PS 2 Задачи 2018 02

**Задание 2.1**

1. Вывести на экран все свойства объекта (**Property**) который, описывает процессы, выполняемые на компьютере (**Get-Process**).
2. Вывести на экран все методы (**Method**) работы с процессами.
3. Вывести на экран свойства процессов и методы работы с ними.

**Задание 2.2**

1. Найти директорию, где имеется достаточно файлов.
2. Вывести содержимое найденной директории. Список выводимых свойств: CreationTime, Name, Mode, Length.

**Задание 2.3**

1. Найти директорию, где имеется достаточно файлов.
2. Вывести содержимое найденной директории. Список выводимых свойств: **CreationTime**, **Name**, **Mode**, **Length**.
3. Вывести на экран информацию из п.2 упорядочив по времени создания объекта (**CreationTime**).
4. Вывести на экран информацию из п.2 упорядочив её по имени (**Name**) в обратном порядке (параметр **−Descending**).

**Задание 2.4**

1. Вывести список, установленного на компьютере ПО. Список выводимых свойств: **Name, Vendor, InstallDate**.
2. Упорядочить выводимую информацию по дате инсталляции ПО (**InstallDate**) по убыванию (параметр **−Descending**).
3. Для п.2, только 5 последних установок.
4. Для п.3, добавить 3 самые ранние установки.

**Задание 2.5**

1. Вывести список процессов (**Get-Process**).
2. Посмотреть свойства, описывающие процессы (**Get-Member** )
3. Вывести только имя процесса, идентификатор и процессное время.
4. Полученный список процессов упорядочить по убыванию процессного времени.
5. Для п.4, показать 6 самых затратных процессов (максимальное процессное время).

**Задание 2.6**

1. Занести произвольную текстовую строку в переменную, имя которой начинается с символа **V**.
2. Создать директорию «№гр\_Фамилия». (**New-Item**)
3. Вывести в файл (в созданной директории) список всех переменных, начинающихся с символа **V**. (**dir** с диска **Variable:**)
4. Вывести на экран содержимое созданного файла. (**type**)

**Задание 2.7**

Для следующих операции занести в файл:

**$v\_int = 150 + 401**

**$v\_str = “Строка символов”**

**$v\_log = ( 2 -eq 2 )**

**$v\_obj = dir | Select-Object -First 5**

1. Результат каждой операции.
2. После результата каждой операции записать тип результата через **GetType().Name**, а потом через **Get-Member**
3. Вывести на экран содержимое созданного файла.

**Задание 2.8**

1. В первую переменную поместить в одинарных кавычках (‘’) строку, в которой помимо текста присутствуют две переменные.
2. Присвоить второй переменной числовое значение, а третей – символьное значение.
3. Вывести на экран содержимое первой переменной.
4. Повторить п.1 с двойными кавычками (“”) и п.3.

**Задание 2.9**

1. Поместить строку из 4 символов в переменную **$var\_1.**
2. Повторить в переменной эту строку 3 раза.
3. Добавить в начало этой строки слово ‘**Старт**’, а в конец – слово ‘**Финиш**’.
4. Вывести на экран содержимое созданной переменной.

**Задание 2.10**

Изменить функцию **Hello**, чтобы она запрашивала у пользователя его имя и вводила:

**Добрый день,** *имя\_пользователя*  **!!!**

**Задание 2.11**

Написать функцию:

1. Узнает имя пользователя и приветствует его.
2. Ввести числовое значения двух переменных.
3. Вывести на консоль результат деления первого числа на второе.

**Задание 2.12**

Написать функцию:

1. Узнает имя пользователя и приветствует его.
2. Ввести числовое значения двух переменных.
3. Вывести в файл строку:

*число\_1* **/** *число\_2* **=** *результат\_деления*

**Задание 2.13**

Написать функцию:

1. Узнает имя пользователя и приветствует его.
2. Ввести числовое значения двух переменных.
3. Вывести в файл шесть строк показывающие все операции сравнения с этими числами:

*число\_1 оп\_ср**число\_2 –>* *результат\_сравнения*

Например:

15 < 25 –*>* True

**Задание 2.14**

1. Занести в переменную шаблон **’?a\*’**.
2. Сравнить с шаблоном следующие строки:

*a, пустая строка, ba, ab, abc, babc, 1avatar*

**Задание 2.15**

1. Создать текстовый файл на диске с расширением **TXT***.* В этот файл вывести содержимое корневого каталога диска **C:**
2. Сохранить в переменной **$ps\_drive** список дисков.
3. Записать в конец созданного в п.1 файла содержимое из переменной **$ps\_drive**
4. Вывести на экран содержимое созданного файла.

**Задание 2.16**

1. Создайте каталог с именем «№гр\_Фамилия».
2. В созданном каталоге создайте несколько текстовых файлов и поместите в них несколько строк.
3. Сохраните содержимое созданного каталога в переменной **$ps\_dir**, а список дисков в переменной **$ps\_drive**.
4. Выведите на экран содержимое переменных **$ps\_dir** и **$ps\_drive**.

**Задание 2.17**

1. Вывести на экран список выполняемых процессов (**Get-Process**) по маске имени, в которое входит слово **word** (\* *word* \*).
2. Запустить программу **Word**: найти каталог, где расположен файл **WinWord.exe** и выполнить его.
3. Вывести на экран список выполняемых процессов (**Get-Process**) по маске имени, в которое входит слово **word** (\* *word* \*).
4. Остановить процесс **WinWord** (**Stop-Process**) по его идентификатору (**id**).

**Задание 2.18**

1. Вывести на экран список сервисов (**Get-Sevice**).
2. Посмотреть структуру объектов, описывающих сервисы (**Get-Member**).
3. Вывести информацию о 15 сервисах, упорядочив её по названию сервисов (**Sort-Object**). В выводимой информации отразить имя сервиса, статус, **DisplayName** (**Select-Object**).
4. Повторить п.3 с следующим ограничением: вывести на экран список сервисов, имена которых начинается на букву ‘w’ (*маска* **w\***).

**Задание 2.19**

1. Вывести список всех выполняемых процессов. (**Get-Process**)
2. Вывести список всех выполняемых процессов, упорядочив его по убыванию процессорного времени. (**Sort-Object**)

**Задание 2.20**

1. Вывести список 10 процессов, которые максимально используют время центрального процессора. (**Get-Process**)
2. Занести этот список в переменную.(**=**)
3. Создать «№гр\_Фамилия» директорию.(**New-Item**) В ней создать файл в который записать содержимое из переменной п.2.(**>**)
4. Вывести на экран содержимое созданного файла.(**type**)

**Задание 2.21**

Написать функцию:

1. Ввести числовое значения трех переменных.
2. Вывести в файл три строки показывающие следующие операции с этими числами:

*число\_1* **<** *число\_2* **и** *число\_2* **<** *число\_3* **=** *результат*

*число\_1* **<** *число\_2* **или** *число\_2* **<** *число\_3* **=** *результат*

**не****(** *число\_1* **<** *число\_3* **) =** *результат*

Например:

15 < 25 и 25 < -5 = False

**Задание 2.22**

1. Вывести список всех сервисов (**Get-Service**).
2. Используя **Where-Object** вывести сервисы, которые остановлены.

**Задание 2.23**

1. Создать свой каталог с именем «№гр\_Фамилия».
2. Найти каталог, в котором имеются и подкаталоги и файлы.
3. Вывести в файл в каталоге «№гр\_Фамилия» список всех подкаталогов из п.2.(**dir**, **>**, **where**)
4. В конец этого файла добавить список всех файлов из п.2. (**dir**, **>**, **where**)
5. Содержимое созданного файла вывести на экран. (**type**)

**Задание 2.24**

1. Вывести в файл список всех процессов с положительным временем процессора (**ps**, **>**, **where**)
2. В конец этого файла добавить список процессов, которые более 10 сек. занимали процессор (**ps**, **>**, **where**)
3. Содержимое созданного файла вывести на экран. (**type**)

**Задание 2.25**

1. Вывести в файл список всех процессов с положительным временем процессора (**ps**, **>**, **where**)
2. В конец этого файла добавить список процессов, у которых в имени компании (**Company**) присутствует слово «**Microsoft**». По каждому процессу вывести информацию: *имя, идентификатор, время старта, размер процессного времени, имя компании*.
3. Содержимое созданного файла вывести на экран. (**type**)