Операционные системы

Отчет по лабораторной работе № 4

Выполнил: студент группы 4311

Наумов Д.О.

Цель работы

1. Изучение объектов WSH.
2. Изучение приемов программирования с использованием объектов WSH.
3. Получение навыков разработки сценариев.

Примеры

Пример 1

**Листинг программы:**

// fsoDirs.js - получение системных каталогов

var fso=WScript.CreateObject("Scripting.FileSystemObject");

var s="Специальные каталоги:";

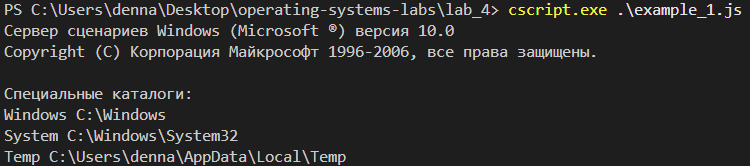
s+="\nWindows "+fso.GetSpecialFolder(0);

s+="\nSystem "+fso.GetSpecialFolder(1);

s+="\nTemp "+fso.GetSpecialFolder(2);

WScript.Echo(s);

**Пример работы программы:**

****

Пример 2

**Листинг программы:**

// fsoEnumDrv.js – вывод списка подключенных дисков

var fso=WScript.CreateObject("Scripting.FileSystemObject");

var e=new Enumerator(fso.Drives);

var s="";

var n=""

for(;!e.atEnd();e.moveNext())

{

var x=e.item();

s=s+x.DriveLetter;

s+=" - ";

switch(x.DriveType)

{

case 1: n="Отключаемый";break;

case 2: n="Жёсткий";break;

case 3: n="Сетевой";break;

case 4: n="CD-ROM";break;

case 5: n="RAM диск";break;

default:

n="Unknown";

}

s+=n+", ";

if(x.DriveType==3)

n=x.ShareName;

else if (x.IsReady)

n=x.VolumeName;

else

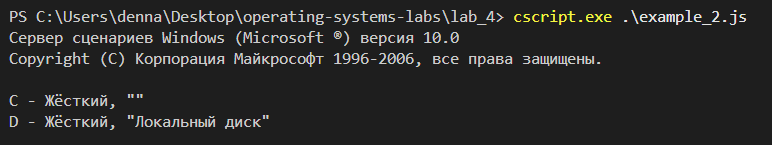
n="[Drive not ready]";

s+="\""+n+"\"\n";

}

WScript.Echo(s);

**Пример работы программы:**

****

Пример 3

**Листинг программы:**

// fsoCreateText.js - создание файла с деревом каталогов

var of;

function ProcessDirectory(dir,prefix)

{

var foldPrefix=prefix+"-";

prefix+="|";

var newPrefix=prefix+" ";

var fc = new Enumerator(dir.SubFolders);

while(!fc.atEnd())

{

of.WriteLine(foldPrefix+fc.item().Name);

ProcessDirectory(fc.item(),newPrefix);

fc.moveNext();

}

fc = new Enumerator(dir.Files);

while(!fc.atEnd())

{

of.WriteLine(prefix+fc.item().Name);

fc.moveNext();

}

}

if(WScript.Arguments.Count()==2)

{

var fso=WScript.CreateObject("Scripting.FileSystemObject");

var of=fso.CreateTextFile(WScript.Arguments(0),true);

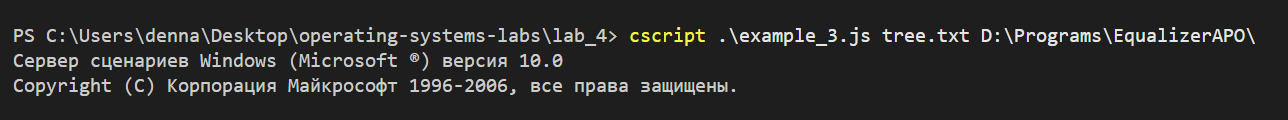
ProcessDirectory(fso.GetFolder(WScript.Arguments(1)),"")

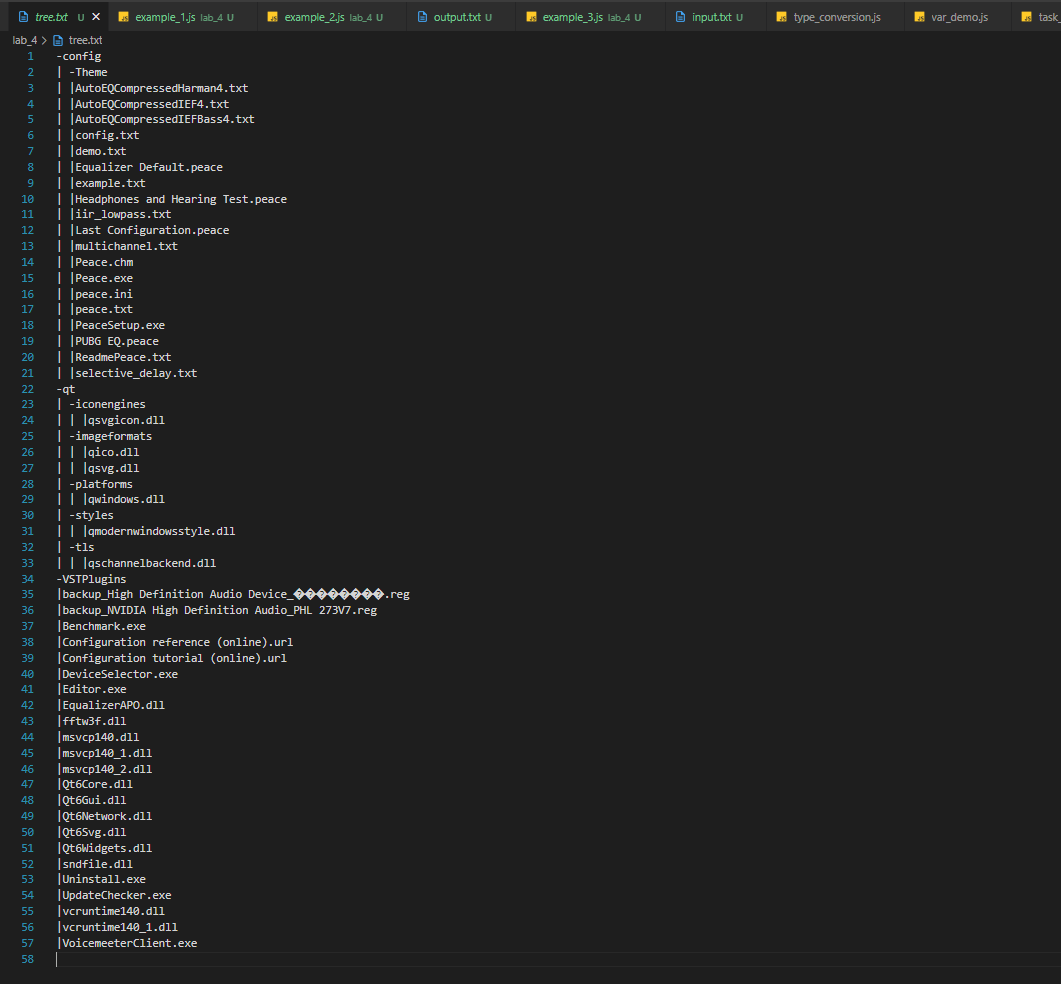
of.Close()

}else

WScript.Echo("Использование: cscript //nologo " + WScript.ScriptFullName + " tree.txt c:\\windows\\temp");

**Пример работы программы:**

****

****

Пример 4

**Листинг программы:**

// fsoCreateText.js - создание файла с таблицей умножения

if(WScript.Arguments.Count()==1)

{

var fso=WScript.CreateObject("Scripting.FileSystemObject");

var tf=fso.CreateTextFile(WScript.Arguments(0),true);

tf.WriteLine("Таблица Пифагора:");

var i;

tf.Write(" |");

for(i=1;i<10;i++)

{

tf.Write(" "+i+"|");

}

tf.WriteLine("");

tf.WriteLine("-+--+--+--+--+--+--+--+--+--+");

var j;

for(i=1;i<10;i++)

{

tf.Write(i);

for(j=1;j<10;j++)

{

tf.Write("|");

if(i\*j<10)

77

tf.Write(" ");

tf.Write(i\*j);

}

tf.WriteLine("|");

}

tf.WriteLine("-+--+--+--+--+--+--+--+--+--+");

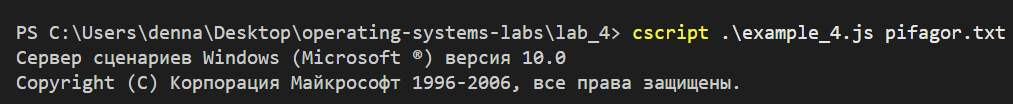
tf.Close();

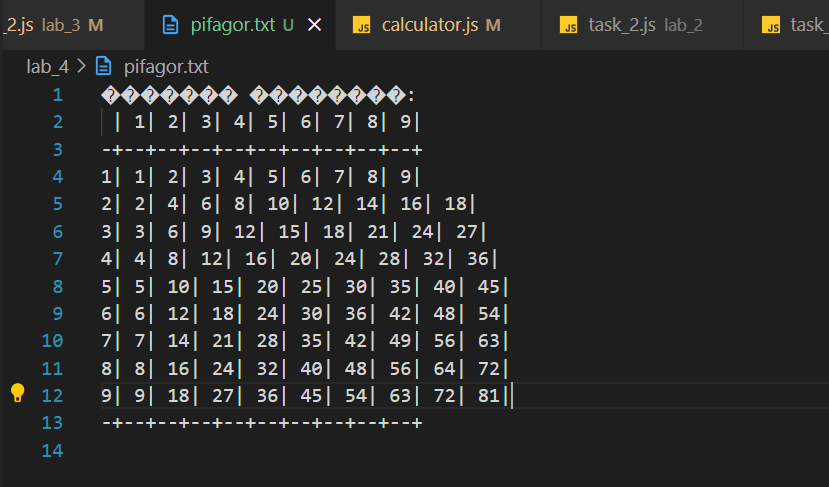
}else

WScript.Echo("Использование: cscript //nologo "+WScript.ScriptFullName+

" pifagor.txt");

**Пример работы программы:**

****

****

Задания

Задание 1

Разработать процедуру, выполняющую заданную в индивидуальном задании последовательность операций над файлами (каталогами) с помощью различных объектов работы с файловой системой. Организовать ввод всех исходных данных в процедуру с помощью параметров командной строки при вызове процедуры. Предусмотреть возможность перенаправления результатов выполнения процедуры в файл.

3. Переименование каталогов:

3.1. Создать исходный каталог;

3.2. С помощью Блокнота создать исходный файл;

3.3. Переименовать исходный каталог в результирующий каталог;

3.4. Изменить атрибуты файла в результирующем каталоге;

3.5. Вывести оглавление исходного и результирующего каталогов;

3.6. Вывести содержимое файла результирующего каталога;

**Листинг программы:**

var shell = new ActiveXObject("WScript.Shell");

var fso = new ActiveXObject("Scripting.FileSystemObject");

var outputFile = null;

var outputBuffer = [];

function echo(message) {

    outputBuffer.push(message);

    if (!outputFile) {

        WScript.Echo(message);

    }

}

function showUsage() {

    var message = "Usage: cscript " + WScript.ScriptName + " <source\_dir> <target\_dir> <filename> <file\_content> [output\_file]\n" +

                  "\n" +

                  "Parameters:\n" +

                  "  source\_dir   - Source directory name\n" +

                  "  target\_dir   - Target directory name\n" +

                  "  filename     - Name of the file to create\n" +

                  "  file\_content - Content for the file\n" +

                  "  output\_file  - Optional path to file for output redirection (relative or absolute)\n" +

                  "\n" +

                  "Examples:\n" +

                  "  cscript rename\_procedure\_mixed.js old\_folder new\_folder myfile.txt \"Hello World\"\n" +

                  "  cscript rename\_procedure\_mixed.js temp result data.txt \"Sample content\" output.log";

    echo(message);

    return message;

}

function executeCommand(command) {

    try {

        // Устанавливаем английскую локаль для команды dir

        var exec = shell.Exec("cmd.exe /c chcp 437 >nul & " + command);

        var output = "";

        while (!exec.StdOut.AtEndOfStream) {

            output += exec.StdOut.ReadLine() + "\n";

        }

        return output;

    } catch (e) {

        return "Ошибка: " + e.message;

    }

}

function getDirectoryContents(path) {

    try {

        var folder = fso.GetFolder(path);

        var contents = "";

        contents += "Файлы:\n";

        var files = new Enumerator(folder.Files);

        for (; !files.atEnd(); files.moveNext()) {

            var file = files.item();

            contents += "  " + file.Name + " (" + file.Size + " байт, атрибуты: " + getAttributesText(file.Attributes) + ")\n";

        }

        contents += "Подкаталоги:\n";

        var subfolders = new Enumerator(folder.SubFolders);

        var hasSubfolders = false;

        for (; !subfolders.atEnd(); subfolders.moveNext()) {

            var subfolder = subfolders.item();

            contents += "  " + subfolder.Name + "\n";

            hasSubfolders = true;

        }

        if (!hasSubfolders) {

            contents += "  (нет)\n";

        }

        return contents;

    } catch (e) {

        return "Ошибка чтения каталога: " + e.message;

    }

}

function getAttributesText(attributes) {

    var attrs = [];

    if (attributes & 1) attrs.push("Только чтение");

    if (attributes & 2) attrs.push("Скрытый");

    if (attributes & 4) attrs.push("Системный");

    if (attributes & 32) attrs.push("Архивный");

    return attrs.length > 0 ? attrs.join(",") : "Обычный";

}

function waitForProcessCompletion(processName, timeout) {

    var start = new Date().getTime();

    while (new Date().getTime() - start < timeout) {

        var exec = shell.Exec("cmd.exe /c tasklist /FI \"IMAGENAME eq " + processName + "\" /FO CSV");

        var output = "";

        while (!exec.StdOut.AtEndOfStream) {

            output += exec.StdOut.ReadLine() + "\n";

        }

        if (output.indexOf(processName) === -1) {

            return true; // Процесс завершен

        }

        WScript.Sleep(100);

    }

    return false; // Таймаут

}

function forceDeleteFolder(folderPath) {

    try {

        if (!fso.FolderExists(folderPath)) {

            return true;

        }

        // Сначала пытаемся удалить стандартным способом

        fso.DeleteFolder(folderPath);

        return true;

    } catch (e) {

        echo("Стандартное удаление не удалось, пробуем через команду...");

        // Пробуем через команду rd (remove directory) с принудительным удалением

        try {

            var exec = shell.Exec("cmd.exe /c rd /s /q \"" + folderPath + "\"");

            while (!exec.StdOut.AtEndOfStream) {

                exec.StdOut.ReadAll();

            }

            while (!exec.StdErr.AtEndOfStream) {

                exec.StdErr.ReadAll();

            }

            // Проверяем, удалилась ли папка

            WScript.Sleep(1000);

            return !fso.FolderExists(folderPath);

        } catch (e2) {

            echo("Ошибка принудительного удаления: " + e2.message);

            return false;

        }

    }

}

function writeOutputToFile() {

    if (outputFile) {

        try {

            var file = fso.CreateTextFile(outputFile, true, true);

            file.Write(outputBuffer.join("\n"));

            file.Close();

            WScript.Echo("Вывод записан в файл: " + outputFile);

        } catch (e) {

            WScript.Echo("✗ Ошибка записи в файл '" + outputFile + "': " + e.message);

        }

    }

}

function main() {

    var args = WScript.Arguments;

    if (args.length < 4) {

        showUsage();

        writeOutputToFile();

        WScript.Quit(1);

    }

    var sourceDir = args(0);

    var targetDir = args(1);

    var fileName = args(2);

    var fileContent = args(3);

    outputFile = args.length > 4 ? args(4) : null;

    echo("=== Процедура переименования каталогов ===");

    echo("Исходный каталог: " + sourceDir);

    echo("Целевой каталог: " + targetDir);

    echo("Имя файла: " + fileName);

    echo("Содержимое файла: " + fileContent);

    echo("===========================================");

    // 3.1. Создать исходный каталог

    echo("\n3.1. Создание исходного каталога: " + sourceDir);

    try {

        if (fso.FolderExists(sourceDir)) {

            echo("Каталог уже существует, удаление...");

            if (!forceDeleteFolder(sourceDir)) {

                throw new Error("Не удалось удалить существующий каталог");

            }

        }

        fso.CreateFolder(sourceDir);

        echo("✓ Исходный каталог успешно создан");

    } catch (e) {

        echo("✗ Ошибка создания исходного каталога: " + e.message);

        writeOutputToFile();

        WScript.Quit(1);

    }

    // 3.2. С помощью Блокнота создать исходный файл

    echo("\n3.2. Создание файла в Блокноте: " + fileName);

    try {

        var filePath = fso.BuildPath(sourceDir, fileName);

        var file = fso.CreateTextFile(filePath, true);

        file.Write(fileContent);

        file.Close();

        echo("✓ Файл успешно создан: " + filePath);

        // Открываем в Блокноте для подтверждения

        echo("Открытие файла в Блокноте для проверки...");

        shell.Run("notepad.exe \"" + filePath + "\"", 1, true);

        echo("✓ Блокнот закрыт, файл сохранен");

        // Ждем завершения всех процессов Блокнота

        echo("Ожидание завершения процессов Блокнота...");

        if (!waitForProcessCompletion("notepad.exe", 5000)) {

            echo("Предупреждение: Процессы Блокнота могут быть еще активны");

        }

    } catch (e) {

        echo("✗ Ошибка создания файла: " + e.message);

        writeOutputToFile();

        WScript.Quit(1);

    }

    // Вывод оглавления исходного каталога после создания файла

    echo("\nОглавление исходного каталога после создания файла:");

    echo("Исходный каталог '" + sourceDir + "':");

    if (fso.FolderExists(sourceDir)) {

        echo(getDirectoryContents(sourceDir));

    } else {

        echo("  НЕ СУЩЕСТВУЕТ - ОШИБКА!");

    }

    echo("Содержимое каталога через команду 'dir':");

    var dirOutput = executeCommand("dir \"" + sourceDir + "\" /A");

    echo(dirOutput);

    // 3.3. Переименовать исходный каталог в результирующий каталог

    echo("\n3.3. Переименование каталога из '" + sourceDir + "' в '" + targetDir + "'");

    try {

        if (fso.FolderExists(targetDir)) {

            echo("Целевой каталог существует, удаление...");

            if (!forceDeleteFolder(targetDir)) {

                throw new Error("Не удалось удалить существующий целевой каталог");

            }

        }

        // Добавляем задержку перед переименованием

        echo("Ожидание перед переименованием...");

        WScript.Sleep(2000);

        fso.MoveFolder(sourceDir, targetDir);

        echo("✓ Каталог успешно переименован");

    } catch (e) {

        echo("✗ Ошибка переименования каталога: " + e.message);

        echo("Попытка альтернативного метода переименования...");

        // Альтернативный метод: копирование и удаление

        try {

            if (!fso.FolderExists(sourceDir)) {

                throw new Error("Исходный каталог не существует");

            }

            // Копируем содержимое

            fso.CopyFolder(sourceDir + "\\\*", targetDir);

            echo("Содержимое скопировано в целевой каталог");

            // Удаляем исходный каталог

            if (!forceDeleteFolder(sourceDir)) {

                echo("Предупреждение: Не удалось удалить исходный каталог после копирования");

            } else {

                echo("Исходный каталог удален");

            }

            echo("✓ Каталог успешно перемещен альтернативным методом");

        } catch (e2) {

            echo("✗ Альтернативный метод также не удался: " + e2.message);

            writeOutputToFile();

            WScript.Quit(1);

        }

    }

    // 3.4. Изменить атрибуты файла в результирующем каталоге

    echo("\n3.4. Изменение атрибутов файла в целевом каталоге");

    try {

        var targetFilePath = fso.BuildPath(targetDir, fileName);

        var targetFile = fso.GetFile(targetFilePath);

        echo("Текущие атрибуты: " + getAttributesText(targetFile.Attributes));

        // Устанавливаем атрибуты: Только чтение + Скрытый

        targetFile.Attributes = 1 + 2; // ReadOnly + Hidden

        echo("Установлены атрибуты: Только чтение + Скрытый");

        echo("✓ Атрибуты файла успешно изменены");

        // Показываем атрибуты через команду attrib (на английском)

        var attribOutput = executeCommand("attrib \"" + targetFilePath + "\"");

        echo("Атрибуты (через команду attrib):");

        echo(attribOutput);

    } catch (e) {

        echo("✗ Ошибка изменения атрибутов файла: " + e.message);

    }

    // 3.5. Вывести оглавление исходного и результирующего каталогов

    echo("\n3.5. Оглавление исходного и результирующего каталогов:");

    // Вывод исходного каталога

    echo("Исходный каталог '" + sourceDir + "':");

    if (fso.FolderExists(sourceDir)) {

        echo(getDirectoryContents(sourceDir));

    } else {

        echo("  НЕ СУЩЕСТВУЕТ - переименован в целевой каталог");

    }

    echo("Содержимое каталога через команду 'dir':");

    var dirOutputSource = executeCommand("dir \"" + sourceDir + "\" /A");

    echo(dirOutputSource);

    echo("Целевой каталог '" + targetDir + "':");

    if (fso.FolderExists(targetDir)) {

        echo(getDirectoryContents(targetDir));

    } else {

        echo("  НЕ СУЩЕСТВУЕТ - ОШИБКА!");

    }

    // Вывод через команду dir с английской локалью

    echo("Содержимое каталога через команду 'dir':");

    var dirOutputTarget = executeCommand("dir \"" + targetDir + "\" /A");

    echo(dirOutputTarget);

    // 3.6. Вывести содержимое файла результирующего каталога

    echo("\n3.6. Содержимое файла из целевого каталога:");

    try {

        var finalFilePath = fso.BuildPath(targetDir, fileName);

        if (fso.FileExists(finalFilePath)) {

            // Сначала снимаем атрибут "Только чтение" чтобы можно было прочитать файл

            var finalFile = fso.GetFile(finalFilePath);

            finalFile.Attributes = 0; // Normal

            var file = fso.OpenTextFile(finalFilePath, 1); // 1 = ForReading

            var content = file.ReadAll();

            file.Close();

            echo("Файл: " + finalFilePath);

            echo("Содержимое:");

            echo("------------------------");

            echo(content);

            echo("------------------------");

            echo("✓ Содержимое файла успешно отображено");

            // Восстанавливаем атрибуты

            finalFile.Attributes = 1 + 2; // ReadOnly + Hidden

        } else {

            echo("✗ Файл не найден: " + finalFilePath);

        }

    } catch (e) {

        echo("✗ Ошибка чтения содержимого файла: " + e.message);

    }

    echo("\n=== Процедура успешно завершена ===");

    echo("Итог:");

    echo("- Исходный каталог создан и заполнен");

    echo("- Каталог переименован в: " + targetDir);

    echo("- Атрибуты файла изменены");

    echo("- Содержимое каталога проверено");

    echo("- Содержимое файла отображено");

    writeOutputToFile();

}

// Основной код выполнения

try {

    if (WScript.Arguments.length === 0 ||

        WScript.Arguments(0) === "/?" ||

        WScript.Arguments(0) === "-help" ||

        WScript.Arguments(0) === "--help") {

        showUsage();

        if (WScript.Arguments.length > 1) {

            outputFile = WScript.Arguments(1);

        }

        writeOutputToFile();

        WScript.Quit(0);

    } else {

        main();

    }

} catch (e) {

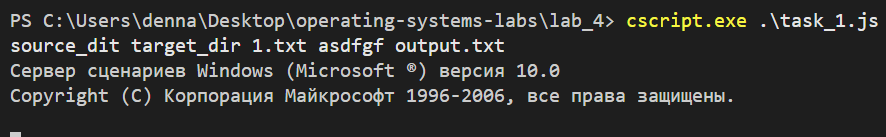
    echo("Критическая ошибка: " + e.message);

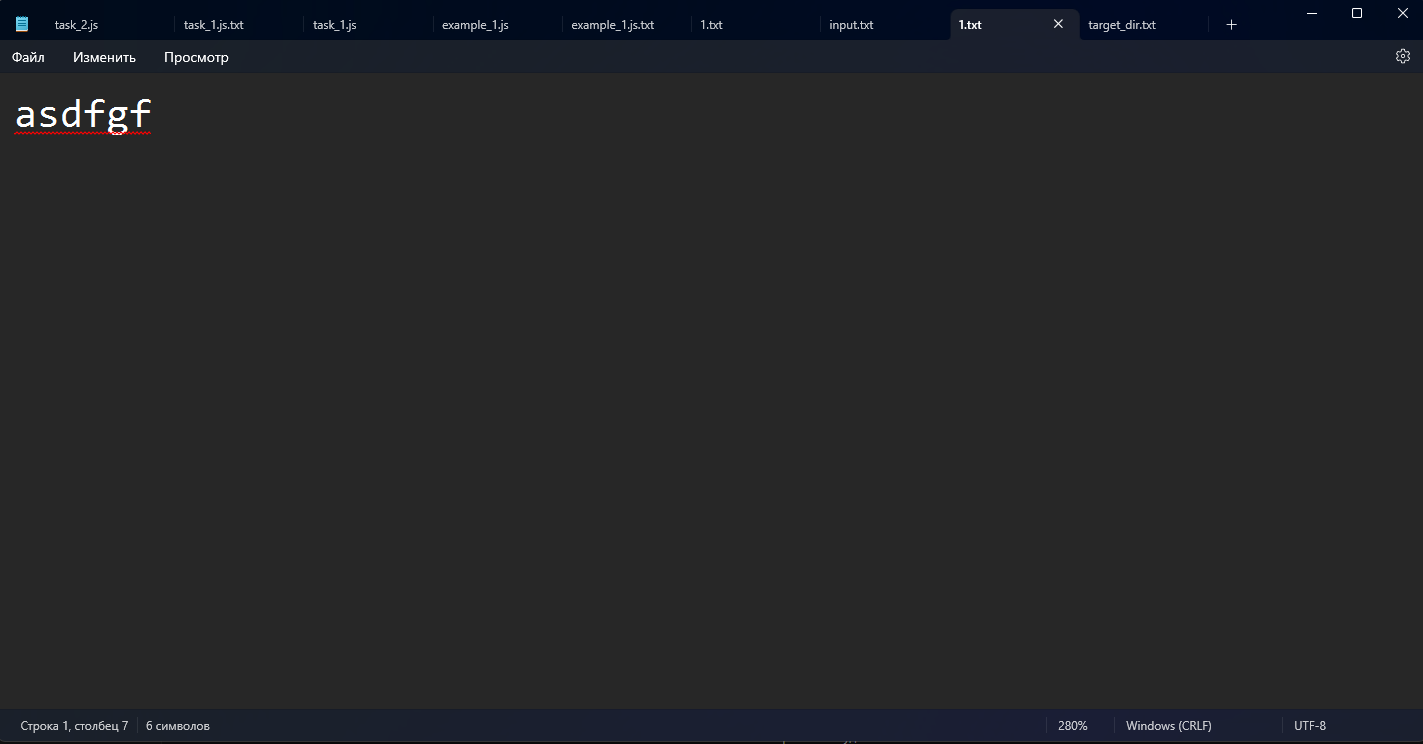
    writeOutputToFile();

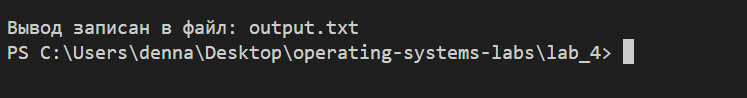
    WScript.Quit(1);

}

**Пример работы программы:**







Содержимое output.txt:

=== Процедура переименования каталогов ===

Исходный каталог: source\_dit

Целевой каталог: target\_dir

Имя файла: 1.txt

Содержимое файла: asdfgf

===========================================

3.1. Создание исходного каталога: source\_dit

✓ Исходный каталог успешно создан

3.2. Создание файла в Блокноте: 1.txt

✓ Файл успешно создан: source\_dit\1.txt

Открытие файла в Блокноте для проверки...

✓ Блокнот закрыт, файл сохранен

Ожидание завершения процессов Блокнота...

Оглавление исходного каталога после создания файла:

Исходный каталог 'source\_dit':

Файлы:

1.txt (6 байт, атрибуты: Архивный)

Подкаталоги:

(нет)

Содержимое каталога через команду 'dir':

’®¬ ў гбва®©бвўҐ C ­Ґ Ё¬ҐҐв ¬ҐвЄЁ.

‘ҐаЁ©­л© ­®¬Ґа в®¬ : 00E8-9417

‘®¤Ґа¦Ё¬®Ґ Ї ЇЄЁ C:\Users\denna\Desktop\operating-systems-labs\lab\_4\source\_dit

10.10.2025 02:59 <DIR> .

10.10.2025 02:59 <DIR> ..

10.10.2025 02:59 6 1.txt

1 д ©«®ў 6 Ў ©в

2 Ї Ї®Є 330я009я726я976 Ў ©в бў®Ў®¤­®

3.3. Переименование каталога из 'source\_dit' в 'target\_dir'

Целевой каталог существует, удаление...

Стандартное удаление не удалось, пробуем через команду...

Ожидание перед переименованием...

✓ Каталог успешно переименован

3.4. Изменение атрибутов файла в целевом каталоге

Текущие атрибуты: Архивный

Установлены атрибуты: Только чтение + Скрытый

✓ Атрибуты файла успешно изменены

Атрибуты (через команду attrib):

HR C:\Users\denna\Desktop\operating-systems-labs\lab\_4\target\_dir\1.txt

3.5. Оглавление исходного и результирующего каталогов:

Исходный каталог 'source\_dit':

НЕ СУЩЕСТВУЕТ - переименован в целевой каталог

Содержимое каталога через команду 'dir':

Volume in drive C has no label.

Volume Serial Number is 00E8-9417

Directory of C:\Users\denna\Desktop\operating-systems-labs\lab\_4

Целевой каталог 'target\_dir':

Файлы:

1.txt (6 байт, атрибуты: Только чтение,Скрытый)

Подкаталоги:

(нет)

Содержимое каталога через команду 'dir':

Volume in drive C has no label.

Volume Serial Number is 00E8-9417

Directory of C:\Users\denna\Desktop\operating-systems-labs\lab\_4\target\_dir

10.10.2025 02:59 <DIR> .

10.10.2025 03:01 <DIR> ..

10.10.2025 02:59 6 1.txt

1 File(s) 6 bytes

2 Dir(s) 330я009я726я976 bytes free

3.6. Содержимое файла из целевого каталога:

Файл: target\_dir\1.txt

Содержимое:

------------------------

asdfgf

------------------------

✓ Содержимое файла успешно отображено

=== Процедура успешно завершена ===

Итог:

- Исходный каталог создан и заполнен

- Каталог переименован в: target\_dir

- Атрибуты файла изменены

- Содержимое каталога проверено

- Содержимое файла отображено

Задание 2

Разработать процедуру запуска Блокнота для создания новой или редактирования существующей процедуры с возможностью ее последующего запуска в оконном режиме или режиме командной строки. Организовать ввод пути и имени редактируемой процедуры и режим ее запуска с помощью параметров командной строки.

**Листинг программы:**

var args = WScript.Arguments;

if (args.length < 5 || args.length > 6) {

    WScript.Echo("Использование: cscript " + WScript.Name + " <inputFile> <maxLength> <maxLineLength> <maxWordLength> <targetLetter> [outputFile]");

    WScript.Quit(1);

}

// Аргументы

var inputFile = args(0);                 // полный или относительный путь к текстовому файлу

var maxLength = parseInt(args(1));

var maxLineLength = parseInt(args(2));

var maxWordLength = parseInt(args(3));

var targetLetter = args(4);

var outputFile = args.length === 6 ? args(5) : "";

// Создаём объекты

var shell = WScript.CreateObject("WScript.Shell");

var fso = new ActiveXObject("Scripting.FileSystemObject");

// Функция вывода с поддержкой файла

function writeOutput(msg) {

    if (outputFile !== "") {

        try {

            var outFile = fso.OpenTextFile(outputFile, 8, true, -1); // 8 = append, -1 = system default (Unicode)

            outFile.WriteLine(msg);

            outFile.Close();

        } catch (e) {

            WScript.Echo("Ошибка записи в файл: " + e.message);

        }

    } else {

        WScript.Echo(msg);

    }

}

// Функция вывода ошибки

function writeError(msg) {

    writeOutput("Ошибка: " + msg);

    WScript.Quit(1);

}

// 1. Создаём/редактируем файл через Блокнот

shell.Run('notepad.exe "' + inputFile + '"', 1, true); // ждём закрытия

// 2. Проверки числовых параметров

if (isNaN(maxLength) || maxLength <= 0 || maxLength % 1 !== 0) writeError("maxLength должно быть положительным целым числом.");

if (isNaN(maxLineLength) || maxLineLength <= 0 || maxLineLength % 1 !== 0) writeError("maxLineLength должно быть положительным целым числом.");

if (isNaN(maxWordLength) || maxWordLength <= 0 || maxWordLength % 1 !== 0) writeError("maxWordLength должно быть положительным целым числом.");

if (targetLetter.length !== 1) writeError("targetLetter должно быть одной буквой.");

// 3. Чтение файла с поддержкой Unicode через ADODB.Stream

if (!fso.FileExists(inputFile)) writeError("файл не найден после редактирования.");

var stream = WScript.CreateObject("ADODB.Stream");

stream.Type = 2; // текст

stream.Charset = "utf-8"; // "unicode" для UTF-16 LE

stream.Open();

stream.LoadFromFile(inputFile);

var text = stream.ReadText();

stream.Close();

// Убираем BOM и лишние пробельные символы в конце

text = text.replace(/^\uFEFF/, "").replace(/[\s\r\n]+$/g, "");

// Проверка конца текста

if (!checkTextEndsWithDot(text)) writeError("текст не заканчивается точкой.");

// Проверка длины текста

if (!checkTextLengthValidity(text, maxLength)) writeError("текст превышает NL символов.");

// 4. Обработка строк

var lines = text.split(/\r?\n/);

var foundWords = [];

var wordCount = 0;

for (var i = 0; i < lines.length; i++) {

    var line = lines[i].replace(/\r$/, ""); // убираем \r если есть

    if (!checkLineLength(line, maxLineLength)) writeError("строка " + (i + 1) + " превышает NS символов.");

    var words = findWordsEndingWithLetter(line, maxWordLength, targetLetter);

    for (var j = 0; j < words.length; j++) {

        foundWords.push(words[j]);

    }

    wordCount += words.length;

}

// Формируем вывод

var output = "Исходный текст:\n" + text + "\n\n";

output += "Найденные слова: " + foundWords.join(", ") + "\n";

output += "Количество слов: " + wordCount + "\n";

// Вывод результатов

writeOutput(output);

function checkTextLengthValidity(text, maxLength) {

    return text.length <= maxLength;

}

function checkTextEndsWithDot(text) {

    return text.charAt(text.length - 1) === ".";

}

function checkLineLength(line, maxLineLength) {

    return line.length <= maxLineLength;

}

function trimLine(line) {

    var start = 0;

    var end = line.length - 1;

    while (start <= end && line.charAt(start) === " ") start++;

    while (end >= start && line.charAt(end) === " ") end--;

    return line.substring(start, end + 1);

}

function splitIntoWords(trimmedLine) {

    return trimmedLine.split(" ");

}

function isValidWord(word, maxWordLength, targetLetter) {

    if (word.length === 0) return false;

    var wordToCheck = removePunctuationFromEnd(word).toLowerCase();

    targetLetter = targetLetter.toLowerCase();

    if (wordToCheck.length > maxWordLength) return false;

    if (wordToCheck.length === 0) return false;

    return wordToCheck.charAt(wordToCheck.length - 1) === targetLetter;

}

function getCleanWord(word) {

    return removePunctuationFromEnd(word);

}

function findWordsEndingWithLetter(line, maxWordLength, targetLetter) {

    var trimmedLine = trimLine(line);

    var words = splitIntoWords(trimmedLine);

    var foundWords = [];

    for (var j = 0; j < words.length; j++) {

        if (isValidWord(words[j], maxWordLength, targetLetter)) {

            var wordToAdd = getCleanWord(words[j]);

            foundWords.push(wordToAdd);

        }

    }

    return foundWords;

}

function isPunctuation(char) {

    return char === "." || char === "," || char === "!" || char === "?" || char === ";";

}

function removePunctuationFromEnd(word) {

    var cleanedWord = word;

    while (cleanedWord.length > 0 && isPunctuation(cleanedWord.charAt(cleanedWord.length - 1))) {

        cleanedWord = cleanedWord.substring(0, cleanedWord.length - 1);

    }

    return cleanedWord;

}

**Пример работы программы:**

