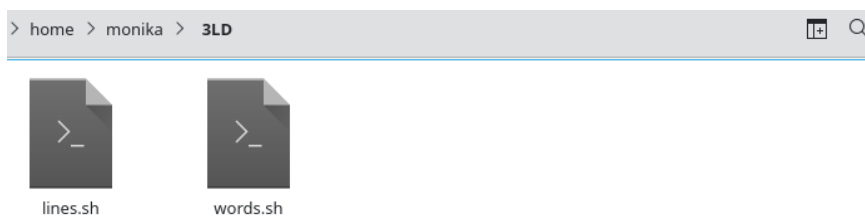
for failas in *; do
    if [ -f "$failas" ]; then
        pavadinimas=$(basename "$failas")
        if [[ "$pavadinimas" == *"$1"* ]]; then
            sudo rm "$failas"
            echo "Ištrinta: $failas"
        fi
    fi
done

echo "Visi failai, turintys žodį '$1' ištrinti."
```

38 pav. failų trynimas (2) (Linux OS)

```
[monika@openmandriva-x8664 3LD]$ sudo bash words.sh linija
[sudo] password for monika:
Ištrinta: linija
Ištrinta: linija1
Ištrinta: linija_1.html
Ištrinta: linija1.txt
Ištrinta: linija_lines
Ištrinta: linija_lines.txt
Ištrinta: linija.txt
Visi failai, turintys žodį 'linija' ištrinti.
```

39 pav. failų trynimas (3) (Linux OS)



40 pav. failų trynimas (4) (Linux OS)

## WINDOWS

PC > Local Disk (C:) > Windows > System32 > 3LD				Search 3LD
Name	Date modified	Type	Size	
linija	5/22/2024 12:37 AM	OpenDocument S...	0 KB	
linija	5/22/2024 12:36 AM	Text Document	0 KB	
linija_1	5/22/2024 12:38 AM	OpenDocument D...	0 KB	
linija_1	5/22/2024 12:38 AM	OpenDocument P...	0 KB	
linija_lines	5/22/2024 12:37 AM	OpenDocument S...	0 KB	
linija_lines	5/22/2024 12:37 AM	Text Document	0 KB	
linija1	5/22/2024 12:37 AM	OpenDocument S...	0 KB	
linija1	5/22/2024 12:36 AM	Text Document	0 KB	
words	5/22/2024 12:35 AM	Windows PowerS...	1 KB	

41 pav. failų trynimas (1) (Windows OS)

```

words.ps1 X
1 param (
2     [string]$word
3 )
4
5 # Patikriname, ar direktorija egzistuoja
6 if (-not (Test-Path -Path ".\")) {
7     Write-Host "Aplankalo nera"
8     exit
9 }
10
11 # Ieškome failų, kuriuose yra žodis ir juos triname
12 Get-ChildItem -File | Where-Object { $_.Name -like "*$word*" } | ForEach-Object {
13     Remove-Item $_.FullName -Force
14     Write-Host "Ištrinta: $($_.FullName)"
15 }

```


42 pav. failų trynimasis (2) (Windows OS)

```

PS C:\Windows\System32> cd 3LD
PS C:\Windows\System32\3LD> .\words.ps1 linija
Ištrinta: C:\Windows\System32\3LD\linija.ods
Ištrinta: C:\Windows\System32\3LD\linija.txt
Ištrinta: C:\Windows\System32\3LD\linija1.ods
Ištrinta: C:\Windows\System32\3LD\linija1.txt
Ištrinta: C:\Windows\System32\3LD\linija1.odg
Ištrinta: C:\Windows\System32\3LD\linija1.odp
Ištrinta: C:\Windows\System32\3LD\linija_lines.ods
Ištrinta: C:\Windows\System32\3LD\linija_lines.txt
Visi failai, turintys žodi 'linija' ištrinti.

```

43 pav. failų trynimasis (3) (Windows OS)

is PC > Local Disk (C:) > Windows > System32 > 3LD				Search 3LD
Name	Date modified	Type	Size	
 words	5/22/2024 12:45 AM	Windows PowerS...	1 KB	

44 pav. failų trynimasis (4) (Windows OS)

## 10. SUKURKITE SCENARIJŲ KURIAM PERDUODAMAS TEKSTINIO FAILO VARDAS. SCENARIJUS TURI ATLIKTI TOKIUS VEIKSMUS: APJUNGTI KAS DVI EILUTES Į VIENĄ. APSKAIČIUOTI NAUJŲ EILUČIŲ ILGIUS IR IŠVESTI ŠIĄ INFORMACIJĄ Į EKRANĄ.

45 pav., 46 pav., 47 pav. ir 48 pav. vaizduojamas failo eilučių apjungimas, prieš ir po eilučių ilgių skaičiavimas Linux operacinėje sistemoje. 49 pav., 50 pav. ir 51 pav. vaizduojamas failo eilučių apjungimas, prieš ir po eilučių ilgių skaičiavimas Windows operacinėje sistemoje.

# LINUX

```
[monika@openmandriva-x8664 3LD]$ cat join.sh
if [ $# -ne 1 ]; then
    echo "Usage: $0 <filename>"
    exit 1
fi

if [ ! -f "$1" ]; then
    echo "File '$1' does not exist."
    exit 1
fi

tmp_file=$(mktemp)
merged_file="${1%.*}_merged.txt"

echo "Failo $1 eil ilgiai:"
while IFS= read -r line; do
    length=$(echo "$line" | wc -c)
    echo "$length" >> "$tmp_file"
done < "$1"
cat "$tmp_file"

awk 'NR%2{printf "%s ",$0;next} 1' "$1" > "$merged_file"

echo "Apjungto $1 failo eil ilgiai:"
while IFS= read -r line; do
    length=$(echo "$line" | wc -c)
    echo "$length"
done < "$merged_file"

rm "$tmp_file"

exit 0
```

45 pav. eilučių jungimas (1) (Linux OS)

```
[monika@openmandriva-x8664 3LD]$ cat linija.txt
eil 1
eil 2
eil 3
eil 3
eil 4
eil 5
eil
eil
eil
eil eil
eil
1
2
3
fun
2
```

46 pav. eilučių jungimas (2) (Linux OS)



```
[monika@openmandriva-x8664 3LD]$ sudo bash join.sh linija.txt
Failo linija.txt eil ilgiai:
6
6
6
6
6
6
4
4
4
8
4
2
2
2
4
2
Apjungto linija.txt failo eil ilgiai:
12
12
12
8
12
6
4
6
```

47 pav. eilučių jungimas (3) (Linux OS)

```
[monika@openmandriva-x8664 3LD]$ cat linija_merged.txt
eil 1 eil 2
eil 3 eil 3
eil 4 eil 5
eil eil
eil eil eil
eil 1
2 3
fun 2
```

48 pav. eilučių jungimas (4) (Linux OS)

## WINDOWS

```
join.ps1* X
1 param (
2     [Parameter(Mandatory=$true)]
3     [string]$file
4 )
5
6 if (-not (Test-Path $file)) {
7     Write-Host "$file nerastas."
8     exit
9 }
10 $oldFile = Get-Content $file
11 $oldLineLengths = @()
12 foreach ($line in $oldFile) {
13     $oldLineLengths += $line.Length
14 }
15 $newFile = "merged_$file"
16 $newLineLengths = @()
17 for ($i = 0; $i -lt $oldFile.Count; $i += 2) {
18     if ($i + 1 -lt $oldFile.Count) {
19         $mergedLine = $oldFile[$i] + $oldFile[$i + 1]
20         $newLineLengths += $mergedLine.Length
21         Add-Content -Path $newFile -Value $mergedLine
22     } else {
23         Add-Content -Path $newFile -Value $oldFile[$i]
24         $newLineLengths += $oldFile[$i].Length
25     }
26 }
27 Write-Host "$file eilučių ilgiai:"
28 foreach ($length in $oldLineLengths) {
29     Write-Host $length
30 }
31 Write-Host "$newFile eilučių ilgiai:"
32 foreach ($length in $newLineLengths) {
33     Write-Host $length
34 }
35 Write-Host "Apjungtos eilutės įrašytos į failą: $newFile"
36
```

49 pav. eilučių jungimas (1) (Windows OS)

```

PS C:\Windows\System32> .\join.ps1 lines.txt
lines.txt eilučių ilgiai:
5
5
5
5
5
4
3
4
7
3
1
1
1
3
1
merged_lines.txt eilučių ilgiai:
10
10
9
7
10
2
4
1
Apjungtos eilutės įrašytos į failą: merged_lines.txt

```

50 pav. eilučių jungimas (2) (Windows OS)

```

C:\Windows\system32>more lines.txt
eil 1
eil 2
eil 3
eil 4
eil 5
eil
eil
eil
eil eil
eil
1
2
3
fun
2

C:\Windows\system32>more merged_lines.txt
eil 1eil 2
eil 3eil 4
eil 5eil
eileil
eil eileil
12
3fun
2

```

51 pav. eilučių jungimas (3) (Windows OS)

# PRIEDAI

## 1. LINUX

### 1.1. Pirmosios užduoties programa

```
who > monika
more monika
```

### 1.2. Antrosios užduoties programa

```
(date && who) > mirbakaite
more mirbakaite
```

### 1.3. Trečiosios užduoties programa

```
sed 's/\([^[:space:]]*\)[[:space:]]*\([^[:space:]]*\)/\2 \1/' test
```

### 1.4. Ketvirtosios užduoties programa

#### 1.4.1. Pirmoji dalis (name.sh)

```
echo Monika Mirbakaite
```

#### 1.4.2. Antroji dalis

```
#include <iostream>

int main() {
    std::cout << "Monika Mirbakaite" << std::endl;
    return 0;
}

g++ code.cpp -o code
./code
```

### 1.5. Penktosios užduoties programa (check.sh)

```
if [ -f $1 ]
then
echo "$1 yra failo pavadinimas"
exit 0
elif [ -d $1 ]
then
echo "$1 yra direktorijos pavadinimas"
exit 0
else
echo "$1 nera nei vienas is paminetu"
exit 1
fi
```

### 1.6. Šeštosios užduoties programa (uppercase.sh)

```
if [$# -lt 2 ]
then
echo "pateikete per mazai failu"
exit 1
fi
```

```

for file in "$@"
do
if [ -f "$file" ]
then
filename=$(basename -- "$file")
if [[ "$filename" != "${filename^^}" ]]
then
mv "$file" "$(dirname "$file")/${filename^^}"
echo "$file pervadintas i ${filename^^}"
else echo "${filename^^} jau pavadintas didziosiomis"
fi
else
echo "$file nera tinkamas"
fi
done

```

## 1.7. Septintosios užduoties programa (user.sh)

```

current_time=$(date "+%Y-%m-%d %H:%M:%S")
last_time=$(last | grep "$1" | grep -v "still logged in" | head -n 1 | awk '{print $5, $6, $7, $8}')

# Check if last_time is empty
if [ -z "$last_time" ]; then
echo "$1 nebuvo prisijunges"
exit 1
fi

difference=$((date -d "$current_time" +%s) - $(date -d "$last_time" +%s))
hours=$((difference / 3600))
difference=$((difference % 3600))
minutes=$((difference / 60))
seconds=$((difference % 60))

hours=$(printf "%02d" $hours)
minutes=$(printf "%02d" $minutes)
seconds=$(printf "%02d" $seconds)

echo "$1 prisijunge: $last_time"
echo "Naudotojas pradirbo: $hours:$minutes:$seconds."

```

## 1.8. Aštuntosios užduoties programa (lines.sh)

```

if [ $# -ne 3 ]; then
echo "Naudojimas: $0 <failo_vardas> <pradžios_eilutė> <pabaigos_eilutė>"
exit 1
fi

# Patikriname, ar failas egzistuoja
if [ ! -f "$1" ]; then
echo "Klaida: Failas '$1' neegzistuoja."
exit 1
fi

if ! [[ $2 =~ ^[0-9]+$ ]]; then
echo "Klaida: Pradžios eilutė turi būti sveikasis skaičius."
exit 1
fi

if ! [[ $3 =~ ^[0-9]+$ ]]; then
echo "Klaida: Pabaigos eilutė turi būti sveikasis skaičius."
exit 1
fi

sed -n "${2},${3}p" "$1"

```

## 1.9. Devintosios užduoties programa (words.sh)

```
if [ "$#" -ne 1 ]; then
    echo "Naudojimas: $0 <zodis>"
    exit 1
fi

for failas in *; do
    if [ -f "$failas" ]; then
        pavadinimas=$(basename "$failas")
        if [[ "$pavadinimas" == *"$1"* ]]; then
            sudo rm "$failas"
            echo "Ištrinta: $failas"
        fi
    fi
done

echo "Visi failai, turintys žodį '$1' ištrinti."
```

## 1.10. Dešimtosios užduoties programa

```
if [ $# -ne 1 ]; then
    echo "Usage: $0 <filename>"
    exit 1
fi

if [ ! -f "$1" ]; then
    echo "File '$1' does not exist."
    exit 1
fi

tmp_file=$(mktemp)

merged_file="${1%.*}_merged.txt"

echo "Failo $1 eil ilgiai:"
while IFS= read -r line; do
    length=$(echo "$line" | wc -c)
    echo "$length" >> "$tmp_file"
done < "$1"
cat "$tmp_file"

awk 'NR%2{printf "%s ",$0;next} 1' "$1" > "$merged_file"

echo "Apjungto $1 failo eil ilgiai:"
while IFS= read -r line; do
    length=$(echo "$line" | wc -c)
    echo "$length"
done < "$merged_file"

rm "$tmp_file"

exit 0
```

## 2. WINDOWS

### 2.1. Pirmosios užduoties programa

```
query user > monika
```

### 2.2. Antrosios užduoties programa

```
date /T & query user > mirbakaite
```

## 2.3. Trečiosios užduoties programa

```
powershell -command "Get-Content test.txt | ForEach-Object { $_-replace `^s*(\S+)\s+(\S+)`, '$2 $1' } |Set-Content test1.txt"
```

## 2.4. Ketvirtosios užduoties programa

### 2.4.1. Pirmoji dalis (name.ps1)

```
Write-Output "Monika Mirbakaite"
```

### 2.4.2. Antroji dalis (test.c)

```
#include <stdio.h>

int main() {
    printf("Monika Mirbakaite");
    return 0;
}

gcc test.c -o test.exe
.\test.exe
```

## 2.5. Penktosios užduoties programa (check.ps1)

```
param (
    [string]$Path
)

if (Test-Path $Path) {
    if (Test-Path $Path -PathType Leaf) {
        Write-Output "$Path yra failo pavadinimas."
    } elseif (Test-Path $Path -PathType Container) {
        Write-Output "$Path yra direktorijos pavadinimas."
    } else {
        Write-Output "$Path nera nei vienas is paminetu."
    }
} else {
    Write-Output "$Path neegzistuoja."
}
```

## 2.6. Šeštosios užduoties programa (uppercase.ps1)

```
param (
    [Parameter(Mandatory=$true, Position=0, ValueFromPipeline=$true)]
    [ValidateNotNullOrEmpty()]
    [string[]]$FileNames
)

if ($FileNames.Count -lt 2) {
    Write-Host "Pateikėte per mažai failų."
    exit
}

foreach ($fileName in $FileNames) {
    try {
        $file = Get-ChildItem -Path $fileName -ErrorAction Stop
        $newName = $file.Name.ToUpper()
        if ($file.Name -ceq $newName) {
            Write-Host "$($file.Name) jau yra didžiosiomis raidėmis, todėl pervadinimo nereikia."
        } else {
            $oldName = $file.Name
            if ($oldName -eq $oldName.ToUpper()) {

```

```

        Write-Host "$oldName jau yra didžiosiomis raidėmis, todėl pervadinimo
nereikia."
    } else {
        Rename-Item -Path $fileName -NewName $newName
        Write-Host "$oldName pervadintas į $newName"
    }
}
} catch {
    Write-Host "$fileName nėra tinkamas failo pavadinimas."
}
}

```

## 2.7. Septintosios užduoties programa (user.ps1)

```

param (
    [string]$username
)
$current_time = Get-Date
$quser_output = quser
Write-Output "quser output:"
Write-Output $quser_output
$last_time_line = ($quser_output | Select-String -Pattern $username | Select-Object -
First 1).Line
Write-Output "Matched line:"
Write-Output $last_time_line
if (-not $last_time_line) {
    Write-Output "Naudotojas nerastas."
    exit
}
$last_time = $last_time_line -replace
'.*\s+(\d{1,2})\/(\d{1,2})\/(\d{4})\s+(\d{1,2}):\d{2}\s+[AP]M).*', '$1'
Write-Output $last_time
try {
    $last_time_dt = [datetime]::ParseExact($last_time, "M/d/yyyy h:mm tt", $null)
    Write-Output $last_time_dt
} catch {
    Write-Output "Nepavyko gauti duomenu."
    exit
}
$difference = ($current_time - $last_time_dt).TotalSeconds
$hours = [math]::Floor($difference / 3600)
$difference = $difference % 3600
$minutes = [math]::Floor($difference / 60)
$seconds = $difference % 60
$hours = "{0:D2}" -f [int]$hours
$minutes = "{0:D2}" -f [int]$minutes
$seconds = "{0:D2}" -f [int]$seconds
Write-Output "$username buvo ptisijungęs: $last_time"
Write-Output "Naudotojas dirbo: ${hours}:${minutes}:${seconds}."

```

## 2.8. Aštuntosios užduoties programa (lines.ps1)

```

param (
    [string]$fileName,
    [int]$start,
    [int]$end
)

if (-not (Test-Path $fileName)) {
    Write-Host "'$fileName' nėra"
    exit
}

if ($start -gt $end) {
    Write-Host "Neteisingi eil rezhiai"
    exit
}

```

```

}

$lines = Get-Content -Path $fileName

$selectedLines = $lines | Select-Object -Skip ($start - 1) -First ($end - $start + 1)

$selectedLines

```

## 2.9. Devintosios užduoties programa (words.ps1)

```

param (
    [string]$word
)

if (-not (Test-Path -Path ".\")) {
    Write-Host "Aplankalo nera"
    exit
}

Get-ChildItem -File | Where-Object { $_.Name -like "$word*" } | ForEach-Object {
    Remove-Item $_.FullName -Force
    Write-Host "Ištrinta: $(_.FullName)"
}

Write-Host "Visi failai, turintys žodi '$word' ištrinti."

```

## 2.10. Dešimtosios užduoties programa (join.ps1)

```

param (
    [Parameter(Mandatory=$true)]
    [string]$file
)

if (-not (Test-Path $file)) {
    Write-Host "$file nerastas."
    exit
}

$oldFile = Get-Content $file
$oldLineLengths = @()
foreach ($line in $oldFile) {
    $oldLineLengths += $line.Length
}

$newFile = "merged_$file"
$newLineLengths = @()
for ($i = 0; $i -lt $oldFile.Count; $i += 2) {
    if ($i + 1 -lt $oldFile.Count) {
        $mergedLine = $oldFile[$i] + $oldFile[$i + 1]
        $newLineLengths += $mergedLine.Length
        Add-Content -Path $newFile -Value $mergedLine
    } else {
        Add-Content -Path $newFile -Value $oldFile[$i]
        $newLineLengths += $oldFile[$i].Length
    }
}

Write-Host "$file eilučių ilgiai:"
foreach ($length in $oldLineLengths) {
    Write-Host $length
}

Write-Host "$newFile eilučių ilgiai:"
foreach ($length in $newLineLengths) {
    Write-Host $length
}

Write-Host "Apjungtos eilutės įrašytos į failą: $newFile"

```



