zadanie 1
Plik ma prawa do wykonywania.

```
┌──(kali㉿kali)-[/home/kali]
└─PS> vim ./triangleArea.ps1

┌──(kali㉿kali)-[/home/kali]
└─PS> ./triangleArea.ps1 "20"
Podaj argumenty

┌──(kali㉿kali)-[/home/kali]
└─PS> ./triangleArea.ps1 "20" "10"
Pole trójkata dla wysokosci rownej 20 oraz dlugosci podstawy rownej 10
Pole wynosi 100

┌──(kali㉿kali)-[/home/kali]
└─PS>
```

skrypt **triangleArea.ps1:**

```
$height = $args[0]
$length = $args[1]

if (!$height -or !$length) {
        Write-Host "Podaj argumenty"
        exit
}

Write-Host "Pole trójkata dla wysokosci rownej $height oraz dlugosci podstawy rownej $length"

$area = 0.5 * $height * $length

Write-Host "Pole wynosi $area"
~
~
```

zadanie 2

```
┌──(kali㉿kali)-[/home/kali]
└─PS> ./isBiggerThan10.ps1
Podaj liczbe: 6
Liczba jest mniejsza od 10

┌──(kali㉿kali)-[/home/kali]
└─PS> ./isBiggerThan10.ps1
Podaj liczbe: 10
Liczba rowna 10

┌──(kali㉿kali)-[/home/kali]
└─PS> ./isBiggerThan10.ps1 15
Podaj liczbe: 15
Liczba jest wieksza od 10
```

skrypt **isBiggerThan10.ps1:**

```
$userInput = Read-Host -Prompt "Podaj liczbe"

$userInput = [double]$userInput

if ($userInput -gt 10) {
        Write-Host "Liczba jest wieksza od 10"
} elseif ($userInput -eq 10) {
        Write-Host "Liczba rowna 10"
} else {
        Write-Host "Liczba jest mniejsza od 10"
}
~
~
~
```

zadanie 3

```
┌──(kali㉿kali)-[/home/kali]
└─PS> ./checkLogin.ps1
Podaj nazwe uzytkownika: trest
Wprowadz haslo: *****
Zle dane

┌──(kali㉿kali)-[/home/kali]
└─PS> ./checkLogin.ps1
Podaj nazwe uzytkownika: login123
Wprowadz haslo: *****
Zalogowano

┌──(kali㉿kali)-[/home/kali]
└─PS>
```

skrypt **checkLogin.ps1:**

```
$login = "login123"
$password = "cisco"


$inputLogin = Read-Host -Prompt "Podaj nazwe uzytkownika"
$inputPassword = Read-Host -Prompt "Wprowadz haslo" -MaskInput


if ($inputLogin -eq $login -and $password -eq $inputPassword) {
        Write-Host "Zalogowano"
} else {
        Write-Host "Zle dane"
}
```

zadanie 4

```
┌──(kali㉿kali)-[/home/kali]
└─PS> ./generateIp.ps1
192.168.1.0
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.4
192.168.1.5
192.168.1.6
192.168.1.7
192.168.1.8
192.168.1.9
```

skrypt **generateIp.ps1:**

```
for ($i = 0; $i -lt 10; $i++) {
        Write-Host "192.168.1.$i"
}
~
~
~
~
```

zadanie 5

```
┌──(kali㉿kali)-[/home/kali]
└─PS> ./somefunctions.ps1
 Data na komputerze Kamyk to 04/28/2024 12:56:47
Wersja systemu na komputerze Kamyk:
Distributor ID: Kali
Description:    Kali GNU/Linux Rolling
Release:        2024.1
Codename:       kali-rolling
Obecny uzytkownik na komputerze Kamyk: kali
Ip na komputerze Kamyk to: 10.0.2.15
```

skrypt **somefunctions.s1**

```powershell
$computerName = "Kamyk"

function currentDate {
        $date = Get-Date
        Write-Host " Data na komputerze $computerName to $date"
}

function systemVersion {
        Write-Host "Wersja systemu na komputerze $($computerName):"
        $(lsb_release -a)
}

function getUser {
        Write-Host "Obecny uzytkownik na komputerze $($computerName): $ENV:USER "
}

function getIp {
        $ip = (hostname -I) -split ' ' | Select-Object -First 1
        Write-Host "Ip na komputerze $($computerName) to: $ip"
}

currentDate
systemVersion
getUser
getIp
```

zadanie 6

Zadanie zrobiłem wraz z kolegą, ponieważ nie mam niestety dostępnego windowsa, co za tym idzie funkcja Get-ComputerInfo nie działała.

```
$csvFilePath = "input_file.csv"
$Cpu = WMIC CPU Get Name

$csvData = Import-Csv -Path $csvFilePath -Delimiter ";"

foreach ($row in $csvData) {

    if ($row.GenerateReport -eq "True") {
        $computerInfo = Get-ComputerInfo

        if ($row.Component -eq "Computername") {
            Write-Host "Computername: $($computerInfo.CsName)"
        }
        if ($row.Component -eq "Manufacturer") {
            Write-Host "Manufacturer: $($computerInfo.BiosManufacturer)"
        }
        if ($row.Component -eq "Model") {
            Write-Host "Model: $($computerInfo.CsModel)"
        }
```

```
        if ($row.Component -eq "SerialNumber") {
            Write-Host "SerialNumber: $($computerInfo.BiosSeralNumber)"
        }
        if ($row.Component -eq "RAM") {
            Write-Output "RAM: $($computerInfo.CsTotalPhysicalMemory / 1GB) GB"
        }
         if ($row.Component -eq "CpuName") {
            Write-Host "Cpu: $($Cpu.replace(' ', ''))"
        }
    }
}
```