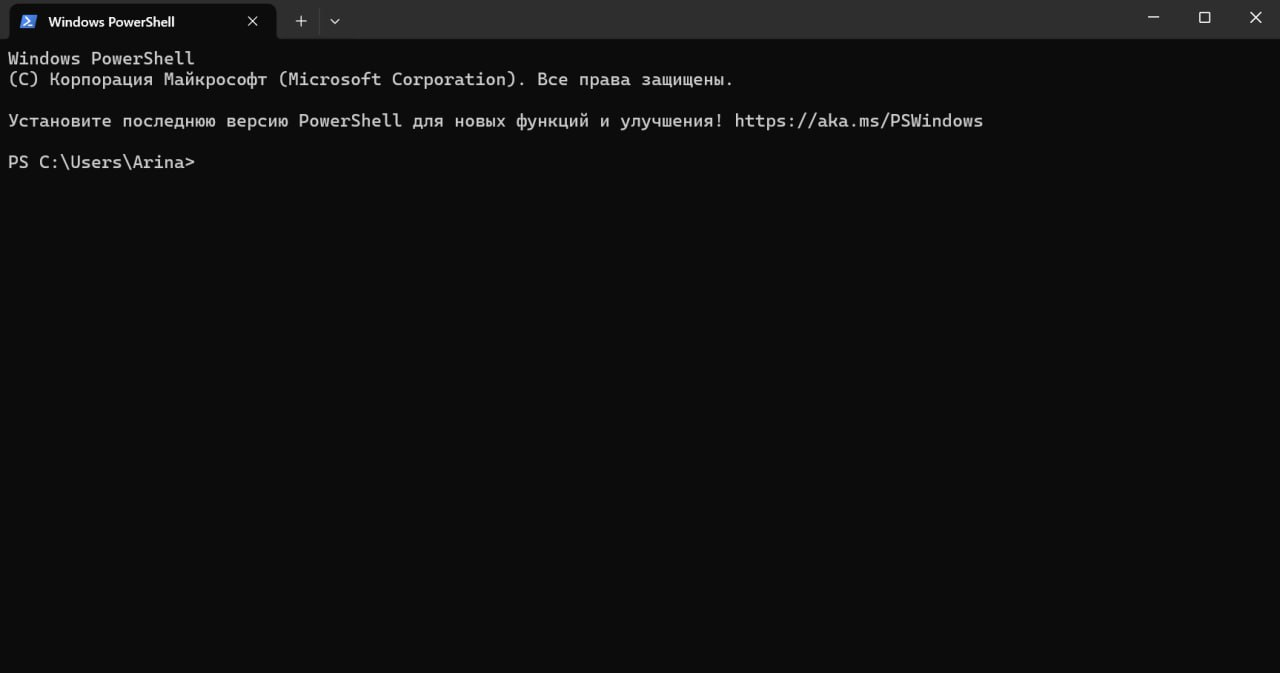
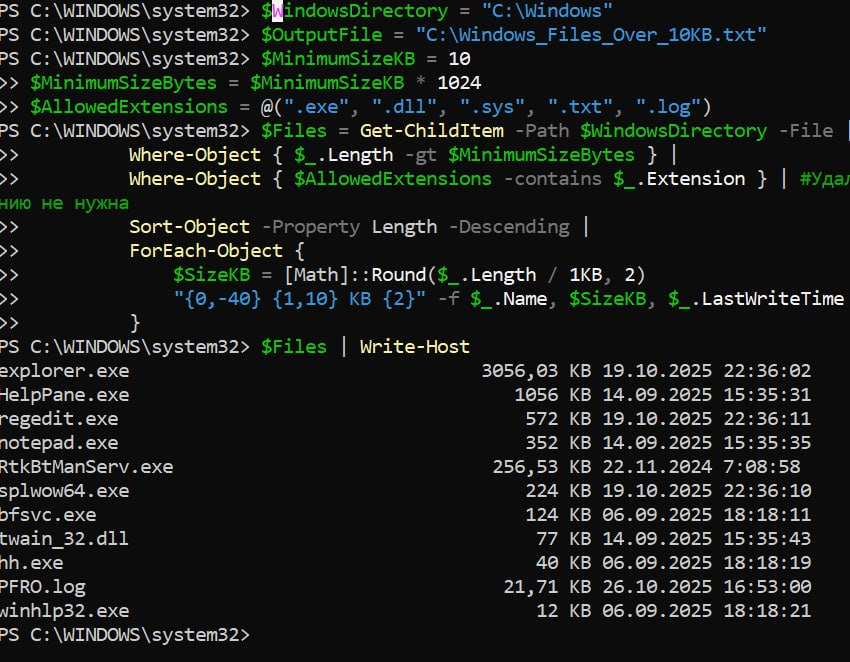
1. открываем оболочку powershell



Задание 4

$env:windir – это переменная окружения, которая содержит путь к каталогу Windows.

2. Get-ChildItem: Командлет Get-ChildItem используется для получения списка всех файлов и папок в указанном каталоге. Параметр -File гарантирует, что будут выбраны только файлы.

3. Where-Object (Фильтрация):

• Первый Where-Object фильтрует файлы, оставляя только те, у которых размер ($\_.Length) больше $MinimumSizeBytes.

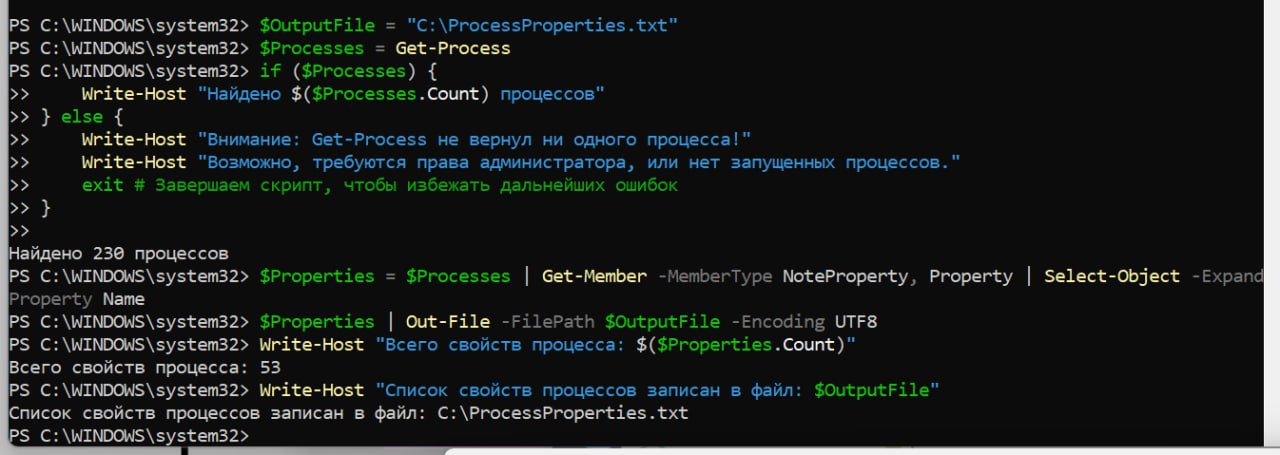
• Второй Where-Object (опционально) фильтрует файлы по расширению. $AllowedExtensions -contains $\_.Extension проверяет, входит ли расширение текущего файла ($\_.Extension) в массив разрешенных расширений $AllowedExtensions. Если расширение не указано, то этот фильтр пропускает все файлы.

4. Sort-Object (Сортировка): Командлет Sort-Object сортирует отфильтрованный список файлов по свойству Length (размеру) в порядке убывания (-Descending).

5. ForEach-Object (Форматирование): Этот блок используется для форматирования вывода. Он вычисляет размер файла в килобайтах и форматирует строку с именем файла, размером и датой последнего изменения. "{0,-40} {1,10} KB {2}" -f $\_.Name, $SizeKB, $\_.LastWriteTime – это оператор форматирования строк PowerShell.

6. Write-Host (Вывод на экран): Write-Host выводит отформатированный список файлов в консоль.

7. Out-File (Вывод в файл



5 задание

1. $OutputFile = "C:\ProcessProperties.txt": Определяет путь к файлу, куда будут записаны свойства процессов.

2. $Processes = Get-Process: Получает список всех запущенных процессов и сохраняет их в переменной $Processes.

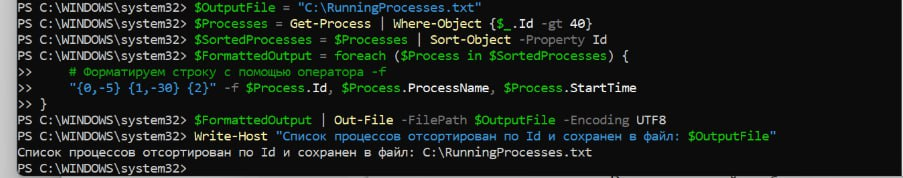
3. $Properties = $Processes | Get-Member -MemberType NoteProperty, Property | Select-Object -ExpandProperty Name: Эта строка делает следующее:

• Get-Member -MemberType NoteProperty, Property: Получает информацию обо всех свойствах и примечаниях каждого процесса из $Processes. Свойства (Property) обычно представляют собой значения

• Select-Object -ExpandProperty Name: Извлекает только имена свойств (свойство Name) из результатов Get-Member. ExpandProperty гарантирует, что $Properties будет массивом строк, а не массивом объектов.

4. $Properties | Out-File -FilePath $OutputFile -Encoding UTF8: Записывает каждое имя свойства из массива $Properties в файл, указанный в $OutputFile. Каждое свойство записывается на новой строке. Кодировка UTF8 обеспечивает корректное отображение всех символов.

5. Write-Host "Всего свойств процесса: $($Properties.Count)": Выводит на экран общее количество свойств процесса, подсчитанное с помощью $Properties.Count.

6 задание

1. $OutputFile = "C:\RunningProcesses.txt": Указывает путь к текстовому файлу, куда будут записаны результаты.

2. $Processes = Get-Process | Where-Object {$\_.Id -gt 40}:

• Get-Process: Получает список всех запущенных процессов.

• Where-Object {$\_.Id -gt 40}: Фильтрует процессы, оставляя только те, у которых Id больше 40. $\_.Id обращается к свойству Id каждого процесса.

3. $SortedProcesses = $Processes | Sort-Object -Property Id: Сортирует отфильтрованные процессы по свойству Id в порядке возрастания (по умолчанию Sort-Object сортирует по возрастанию).

4. Форматирование вывода:

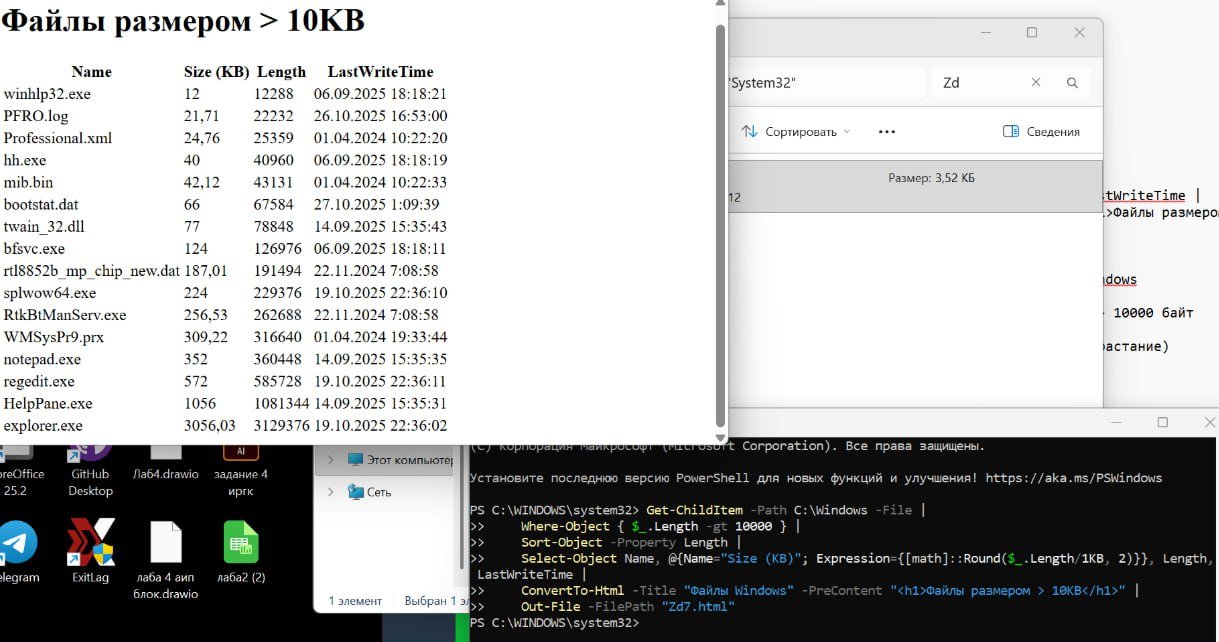
• foreach ($Process in $SortedProcesses): Цикл перебирает каждый процесс в отсортированном списке.

• [PSCustomObject]@{...}: Создает новый пользовательский объект PowerShell (PSCustomObject) для каждого процесса.

• ConvertTo-String: Преобразует каждый PSObject в string, чтобы корректно сохранить в текстовый файл.

5. $FormattedOutput | Out-File -FilePath $OutputFile -Encoding UTF8: Записывает отформатированный вывод (массив строк) в файл, указанный в $OutputFile. -Encoding UTF8 обеспечивает правильную кодировку для всех символов.

6. Write-Host "Список процессов отсортирован по Id и сохранен в файл: $OutputFile": Выводит сообщение об успехе.



7 задание

Get-ChildItem -Path C:\Windows -File - только файлы из Windows

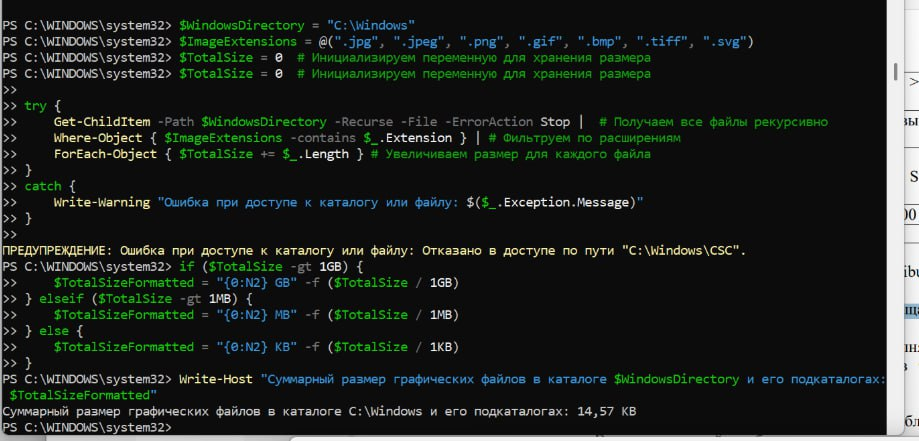
Where-Object { $\_.Length -gt 10000 } - фильтр по размеру > 10000 байт

Sort-Object -Property Length - сортировка по размеру (возрастание)

Select-Object - выбирает параметры для отображения

ConvertTo-Html - создает HTML-формат

Out-File "windows\_files.html" - сохраняет HTML-файл



8 задание

1. $WindowsDirectory = "C:\Windows": Задает путь к каталогу Windows.

2. $ImageExtensions = @(".jpg", ".jpeg", ".png", ".gif", ".bmp", ".tiff", ".svg"): 3. $TotalSize = ...:

• Get-ChildItem -Path $WindowsDirectory -Recurse -File:

\* Get-ChildItem: Получает список элементов (файлов и каталогов) в указанном пути.

\* -Path $WindowsDirectory: Указывает, где искать файлы.

\* -Recurse: Обеспечивает рекурсивный поиск во всех подкаталогах.

\* -File: Фильтрует результат, оставляя только файлы (не каталоги).

• Where-Object { $ImageExtensions -contains $\_.Extension }: Фильтрует файлы, оставляя только те, у которых расширение входит в список $ImageExtensions.

• Measure-Object -Property Length -Sum:

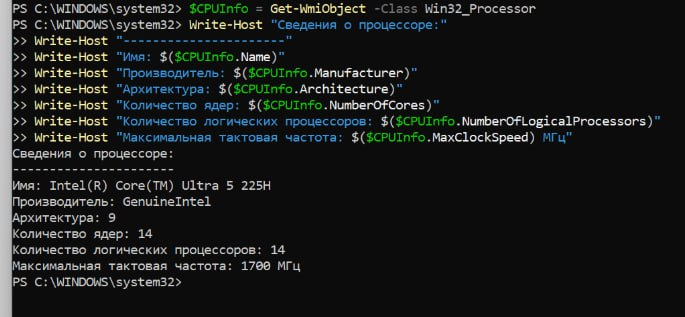
\* Measure-Object: Вычисляет статистику для указанных объектов.

\* -Property Length: Указывает, что нужно вычислять статистику для свойства Length (размер файла в байтах).

\* -Sum: Указывает, что нужно вычислить сумму всех значений свойства Length.

• Select-Object -ExpandProperty Sum: Извлекает только значение суммы из объекта, возвращенного Measure-Object.

5. Write-Host ...: Выводит результат на экран.



9 задание

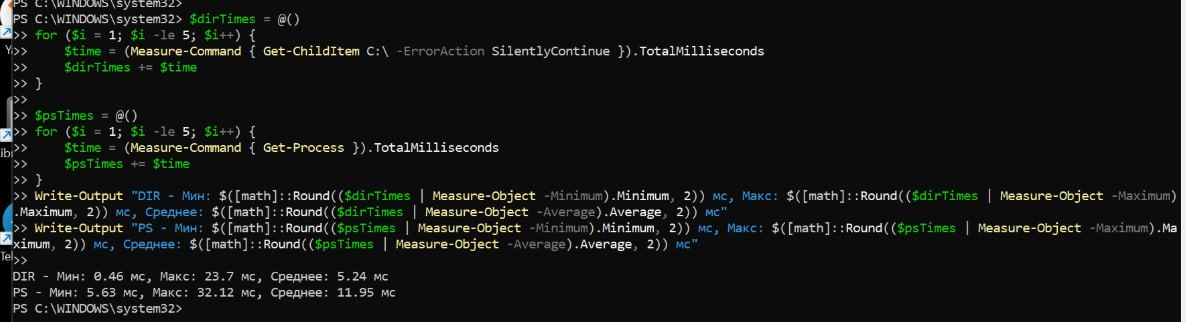
1. $CPUInfo = Get-WmiObject -Class Win32\_Processor:

• Get-WmiObject: Это командлет PowerShell, который используется для получения информации из WMI (Windows Management Instrumentation). WMI предоставляет доступ к различным системным ресурсам и данным.

• -Class Win32\_Processor: Указывает, что нужно получить информацию из класса WMI под названием Win32\_Processor. Этот класс содержит сведения о процессоре компьютера.

2. Write-Host "Сведения о процессоре:" и т.д.: Используются командлеты Write-Host для вывода информации на экран.

• $($CPUInfo.Name): Использует синтаксис расширения переменной PowerShell ($()) для доступа к свойствам объекта $CPUInfo (например, имени процессора, производителю, и т.д.). Каждое свойство извлекается и отображается в консоли.



10 задание

Программа покажет статистику времени выполнения для обеих команд - минимальное, максимальное и среднее время в миллисекундах.