4-Amaliy mashg’ulot. Powershellda ssenariylar yaratish

Ishning maqsadi: Windows OTning buyruqlar satri va PowerShell muhitlari vositasida talabalarda OTning ishini avtomatlashtirishga oid bilimlarini oshirish va ish ko’nikmalarini hosil qilishdan iborat.

Ishda o’rganiladigan vazifalar:

1) Windows operatsion tizimida ishlashni avtomatlashtirish uchun mo'ljallangan standart texnologiyalar (WSH, WMI, ADSI) hamda dasturiy mahsulotlarini (Cmd.exe buyruqli itterpretator, CScript va WScript skript serverlari, Windows PowerShell qobig'i) o’rganish;

2) Windows operatsion tizimi ma'murlari va foydalanuvchilarining kundalik ishini buyruqlar qatorida (interaktiv rejim) bajarish yoki oldindan (PowerShell vositasida) yaratilgan skriptlarni ishga tushirish (to'plamli rejim) orqali avtomatlashtirish imkonini beruvchi Microsoft tomonidan ishlab chiqilgan dasturiy vositalarni ko'rib chiqish;

3) Windowsning barcha versiyalarida mavjud bo'lgan standart Cmd.exe buyruq qatori qobig'ining imkoniyatlarini va ushbu qobiq tomonidan qo'llab-quvvatlanadigan buyruqli fayllari tilini ko'rib chiqish;

4) WMI Command-line (WMIC) dasturi yordamida buyruq satridan WMI ob'ektlari bilan qanday ishlashni ko'rib chiqish.

Vazifalar:

1. **Windows buyruqlar satri** interpretatorida buyruqlardan amaliy foydalanish, buyruqlar bilan tanishish va ularni variantga muvofiq bajarish, skrinshot olish va har bir buyruq tavsifini berish.

Variantlar:

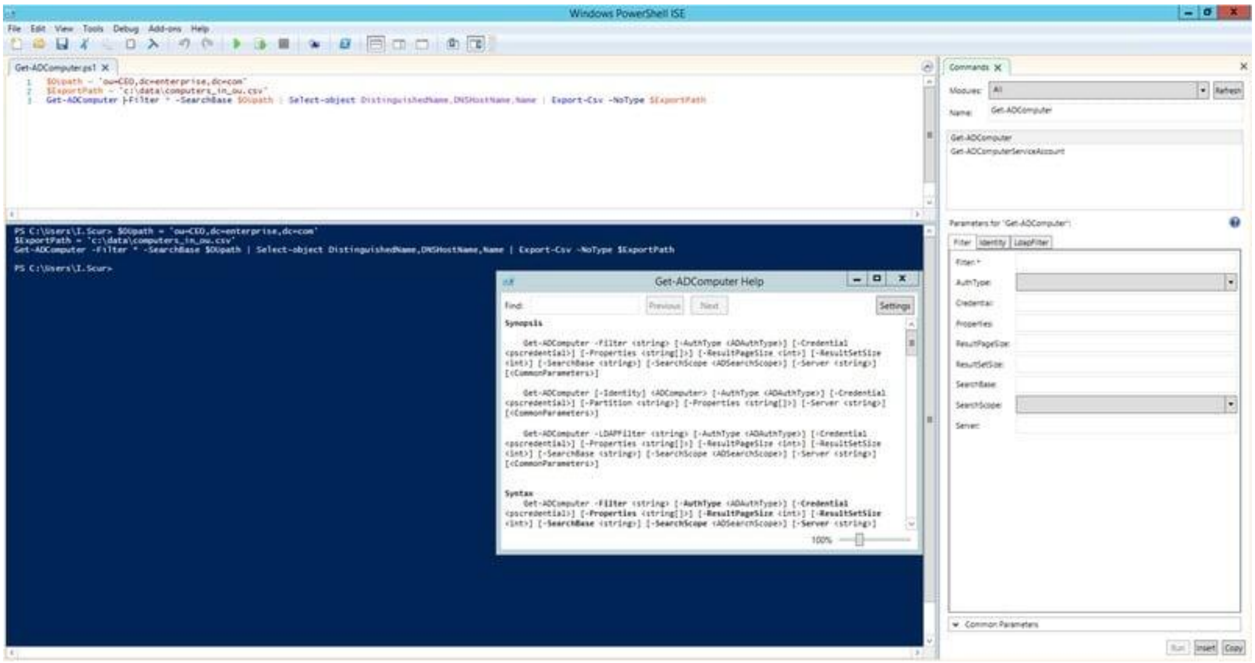
1.	APPEND, COLOR, FC, MSTSC, REM, TIME
2.	ARP, COMMAND, FIND, NBTSTAT, RENAME (REN), TELNET
3.	ASSOC, COMP, FINDSTR, NET, REPLACE, TFTP
4.	AT, COMPACT, FOR, NETCFG, RESET, TIMEOUT
5.	ATTRIB, CONVERT, FORFILES, NETSH, RMDIR, TITLE
6.	Auditpol, COPY, FORMAT, NETSTAT, ROBOCOPY, TRACERT
7.	BASH, Cscript, FSUTIL, NSLOOKUP, ROUTE, TREE
8.	BCDBOOT, DATE, FTP, OPENFILES, RUNAS, TSCON
9.	BCDEDIT, DEBUG, FTYPE, PATH, RUNDLL32, TSDISCON
10.	BOOTCFG, DEFRAG, GETMAC, PATHPING, SC, TSKILL
11.	BOOTIM, DEL, GOTO, PAUSE, SCHTASKS, TYPE
12.	BOOTREC, DevCon, GPRESULT, PING, SCLIST, TypePerf
13.	BOOTSECT, DIANTZ, GPUPDATE, PKGMGR, ScriptRunner, TZUTIL
14.	BREAK, DIR, HELP, PNPUTIL, SET, VaultCmd
15.	CACLS, DISKCOMP, HOSTNAME, POPD, SETLOCAL, VER

16.	CALL, DISKCOPY, iCACLS, POWERCFG, SETX, VERIFIER
17.	CD, DISKPART, IF, PRINT, SFC, VERIFY
18.	CHANGE, DISM, IPCONFIG, PROMPT, SHARE, VOL
19.	CHGLOGON, DISPDIAG, LABEL PUSHD SHIFT VSSADMIN
20.	CHGPORT, DJOIN, LOGMAN, PSR, SHUTDOWN, W32TM
21.	CHGUSR, DOSKEY, LOGOFF, QPROCESS, SLEEP, WAITFOR
22.	CHCP, DRIVERQUERY, MAKECAB, QUERY, SLMGR, WBADMIN
23.	CHKDSK, DxDiag, MBR2GPT, QUSER, SORT, WEVTUTI
24.	CheckNetIsolation, ECHO, MEM, RASDIAL, START, WHERE
25.	CHKNTFS, EDIT, MD, RASPHONE, STORDIAG, WHOAMI
26.	CHOICE, ENDLOCAL, MKLINK, RD, SUBST, WINDIFF
27.	CIPHER, ERASE, MODE, REAGENTC, SxSTrace, WinMgmt
28.	CLEARMGR, ESENTUTL, MORE, RECOVER, SYSTEMINFO, WINRM
29.	CLIP, EVENTCREATE, MOUNTVOL, REG, TAKEOWN, WINRS
30.	CLS, EXPAND, MOVE, REGEDIT, TAR, WINSAT
31.	CMD, EXTRACT, MOVEFILE, REGSVR32, TASKKILL, WMIC
32.	CMDKEY, EXIT, MSG, REGINI, TASKLIST, WSCollect

2. **Windows PowerShell** muhiti cmdletlari bilan ishlash

PowerShell – bu Windows oilasining operatsion tizimlarini sozlash uchun keng imkoniyatlarni ta'minlovchi buyruqlar qatori interfeysiga ega bo’lgan, obyektga yo'naltirilgan dasturlash mexanizmi va skript tili hisoblanadi. Uning buyruqlari (cmdletlari) konsolda va PowerShell ISE (Integrated Scripting Environment) skriptlar yozish muhitini taklif etadi.

https://tproger.ru/translations/powershell-tutorial/



```
$OUpath = 'ou=CEO,dc=enterprise,dc=com'
$ExportPath = 'c:\data\computers_in_ou.csv'
Get-ADComputer -Filter * -SearchBase $OUpath | Select-object DistinguishedName,DNSHostName,Name | Export-Csv -NoType $ExportPath
```

```
PS C:\Users\I.Scur> $OUpath = 'ou=CEO,dc=enterprise,dc=com'
$ExportPath = 'c:\data\computers_in_ou.csv'
Get-ADComputer -Filter * -SearchBase $OUpath | Select-object DistinguishedName,DNSHostName,Name | Export-Csv -NoType $ExportPath

PS C:\Users\I.Scur>
```

```
Synopsis

    Get-ADComputer -Filter <string> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Properties <string[]>] [-ResultPageSize <int>] [-ResultSetSize
<int>] [-SearchBase <string>] [-SearchScope <ADSearchScope>] [-Server <string>]
[<CommonParameters>]

    Get-ADComputer [-Identity] <ADComputer> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Partition <string>] [-Properties <string[]>] [-Server <string>]
[<CommonParameters>]

    Get-ADComputer -LDAPFilter <string> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Properties <string[]>] [-ResultPageSize <int>] [-ResultSetSize
<int>] [-SearchBase <string>] [-SearchScope <ADSearchScope>] [-Server <string>]
[<CommonParameters>]


Syntax
    Get-ADComputer -Filter <string> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Properties <string[]>] [-ResultPageSize <int>] [-ResultSetSize
<int>] [-SearchBase <string>] [-SearchScope <ADSearchScope>] [-Server <string>]
```

PowerShelldagi uchta asosiy cmdletlar:

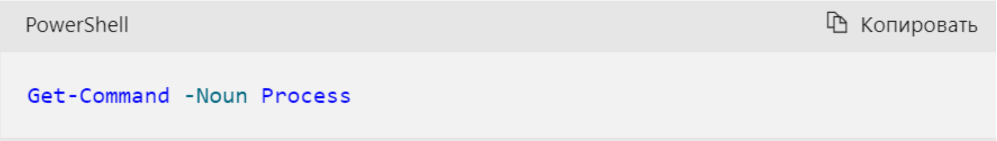
```
Get-Command

Get-Help

Get-Member
```

Get-Help

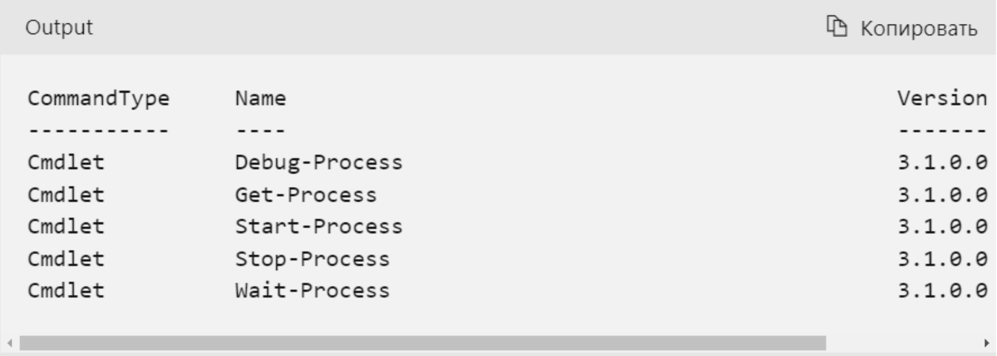
```
    Get-Help -Name Get-Help

    Get-Help -Name Get-Help –Full

Get-Help -Name Get-Command -Full
Get-Help -Name Get-Command -Detailed
Get-Help -Name Get-Command -Examples
Get-Help -Name Get-Command -Online
Get-Help -Name Get-Command -Parameter Noun
    Get-Help -Name Get-Command –ShowWindow

    help Get-Command -Full | Out-GridView

    help *process*

    Get-Help processes
```

Get-Command

```
Get-Command -Noun Process
```



```
Get-Command -Noun Process
```



```
CommandType     Name                                               Version
-----------     ----                                               -------
Cmdlet          Debug-Process                                      3.1.0.0
Cmdlet          Get-Process                                        3.1.0.0
Cmdlet          Start-Process                                      3.1.0.0
Cmdlet          Stop-Process                                       3.1.0.0
Cmdlet          Wait-Process                                       3.1.0.0
```

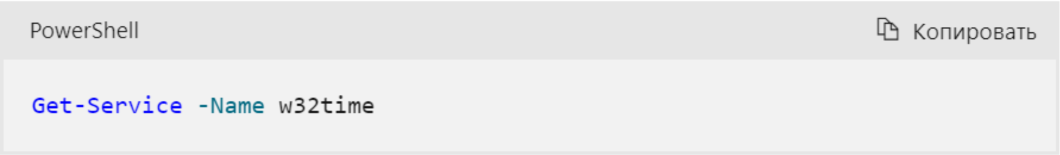
```
Get-Command -Name *service*
```

```
Get-Command -Name *service* -CommandType Cmdlet, Function,
Alias
```

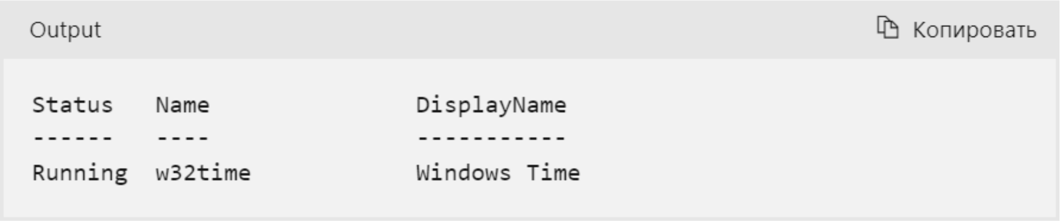
```
Update-Help
```

Get-Member

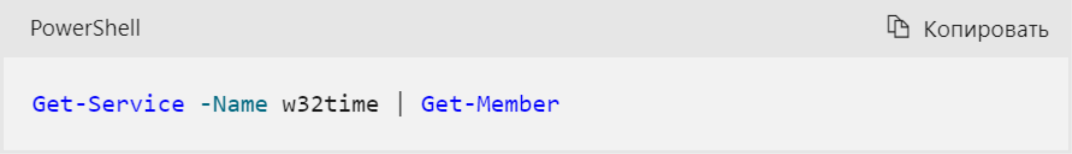
`Get-Member` buyruqlar uchun mavjud bo'lgan ob'ektlar, xususiyatlar va usullarni aniqlashga yordam beradi. Ob'ektga yo'naltirilgan chiqishni ishlab chiqaradigan barcha buyruqlar `Get-Member`ga o'tkazilishi mumkin.



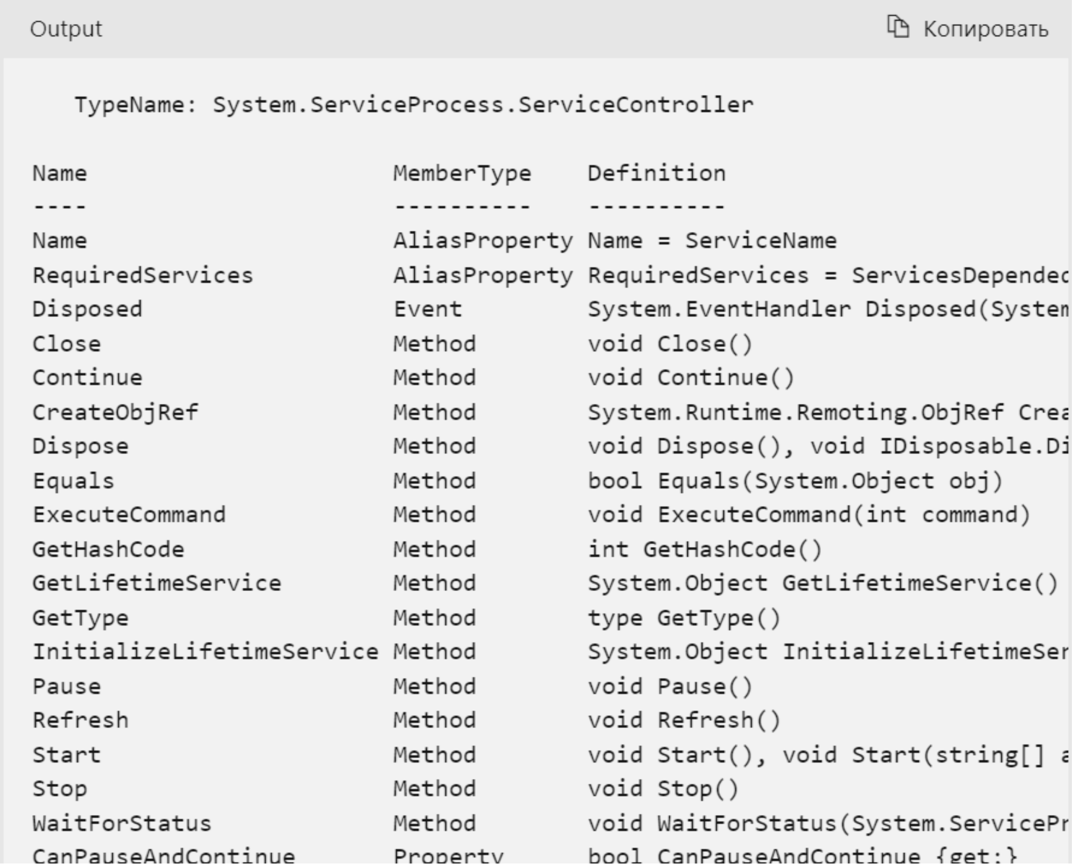
```
Get-Service -Name w32time
```



```
Status   Name               DisplayName
------   ----               -----------
Running  w32time            Windows Time
```



```powershell
Get-Service -Name w32time | Get-Member
```



```
   TypeName: System.ServiceProcess.ServiceController

Name                      MemberType    Definition
----                      ----------    ----------
Name                      AliasProperty Name = ServiceName
RequiredServices          AliasProperty RequiredServices = ServicesDependec
Disposed                  Event         System.EventHandler Disposed(Systen
Close                     Method        void Close()
Continue                  Method        void Continue()
CreateObjRef              Method        System.Runtime.Remoting.ObjRef Crea
Dispose                   Method        void Dispose(), void IDisposable.Di
Equals                    Method        bool Equals(System.Object obj)
ExecuteCommand            Method        void ExecuteCommand(int command)
GetHashCode               Method        int GetHashCode()
GetLifetimeService        Method        System.Object GetLifetimeService()
GetType                   Method        type GetType()
InitializeLifetimeService Method        System.Object InitializeLifetimeSer
Pause                     Method        void Pause()
Refresh                   Method        void Refresh()
Start                     Method        void Start(), void Start(string[] a
Stop                      Method        void Stop()
WaitForStatus             Method        void WaitForStatus(System.ServicePr
CanPauseAndContinue       Property      bool CanPauseAndContinue {get;}
```

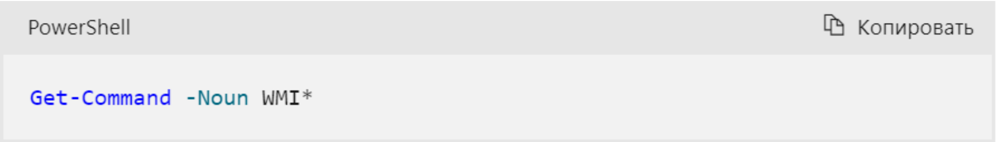
WMI (Windows Management Instrumentation) texnologiyasi Windowsga asoslangan kompyuter tarmog'ining turli qismlarini markazlashtirilgan boshqarish va monitoring qilish uchun Microsoft kompaniyasining asosiy texnologiyalaridan biri hisoblanadi. WMI birinchi navbatda Windows ma'murlari, shuningdek, dasturiy ta'minot ishlab chiquvchilari uchun foydali texnologiya hisoblanadi. WMI texnologiyasi bu korporativ boshqaruv modelini Web (Web-Based Enterprise Management, WBEM) asosida amalga oshirish bo‘lib, u nafaqat Microsoft, balki bir qator boshqa kompaniyalar ishtirokida ishlab chiqilgan. WBEMning vazifasi – bu ma'lum bir uskuna, tarmoq infratuzilmasi, operatsion tizim, fayl tizimi va boshqalarga bog'liq bo'lmagan holda korxonalar axborot muhitini masofadan boshqaruvchi standartlarni ishlab chiqishdan iborat. WBEM umumiy axborot modeli (CIM - Common Information Model) sxemasini taklif qiladi, u yagona kengaytiriladigan ob'ektga yo'naltirilgan model sifatida kompyuter tizimining tuzilishini ifodalaydi va WMI tomonidan qo'llab-quvvatlanadi.

WMI va boshqa ActiveX texnologiyalaridan (masalan, ma'lumotlar bazalariga kirish ruxsatini beruvchi ActiveX Data Object (ADO) yoki kataloglar xizmati bilan ishlash uchun Active Directory Service Interface (ADSI)) foydalanganda WMI ni Windows Script Host (WSH) skriptlari yordamida avtomatlashtirishingiz mumkin. Barcha WSH skript misollari VBScriptda yozilgan bo'ladi.

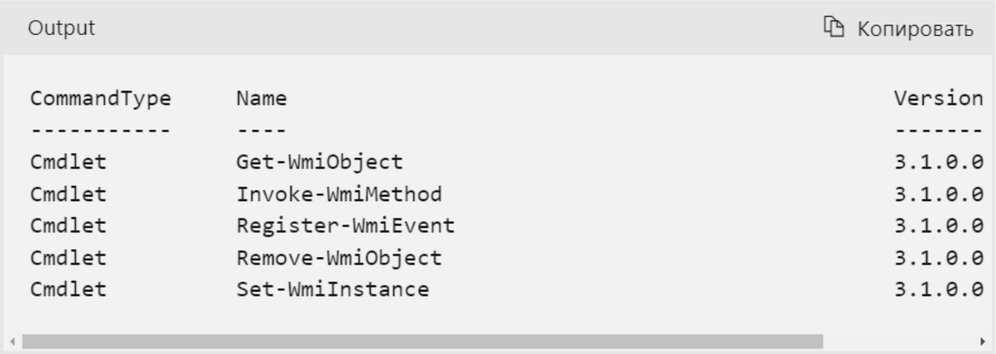
Введение в Windows Management Instrumentation. (WMI) https://script-coding.com/WMI.html

Odatda Windows Management Instrumentation (WMI) kabi texnologiyalar PowerShell cmdletlari bilan foydalanish mumkin. PowerShell-da qo'shimcha dasturiy ta'minot yoki modullarni o'rnatishga hojat qoldirmasdan ishlatiladigan bir nechta lokal WMI cmdletlari mavjud.

PowerShell ishlab chiqilganidan buyon WMI instrumentlari bilan ishlay oladigan cmdletlari mavjud. Masalan, PowerShelldagi `Get-Command` orqali WMI cmdletlarini aniqlash mumkin. Quyidagi PowerShell 5.1 natijalari Windows 10 o’rnatilgan kompyuterdan olingan. Natijalar siz foydalanayotgan PowerShell versiyasiga qarab farq qilishi mumkin.



```
Get-Command -Noun WMI*
```



```
CommandType     Name                                               Version
-----------     ----                                               -------
Cmdlet          Get-WmiObject                                      3.1.0.0
Cmdlet          Invoke-WmiMethod                                   3.1.0.0
Cmdlet          Register-WmiEvent                                  3.1.0.0
Cmdlet          Remove-WmiObject                                   3.1.0.0
Cmdlet          Set-WmiInstance                                    3.1.0.0
```

https://learn.microsoft.com/ru-ru/powershell/scripting/learn/ps101/07-working-with-wmi?view=powershell-7.3

`$PSVersionTable`

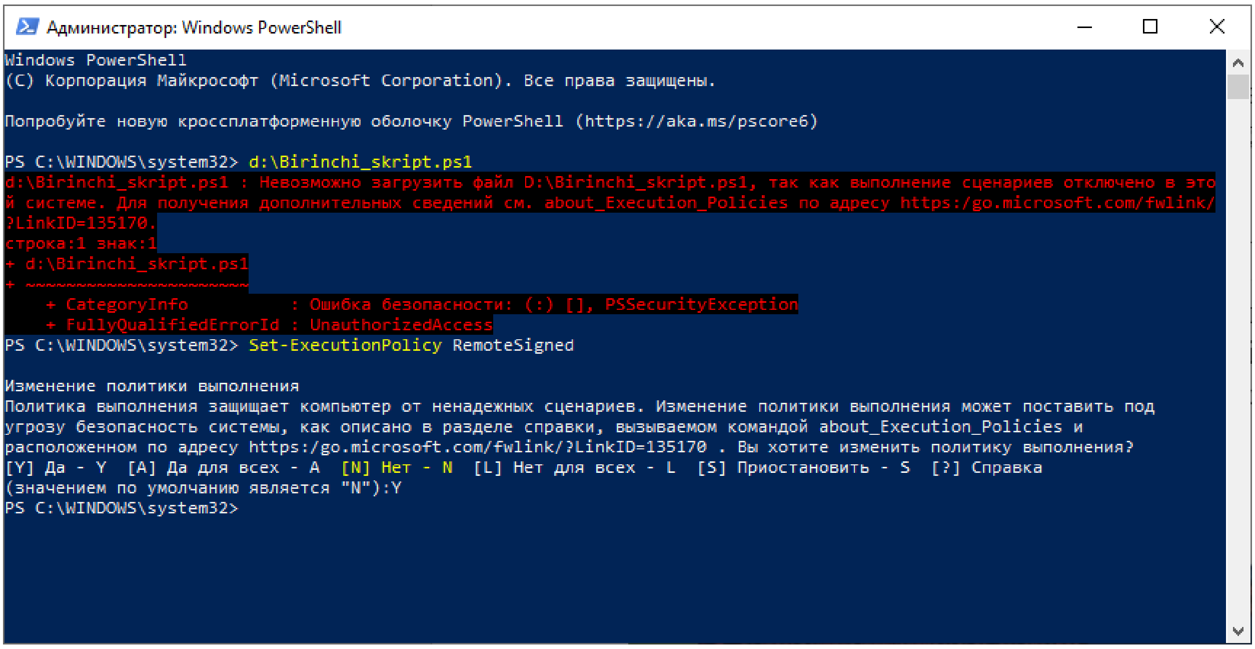
https://info-comp.ru/sisadminst/546-windows-powershell-basics.html

Windows PowerShellda skriptlar yaratish

Windows PowerShell - Bu buyruqlar qatori (cmd)ga o'xshaydi, lekin u yanada samarali buyruqlar qatori interfeysi (CLI) hisoblanib, keng vositalar to'plamidan iborat hamda skriptlari vositasida yanada moslashuvchan va boshqaruvni ta'minlay oladi.

Skript – PowerShell tushunadigan, berilgan ketma-ketlikda bajarishi mumkin bo'lgan, oddiy matnli faylda (.ps1 kengaytmasi bilan) saqlanadigan buyruqlar to'plamidir.

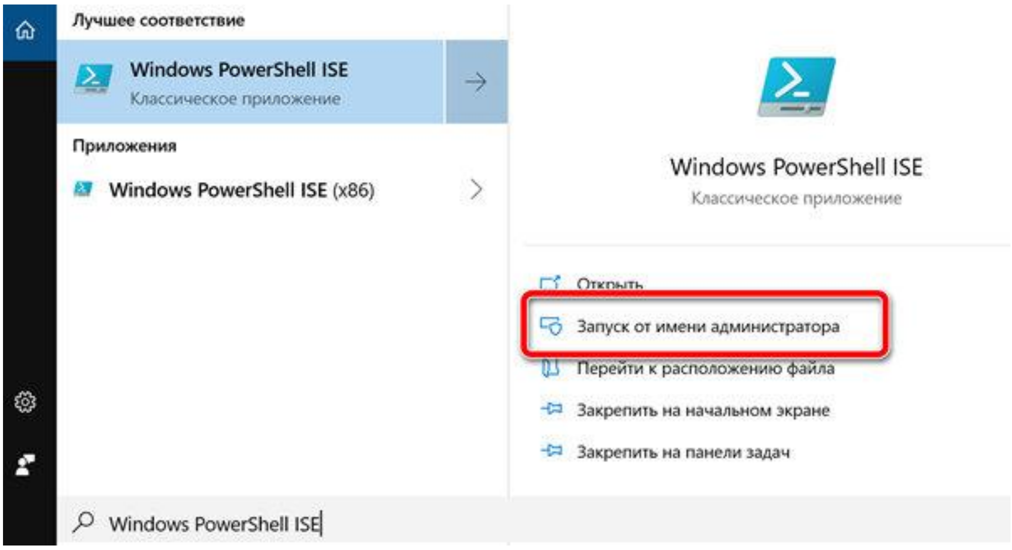
Windowsda .ps1 faylini ikki marta bosish bilan u ishga tushurilmaydi. Bu fayllar PowerShellda skriptni ochish va ishga tushirish orqali amalga oshiriladi. Agar siz «не может быть загружен, потому что запрещено выполнение сценариев в этой системе» ("yuklab bo'lmaydi, chunki bu tizimda skript yaratishga ruxsat berilmaydi") degan xatolik xabarini ko’rsangiz, siz faqat to'g'ri bajarish siyosatini (Set-ExecutionPolicy RemoteSigned buyrug’ini yozib) qo’shib qo’yishingiz kerak.



```
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)

PS C:\WINDOWS\system32> d:\Birinchi_skript.ps1
d:\Birinchi_skript.ps1 : Невозможно загрузить файл D:\Birinchi_skript.ps1, так как выполнение сценариев отключено в это
й системе. Для получения дополнительных сведений см. about_Execution_Policies по адресу https:/go.microsoft.com/fwlink/
?LinkID=135170.
строка:1 знак:1
+ d:\Birinchi_skript.ps1
+ ~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
    + FullyQualifiedErrorId : UnauthorizedAccess
PS C:\WINDOWS\system32> Set-ExecutionPolicy RemoteSigned

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под
угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies и
расположенном по адресу https:/go.microsoft.com/fwlink/?LinkID=135170 . Вы хотите изменить политику выполнения?
[Y] Да - Y  [A] Да для всех - A  [N] Нет - N  [L] Нет для всех - L  [S] Приостановить - S  [?] Справка
(значением по умолчанию является "N"):Y
PS C:\WINDOWS\system32>
```

Skriptlarni yaratish uchun PowerShell ISE konsol ilovasidan foydalanish bosqichlarini quyida ko’rib chiqamiz:

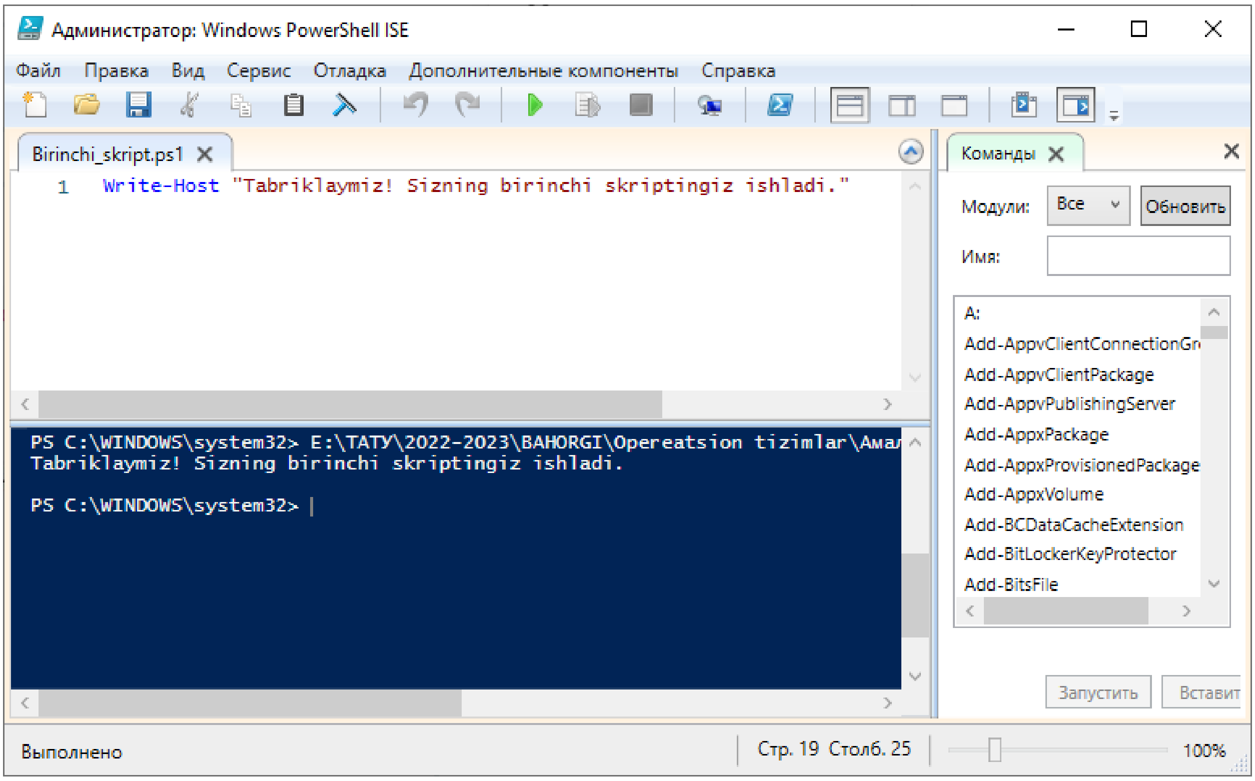
1. Tizim qidiruvi orqali **Windows PowerShell ISE**ni yozing, uning ustiga sichqoncha o'ng tugmasini bosing va ‘**Запуск от имени администратора’**ni tanlang.



2. PowerShell ISEda skript yaratish yoki joylashtirish uchun bo'sh .ps1 faylini yarating. Masalan, unga quyidagi buyruqni yozing va faylni (Birichi_skript.ps1) diskga saqlang:

```
Write-Host "Tabriklaymiz! Sizning birinchi skriptingiz ishladi."
```

Ishga tushiring.



```
Write-Host "Tabriklaymiz! Sizning birinchi skriptingiz ishladi."
```

```
PS C:\WINDOWS\system32> E:\TATY\2022-2023\BAHORGI\Opereatsion tizimlar\Ама
Tabriklaymiz! Sizning birinchi skriptingiz ishladi.

PS C:\WINDOWS\system32> |
```

Yuqoridagi skriptda har bir qatoriga buyruqlarni ketma-ket yozish mumkin. PowerShell skriptini ishga tushurganba, buyruqlar ketma-ket bajariladi.