МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет экономики, менеджмента и информационных технологий

Кафедра Систем управления и информационных технологий в строительстве

КУРСОВОЙ ПРОЕКТ

по дисциплине: Операционные системы

Тема: «Получение списка установленных приложений с помощью скрипта»

Расчетно-пояснительная записка

Разработал студент Е.А. Рыженков

Подпись, дата Инициалы, фамилия

Руководитель К.А. Маковий

Подпись, дата Инициалы, фамилия

Члены комиссии

Подпись, дата Инициалы, фамилия

Подпись, дата Инициалы, фамилия

Нормоконтролер

Подпись, дата Инициалы, фамилия

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата

2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Кафедра Систем управления и информационных технологий в строительстве

ЗАДАНИЕ

на курсовой проект

по дисциплине: Операционные системы

Тема: «Получение списка установленных приложений с помощью скрипта»

Студент бИСТ-221 Рыженков Евгений Александрович

Группа, фамилия, имя, отчество

Вариант 38

Технические условия процессор AMD Ryzen 5 2600 3.7 ГГц, операционная система Windows 10, ОЗУ 16 384 МБ

Содержание и объем работы (графические работы, расчеты и прочее):

25 стр, 6 рисунков, 1 приложение, 1 таблица

Сроки выполнения этапов: анализ и постановка задачи (01.03-15.03); изучение теоретического обоснования работы (15.03-15.04); реализация программного решения (15.04-15.05); оформление пояснительной записки (15.05-05.06).

Срок защиты курсового проекта:

Руководитель К.А. Маковий

Подпись, дата Инициалы, фамилия

Задание принял студент Е.А. Рыженков \_

Подпись, дата Инициалы, фамилия

Замечания руководителя

Содержание

[ВВЕДЕНИЕ 5](#_Toc136893962)

[1 Реестры в windows 6](#_Toc136893963)

[1.1 Общие сведения о реестрах 6](#_Toc136893964)

[1.2 Ветка HKLM и подготовка системы для создания скрипта 15](#_Toc136893965)

[2 Разработка скрипта для получения списка установленных приложений 15](#_Toc136893966)

[2.1 Постановка задачи курсового проекта 15](#_Toc136893967)

[2.2 Обоснование средств разработки 15](#_Toc136893968)

[2.3 Описание работы системы 16](#_Toc136893969)

[ЗАКЛЮЧЕНИЕ 18](#_Toc136893970)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 19](#_Toc136893971)

[ПРИЛОЖЕНИЕ А 21](#_Toc136893972)

# ВВЕДЕНИЕ

В данном курсовом проекте я рассматриваю, как можно создать скрипт для получения списка всех установленных на компьютере приложений, закрепляю знания, получаемые в процессе изучения дисциплины, приобретаю необходимые практические навыки самостоятельной работы с операционной системой Windows.

Целью данного курсового проекта является получение списка всех установленных на компьютере приложений.

Выбранная мной тема является актуальной для многих продвинутых пользователей операционной системы Windows, которые хотели бы ускорить и облегчить процесс администрирования операционной системы.

В работе используется интегрированная среда сценариев Windows называемая PowerShell Integrated Scripting Environment— это поставляемая с пакетом PowerShell оболочка разработки, дополненная отладчиком и редактором сценариев.

# 1 Реестры в windows

# 1.1 Общие сведения о реестрах

Реестр Windows представляет собой иерархическую базу данных, в которой хранятся низкоуровневые настройки для операционной системы Microsoft Windows и приложений, которые предпочитают использовать реестр. Реестр могут использовать ядро, драйверы устройств, службы, диспетчер учетных записей безопасности и пользовательские интерфейсы. Реестр также предоставляет доступ к счетчикам для профилирования производительности системы [1].

Другими словами, реестр Windows содержит информацию, настройки, опции и другие значения для программ и оборудования, установленных во всех версиях операционных систем Microsoft Windows. Например, когда программа установлена, в реестр Windows добавляется новый подраздел, содержащий такие параметры, как местоположение программы, ее версия и способ запуска программы.

В Windows 10, согласно сведениям из HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\hivelist файлы реестра хранятся в следующих местах:

* 01 = Ветка реестра «HKEY\_LOCAL\_MACHINE\HARDWARE» формируется в зависимости от оборудования (динамически);
* 02 = Ветка реестра «HKEY\_LOCAL\_MACHINE\BCD00000000» формируется из файла «%SystemRoot%\Boot\BCD»
* 03 = Ветка реестра «HKEY\_LOCAL\_MACHINE\SYSTEM» формируется из файла «%SystemRoot%\System32\config\SYSTEM»
* 04 = Ветка реестра «HKEY\_LOCAL\_MACHINE\SOFTWARE» формируется из файла «%SystemRoot%\System32\config\SOFTWARE»
* 05 = Ветка реестра «HKEY\_LOCAL\_MACHINE\SECURITY» формируется из файла «%SystemRoot%\System32\config\SECURITY»
* 06 = Ветка реестра «HKEY\_LOCAL\_MACHINE\SAM» формируется из файла «%SystemRoot%\System32\config\SAM»
* 07 = Ветка реестра «HKEY\_USERS\.DEFAULT» формируется из файла «%SystemRoot%\System32\config\DEFAULT»
* 08 = Ветка реестра «HKEY\_USERS\S-1-5-18» формируется из файла «%SystemRoot%\System32\config\systemprofile\NTUSER.DAT» (относится к учётной записи system)[1]
* 09 = Ветка реестра «HKEY\_USERS\S-1-5-19» формируется из файла «%SystemRoot%\ServiceProfiles\LocalService\NTUSER.DAT» (относится к учётной записи LocalService)
* 10 = Ветка реестра «HKEY\_USERS\S-1-5-20» формируется из файла «%SystemRoot%\ServiceProfiles\NetworkService\NTUSER.DAT» (относится к учётной записи NetworkService)
* 11 = Ветка реестра «HKEY\_USERS\<SID\_пользователя>» формируется из файла «%USERPROFILE%\NTUSER.DAT», где под %USERPROFILE% подразумевается путь к папке соответствующего пользователя.
* 12 = Ветка реестра «HKEY\_USERS\<SID\_пользователя>\_Classes» формируется из файла «%USERPROFILE%\AppData\Local\Microsoft\Windows\UsrClass.dat»

Резервные копии файлов реестра DEFAULT, SAM, SECURITY, SOFTWARE и SYSTEM находятся в папке «%SystemRoot%\System32\config\RegBack». Само резервное копирование производится силами Планировщика задач в 0 ч. 00 мин. каждые 10 дней по заданию «RegIdleBackup», расположенному в иерархии задач по пути «\Microsoft\Windows\Registry». Ветка реестра «HKEY\_CURRENT\_USER» формируется из файла «%USERPROFILE%\NTUSER.DAT», где под %USERPROFILE% подразумевается путь к папке текущего пользователя. Реестр 64-разрядных версий Windows XP и Windows Server 2003 и Windows Vista подразделяется на 32- и 64-разрядные разделы. Большинство 32-разрядных разделов имеют те же имена, что и их аналоги в 64-разрядном разделе, и наоборот [14]. По умолчанию редактор реестра 64-разрядных версий Windows XP, Windows Server 2003 и Windows Vista отображает 32-разрядные разделы в узле HKEY\_LOCAL\_MACHINE\Software\WOW6432Node.

Описание разделов реестров:

* HKEY\_CURRENT\_USER. Данный раздел содержит настройки текущего активного пользователя, вошедшего в систему. Здесь хранятся папки пользователя, цвета экрана и параметры панели управления. Эти сведения сопоставлены с профилем пользователя. Вместо полного имени раздела иногда используется аббревиатура HKCU. Хотя этот раздел выглядит как один из основных в редакторе реестра, он является всего лишь ссылкой на один из профилей HKEY\_USERS\.
* HKEY\_USERS. Раздел HKEY\_USERS (псевдоним HKU) содержит информацию о профилях всех пользователей данного компьютера. Данный раздел практически никогда не используется пользователями. Следует отметить связь данного корневого раздела с разделом HKEY\_CURRENT\_USER, который фактически является ссылкой подраздела корневого раздела HKEY\_USERS, хранящего сведения о текущем пользователе.
* HKEY\_LOCAL\_MACHINE. Раздел содержит параметры конфигурации, относящиеся к данному компьютеру (для всех пользователей). Вместо полного имени раздела иногда используется аббревиатура HKLM.
* HKEY\_CLASSES\_ROOT. Является подразделом HKEY\_LOCAL\_MACHINE\Software\Classes. В основном, содержит информацию о зарегистрированных типах файлов и объектах COM и ActiveX. Вместо полного имени раздела иногда используется аббревиатура HKCR. Начиная с Windows 2000 эти сведения хранятся как в LOCAL\_MACHINE, так и в CURRENT\_USER. Раздел HKLM\Software\Classes содержит параметры по умолчанию, которые относятся ко всем пользователям локального компьютера. Параметры, содержащиеся в разделе HKCU\Software\Classes, относятся только к текущему пользователю, имея приоритет над стандартными. Раздел CLASSES\_ROOT включает в себя данные из обоих источников. Кроме того, раздел CLASSES\_ROOT предоставляет объединённые данные программам, написанным под ранние версии Windows. Изменения настроек текущего пользователя выполняются в разделе HKCU\Software\Classes. Модификация параметров по умолчанию должна производиться в разделе HKLM\Software\Classes. Данные из разделов, добавленных в CLASSES\_ROOT, будут сохранены системой в разделе HKLM\Software\Classes. Если изменяется параметр в одном из подразделов раздела CLASSES\_ROOT и такой подраздел уже существует в HKCU\Software\Classes, то для хранения информации будет использован раздел HKCU\Software\Classes, а не аналогичный раздел в LOCAL\_MACHINE.
* HKEY\_CURRENT\_CONFIG. Данный раздел содержит сведения о профиле оборудования, используемом локальным компьютером при запуске системы. Является ссылкой на HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Hardware Profiles\Current
* HKEY\_DYN\_DATA. Данный раздел имеется только в реестре ОС семейства Windows 9x/ME. Содержит динамически изменяемые данные о компьютере (загрузка процессора, размер файла подкачки и т. п.).

В таблице 1 перечислены типы данных, которые в настоящее время определены и используются Windows. Максимальная длина имени значения выглядит следующим образом [2]:

* Windows Server 2003, Windows XP и Windows Vista: 16 383 символа
* Windows 2000: 260 символов ANSI или 16 383 символа Юникода
* Windows Millennium Edition/Windows 98/Windows 95: 255 символов

Длинные значения (более 2048 байт) должны храниться в виде файлов с именами файлов, хранящимися в реестре. Это помогает реестру работать эффективно. Максимальный размер значения выглядит следующим образом [13]:

* Windows NT 4.0/Windows 2000/Windows XP/Windows Server 2003/Windows Vista: доступная память
* Windows Millennium Edition/Windows 98/Windows 95: 16 300 байт

|  |  |  |
| --- | --- | --- |
| Таблица 1 – типы данных в windows | | |
| Имя | Тип данных | Описание |
| Двоичное значение | REG\_BINARY | Необработанные двоичные данные. Большинство сведений о компонентах оборудования хранятся в виде двоичных данных и отображаются в редакторе реестра в шестнадцатеричном формате. |
| Параметр DWORD | REG\_DWORD | Данные, представленные числом длиной 4 байта (32-разрядное целое число). Многие параметры драйверов и служб устройств имеют этот тип и отображаются в редакторе реестра в двоичном, шестнадцатеричном или десятичном формате. Связанные значения DWORD\_LITTLE\_ENDIAN (младший значащий байт находится по самому младшему адресу) и REG\_DWORD\_BIG\_ENDIAN (младший значащий байт находится по самому старшему адресу). |
| Расширяемый строковый параметр | REG\_EXPAND\_SZ | Строка данных переменной длины. Этот тип данных включает переменные, которые разрешены при использовании данных программой или службой. |
| Многострочный параметр | REG\_MULTI\_SZ | Несколько строк. Значения, содержащие списки или несколько значений в форме, доступной для чтения пользователями, обычно являются этим типом. Записи разделяются пробелами, запятыми или другими знаками. |
| Строковый параметр | REG\_SZ | Текстовая строка фиксированной длины. |
| Двоичное значение | REG\_RESOURCE\_LIST | Ряд вложенных массивов, предназначенных для хранения списка ресурсов, используемых драйвером аппаратного устройства или одним из физических устройств, которые он контролирует. Эти данные обнаруживаются и записываются системой в дерево \ResourceMap и отображаются в редакторе реестра в шестнадцатеричном формате в виде двоичного значения. |
| Двоичное значение | REG\_RESOURCE\_REQUIREMENTS\_LIST | Ряд вложенных массивов, предназначенных для хранения списка возможных аппаратных ресурсов, используемых драйвером устройства или одного из физических устройств, которые он контролирует. Система записывает подмножество этого списка в дерево \ResourceMap. Эти данные обнаруживаются системой и отображаются в редакторе реестра в шестнадцатеричном формате в виде двоичного значения. |
| Двоичное значение | REG\_FULL\_RESOURCE\_DESCRIPTOR | Ряд вложенных массивов, предназначенных для хранения списка ресурсов, используемых физическим аппаратным устройством. Эти данные обнаруживаются и записываются системой в дерево \HardwareDescription и отображаются в редакторе реестра в шестнадцатеричном формате в виде двоичного значения. |
| Нет | REG\_NONE | Данные без определенного типа. Эти данные записываются в реестр системой или приложениями и отображаются в редакторе реестра в шестнадцатеричном формате в виде двоичного значения. |
| Ссылка | REG\_LINK | Строка Юникода, именуемая символьной ссылкой. |
| Параметр QWORD | REG\_QWORD | Данные, представленные числом, которое является 64-разрядным целым числом. Эти данные отображаются в редакторе реестра в виде двоичного значения и были введены в Windows 2000. |

Способ хранения параметров и настроек операционной системы при помощи реестра Windows часто подвергается критике по следующим причинам [3]:

* Реестр хранится в небольшом количестве объёмных (по нескольку сотен мегабайт) файлов, вследствие чего высока вероятность повреждения данных. Также большие файлы подвержены фрагментации, что замедляет доступ к реестру на обычных жёстких дисках (твердотельные накопители решают эту проблему).
* В связи с тем, что, помимо настроек, в реестре хранится различная информация системы и приложений (например, многие приложения хранят в реестре список недавно открытых файлов), размер реестра значительно увеличивается по мере использования операционной системы. Эта проблема частично решается при помощи специальных утилит.
* Не все настройки системы хранятся в реестре, соответственно, перенос настроек системы путём копирования реестра невозможен.
* Некоторые программы не могут работать без параметров, занесённых в реестр, что создаёт трудности при переносе их с компьютера на компьютер, или теряют данные после переустановки операционной системы.
* Реестр очень похож на виртуальную файловую систему, работающую поверх реальной файловой системы. При этом, как для файловой системы, реестр очень неустойчив к сбоям. Фактически один неверный байт в файле куста (ветки реестра, подобие корневому каталогу в файловых системах) приводит к сбою при попытке загрузки куста, и (для системных кустов) отказу операционной системы. Надёжность такого реестра никогда не будет выше надёжности файловой системы, на которой расположены его файлы. Однако версии Windows начиная с NT используют логи транзакций в качестве средства защиты от повреждений при обновлении реестра. Текущие версии Windows используют логи двух уровней для того, чтобы обеспечить целостность реестра в случае отказа питания или других подобных событий во время обновления реестра. Даже в случае ошибки, из-за которой невозможно восстановить реестр, Windows может починить и инициализировать повреждённые записи реестра во время загрузки системы.

# 1.2 Подготовка системы для создания скрипта

Для создания и выполнения скриптов необходимы некоторые условия [4]:

* пользователь должен обладать правами администратора;
* необходимо настроить политику запуска скриптов.

Для того чтобы выдать пользователю права администратора существует несколько способов [5]:

1. нажать сочетание клавиш «WIN + R» для открытия окна «Выполнить». В появившемся окне ввести «gpedit.msc». Откроется редактор групповой политики. Далее, зайдя в раздел, расположенный слева, под названием «Параметры безопасности» надо активировать параметр «Состояние учётной записи Администратора». Это делается двойным кликом. Повышение прав доступа станет возможным после перезагрузки системы. Окно для выдачи прав администратора представлено на рисунке 1.

|  |
| --- |
|  |
| Рисунок 1 – Выдача прав администратора через редактор локальной групповой политики |

1. если ваша версия Windows имеет статус профессиональной, можно применить и метод учётных записей. Их параметры имеются в любой версии операционной системы. Нужно сходным образом вызвать окно «Выполнить», но прописать уже другую команду — «control userpasswords2». Чтобы управлять учётными записями пользователей надо нажать кнопку «Дополнительно», находящуюся на поле «Дополнительное управление пользователями». Появится окно, где слева нужно открыть раздел «Пользователи». В нём, уже с правой стороны, необходимо дважды нажать на строку «Администратор». Во вновь открывшемся окне следует убрать галочку в квадратике «Отключить учётную запись». По аналогии с прошлым методом требуется перегрузить компьютер или ноутбук. Окно для выдачи прав администратора представлено на рисунке 2.

|  |
| --- |
|  |
| Рисунок 2 – Выдача прав администратора через настройку учетных записей |

Если на данный момент попробовать выполнить какой-либо скрипт то увидим ошибку, представленную на рисунке 3.

|  |
| --- |
| Не удается загрузить файл ps1, так как выполнение скриптов запрещено для данной системы. |
| Рисунок 3 – Ошибка при запуске скрипта |

Чтобы изменить текущее значение политики запуска PowerShell скриптов, используется командлет Set-ExecutionPolicy. Разрешим запуск локальных скриптов с помощью команды «Set-ExecutionPolicy RemoteSigned» [6]. Подтвердите изменение политики запуска PS1 скриптов, нажав Y или A.

После выполнения этих действий можно приступать к работе со скриптами.

# 2 Разработка скрипта для получения списка установленных приложений

# 2.1 Постановка задачи курсового проекта

Задачей курсового проекта является получение списка установленных приложений. Для выполнения этой задачи необходим скрипт, который будет обращаться к ветке реестра HKLM, находить в нем данные об установленных приложениях и записывать эти данные в текстовый файл [7]. Создание текстового файла является наиболее универсальным способом получения списка установленных приложений, так как позволяет использовать эти данные для чтения другими программами или приложениями с последующим их редактированием [8]. Это является несомненным плюсом по сравнению с обычным выводом на экран пользователю. Также плюсом записи данных в файл является то, что данные сохраняются и в случае закрытия окна и последующего обновления списка установленных приложений, до повторного запуска скрипта остаются старые данные, что может быть полезно, если необходимо сравнивать список установленных приложений в разные временные промежутки.

# 2.2 Обоснование средств разработки

Для написания, отладки и проверки работоспособности скрипта будет использоваться среда сценариев PowerShell ISE [9]. Эта программа позволяет печатать скрипт и выполнять его в одном окне, что значительно упрощает отладку и увеличивает скорость работы со скриптом [14]. Однако использовать ее не обязательно, можно создать обычный текстовый файл, написать в нем скрипт и сохранить файл в формате скрипта. Однако при таком способе усложняется отладка и увеличивается общее время написания скрипта, поэтому в данной курсовой работе будет использоваться PowerShell ISE [15].

# 2.3 Описание работы системы

При открытии программы - скрипта в Powershell ISE, выводится окно, представленное на рисунке 4.

|  |
| --- |
|  |
| Рисунок 4 – Окно PowerShell ISE со скриптом |

Создание скрипта начинается с загрузки двух классов .NET Framework: System.Drawing и System.Windows.Forms. Затем запускается новый экземпляр класса платформа .NET Framework System.Windows.Forms.Form. Это предоставляет пустую форму или окно, в которое можно приступить к добавлению элементов управления [10]. После создания экземпляра класса "window\_form" назначаются значения для трех свойств этого класса:

* «.Text» - это будет заголовком окна;
* «.Width» и «.Height» - это размер формы в пикселях;
* «.AutoSize» - это динамическое изменение размера окна.

Далее создается переменная all\_programs и в нее записываются значения по пути «HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\». По этому пути в ветке «HKEY\_LOCAL\_MACHINE» хранятся данные об установленных приложениях [11]. Далее во второй строчке скрипта данные из переменной «all\_programs» записываются в файл «output.txt» по пути «C:\\output.txt» с помощью команды «Out-File». Out-File – является основным способом сохранения текстовых данных на жесткий диск. Он является более универсальным по сравнению с другими командами (командлетами).

Далее в строках 12 – 18 создается текстовое поле [12]. В 13 строке включается отображение или ввод нескольких строк текста. В 14 строке настраивается размер окна. В 15 строке «.Location» задает координаты левого верхнего угла элемента управления относительно левого верхнего угла его контейнера. В 16 строке создается ползунок для перемещения вверх-вниз. В 17 строке настраивается шрифт окна. В 18 строке в пустую форму добавляется текстовое поле. В 19 – 21 строке форма выводится на экран.

После запуска скрипта в корневом каталоге диска C: создается текстовый файл, содержание которого представлено на рисунке 5 и выводится окно, изображенное на рисунке 6.

|  |
| --- |
|  |
| Рисунок 5 – Список установленных приложений, записанный в текстовый файл |

Файл создается при запуске скрипта и перезаписывается, если он уже существует. На данном рисунке показан список установленных приложений на практически чистой системе виртуальной машины. На других системах данный список будет в разы больше.

|  |
| --- |
|  |
| Рисунок 6 – Окно вывода списка установленных приложений |

На данном рисунке можно увидеть окно с заданным в скрипте названием, а также текстовое поле выделенное синей рамкой. В правом нижнем углу окна можно заметить знак для изменения размера окна. В текстовом поле можно увидеть данные полностью соответствующие данным записанным в файле.

# ЗАКЛЮЧЕНИЕ

При выполнении данного курсового проекта был создан скрипт для получения списка всех установленных на компьютере приложений, с помощью PowerShell Integrated Scripting Environment. Получилось ускорить и облегчить процесс администрирования операционной системы. Также изучены:

* среда сценариев для разработки скриптов Powershell ISE;
* ветки реестров, работа подраздела реестра.

Благодаря данной работе улучшено знания об операционной системе Windows 10.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Реестр Windows – описание, настройка и полезные ветки реестра [Электронный ресурс] – URL: <https://info-comp.ru/vseowindowsst/44-sisreestr.html> (дата обращения 01.06.2023)
2. Типы данных Windows // Обучающий сайт Microsoft Learn – URL: <https://learn.microsoft.com/ru-ru/windows/win32/winprog/windows-data-types> (дата обращения 01.06.2023)
3. Реестр Windows // Электронный ресурс – URL: <https://ru.wikipedia.org/wiki/Реестр_Windows> (дата обращения 01.06.2023)
4. About\_Scripts [Электронный ресурс] – URL: <https://learn.microsoft.com/ru-ru/powershell/module/Microsoft.powershell.core/about/about_scripts?view=powershell-7.2&viewFallbackFrom=powershell-7.1> (дата обращения 01.06.2023)
5. Как получить права администратора в Windows // Электронный ресурс – URL: <https://sddhelp.ru/articles/kak-poluchit-prava-administratora-v-windows/> (дата обращения 01.06.2023)
6. Настройка политики запуска скриптов (Execution Policy) PowerShell // Электронный ресурс – URL: <https://winitpro.ru/index.php/2020/06/03/powershell-execution-policy-zapusk-scriptov/> (дата обращения 01.06.2023)
7. Работа с записями реестра // Электронный ресурс – URL: <https://learn.microsoft.com/ru-ru/powershell/scripting/samples/working-with-registry-entries?view=powershell-7.2> (дата обращения 01.06.2023)
8. Out-File // Электронный ресурс – URL: <https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/out-file?view=powershell-7.3> (дата обращения 01.06.2023)
9. PowerShell\_ise // Электронный ресурс – URL: <https://learn.microsoft.com/ru-ru/windows-server/administration/windows-commands/powershell_ise> (дата обращения 01.06.2023)
10. GUI на Powershell // Электронный ресурс – URL: <https://coolcode.ru/gui-na-powershell/> (дата обращения 01.06.2023)
11. Управление в Powershell реестром // Электронный ресурс – URL: <https://fixmypc.ru/post/rabota-v-powershell-s-reestrom/> (дата обращения 01.06.2023)
12. Создание настраиваемого поля ввода // Электронный ресурс – URL: <https://learn.microsoft.com/ru-ru/powershell/scripting/samples/creating-a-custom-input-box?view=powershell-7.3> (дата обращения 01.06.2023)
13. Внутреннее устройство Microsoft Windows. 6-е изд. / Руссинович М., Соломон Д., Алекс И. – Издательский дом "Питер", 2017. – 944 с.
14. Введение в Windows PowerShell./ Попов А. В. – БХВ-Петербург, 2012. – 455 с.
15. Введение в PowerShell // Электронный ресурс – URL: <http://get-powershell.ru/2015/vvedenie-v-powershell/> (дата обращения 01.06.2023)

# ПРИЛОЖЕНИЕ А

ЛИСТИНГ ПРОГРАММЫ:

Add-Type -assembly System.Windows.Forms

Add-Type -AssemblyName System.Drawing

$window\_form = New-Object System.Windows.Forms.Form

$window\_form.Text ='Список установленных приложений'

$window\_form.Width = 1080

$window\_form.Height = 300

$window\_form.AutoSize = $true

$all\_programs = Get-ChildItem -Path 'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\'

$all\_programs | Out-File -FilePath C:\output.txt

$TextBox = New-Object System.Windows.Forms.TextBox

$TextBox.Multiline = $true

$TextBox.Size = New-Object System.Drawing.Size(1290, 200)

$TextBox.Location = New-Object System.Drawing.Point(20, 20)

$TextBox.ScrollBars = "Vertical"

$TextBox.Font = "Lucida console, 9 pt"

$window\_form.Controls.Add($TextBox)

$TextBox.Text = $all\_programs | Format-Table | Out-String

$window\_form.ShowDialog()