ФИЛИАЛ ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «МЭИ»

В Г. СМОЛЕНСКЕ

Кафедра вычислительной техники

Дисциплина: ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

АВТОМАТИЗИРОВАННЫХ СИСТЕМ

Лабораторная работа № 10.

СОЗДАНИЕ СКРИПТОВ КОМАНДНОГО ИНТЕРПРЕТАТОРА BASH

|  |  |  |
| --- | --- | --- |
|  | Студент: | Старостенков А.А. |
|  | Группа: | ВМ-22 (маг.) |
|  | Вариант: | 15 |
|  | Преподаватель: | Федулов Я.А. |

СМОЛЕНСК

2023

**Задание**:

1. Создайте файл исходного текста программы с помощью редактора nano или gedit
2. Разработайте алгоритм скрипта в соответствии с вариантом задания.
3. Разбейте алгоритм на несколько логических блоков.
4. Создайте исходный текст скрипта, реализующей разработанный алгоритм.
5. Разрешите доступ к файлу скрипта на исполнение командой chown u+x script.sh, где script.sh – название скрипта.
6. Устраните синтаксические ошибки, если такие имеются.
7. Разработайте набор тестов с учетом особенностей реализации алгоритма.
8. Проверьте правильность функционирования программы с использованием набора тестов.
9. Устраните семантические ошибки в программе, если такие имеются.

Вариант 15.

Написать программу подсчета количества слов для всех текстовых файлов \*.txt заданного каталога. Формат вызова программы: wordcount <путь>. Программа должна выводить список файлов в два столбца: имя файла, количество слов в файле.

**Выполнение**

Для удобства напишем bash-скрипт генерации и заполнения текстовых файлов в указанном каталоге. Так мы облегчим себе работу, ведь не придётся самостоятельно создавать текстовые файлы, заполнять их словами и т.д. Мы автоматизируем этот процесс.

Этот скрипт будет принимать два основных параметра: путь к каталогу и количество файлов для создания.

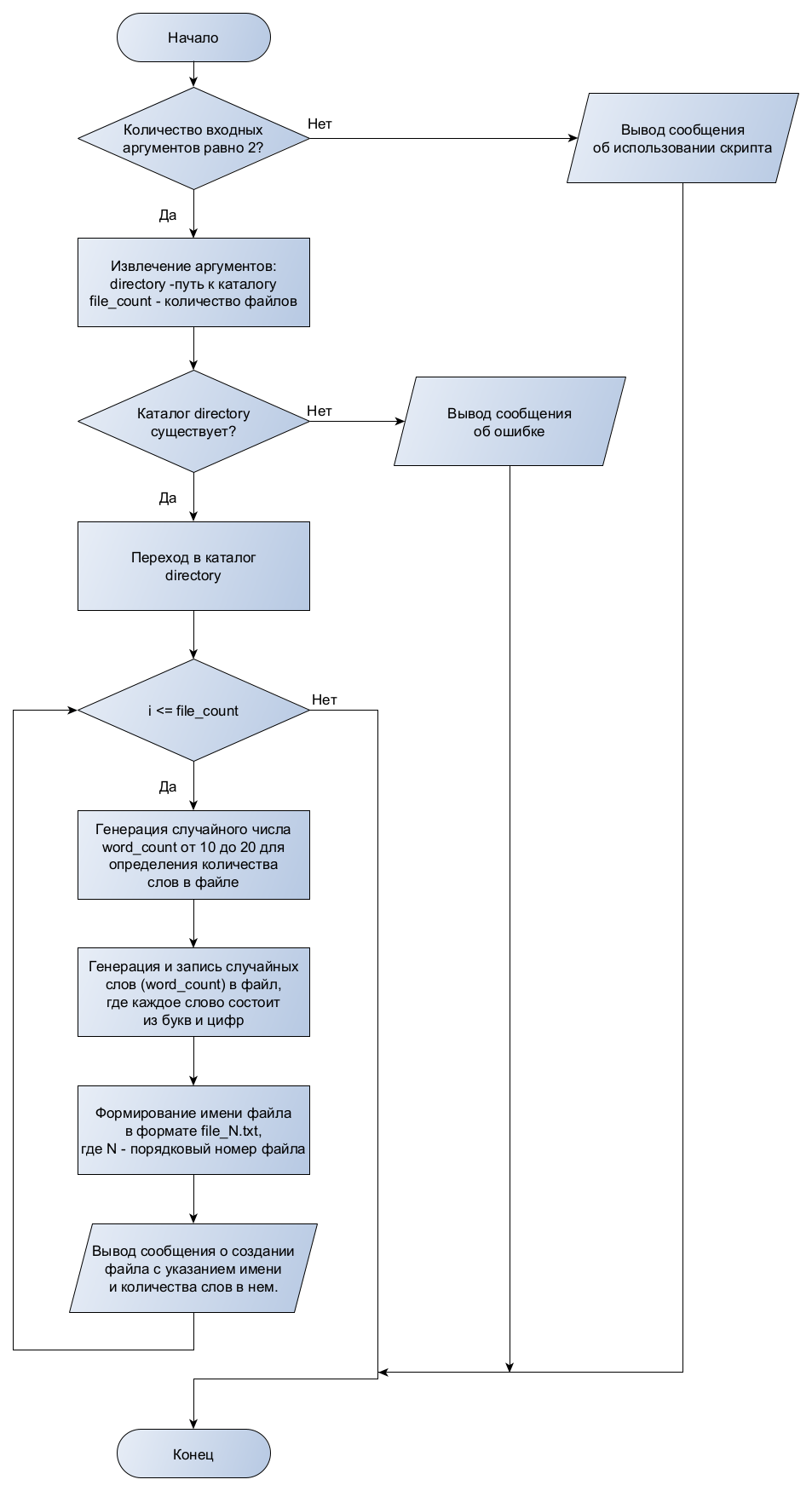


Рисунок 1 – схема алгоритма генерации текстовых файлов

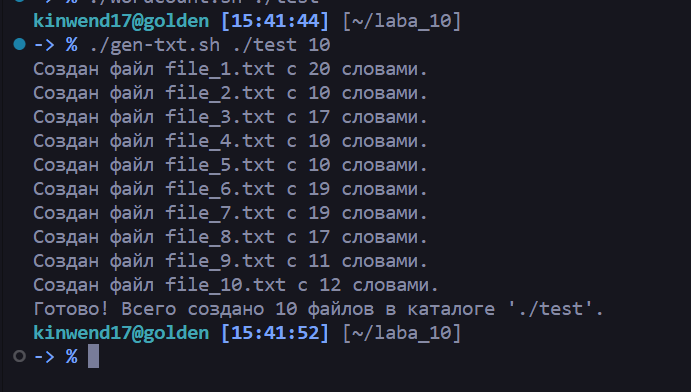


Рисунок 2 – Пример вывода bash-скрипта генерации файлов

Наша задача заключается в написании bash-скрипта, который выполняет подсчет количества слов во всех текстовых файлах (\*.txt) в указанном каталоге и выводит результат в формате "имя файла - количество слов". Вызов программы должен быть в формате wordcount <путь>.

Алгоритм разработанного скрипта представлен на рисунке 4.

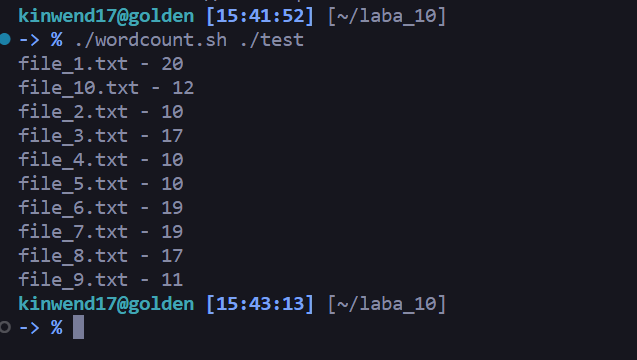


Рисунок 3 –Вывода bash-скрипта wordcount

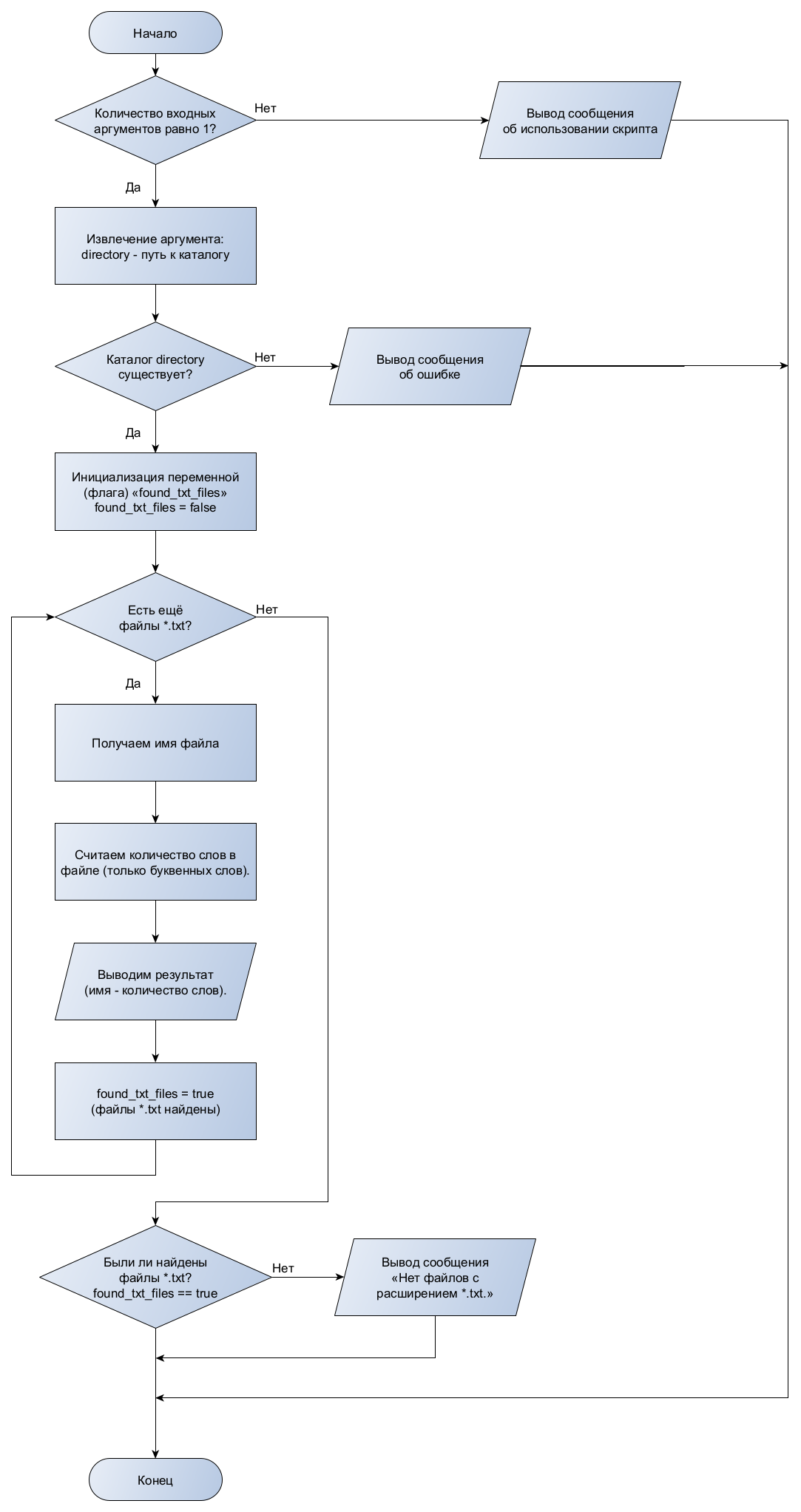


Рисунок 4 – Пример вывода bash-скрипта генерации файлов

**Контрольные вопросы:**

1. Для чего служит переменная $#?

Переменная $# – это переменная командной строки в языке shell, которая содержит количество аргументов (параметров), переданных скрипту или команде. Например, если запустить некий скрипт ./myscript.sh arg1 arg2 arg3, то $# будет равно 3, так как переданы три аргумента.

1. Для чего служит переменная $@?

Переменная $@ – это переменная, которая представляет все аргументы (параметры) командной строки, переданные скрипту или команде, в виде списка отдельных аргументов. Каждый аргумент отделен от другого пробелом. Например, если запустить некий скрипт ./myscript.sh arg1 arg2 arg3, то $@ будет представлять список: arg1 arg2 arg3.

1. Для чего служит переменная $0?

Переменная $0 – это переменная, которая представляет имя самого скрипта, который был запущен. Если запустить некий скрипт ./myscript.sh, то $0 будет равно «./myscript.sh».

1. Для чего служит переменная $1, $2, ... ?

Переменные $1, $2, и так далее – это переменные, которые представляют отдельные аргументы (параметры) командной строки, переданные скрипту или команде. $1 представляет первый аргумент, $2 - второй аргумент, и так далее. Например, если запустить скрипт ./myscript.sh arg1 arg2 arg3, то $1 будет равно «arg1», $2 будет равно «arg2», и $3 будет равно «arg3».

**Приложение А. Bash-скрипт для генерации и заполнении текстовых файлов в указанном каталоге.**

#!/bin/bash

# Проверяем, что передано два аргумента

if [ "$#" -ne 2 ]; then

echo "Использование: $0 <путь\_к\_каталогу> <количество\_файлов>"

exit 1

fi

# Извлекаем аргументы

directory="$1"

file\_count="$2"

# Проверяем, что каталог существует

if [ ! -d "$directory" ]; then

echo "Каталог '$directory' не существует."

exit 1

fi

# Переходим в указанный каталог

cd "$directory" || exit 1

# Генерируем и записываем файлы

for ((i = 1; i <= file\_count; i++)); do

# Генерируем случайное количество (от 10 до 20) слов

word\_count=$((RANDOM % 11 + 10))

# Генерируем случайные слова и записываем их в файл

words=""

for ((j = 1; j <= word\_count; j++)); do

word=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 5 | head -n 1)

words="$words $word"

done

# Создаем файл и записываем в него сгенерированные слова

filename="file\_$i.txt"

echo "$words" > "$filename"

echo "Создан файл $filename с $word\_count словами."

done

echo "Готово! Всего создано $file\_count файлов в каталоге '$directory'."

**Приложение Б. Bash-скрипт подсчета количества слов**

#!/bin/bash

# Проверка наличия аргумента (пути к каталогу)

if [ "$#" -ne 1 ]; then

echo "Формат вызова: wordcount <путь>"

exit 1

fi

# Путь к целевому каталогу

directory="$1"

# Проверка существования каталога

if [ ! -d "$directory" ]; then

echo "Указанный каталог не существует."

exit 1

fi

# Переменная, чтобы отслеживать наличие файлов \*.txt

found\_txt\_files=false

# Поиск текстовых файлов и подсчет слов

for file in "$directory"/\*.txt; do

if [ -f "$file" ]; then

# Имя файла

filename=$(basename "$file")

# Подсчет количества слов в файле (только буквенных слов)

word\_count=$(cat "$file" | tr -s '[:space:]' '\n' | grep -o -E '\w+' | wc -l)

# Вывод результата (имя файла и количество слов)

echo "$filename - $word\_count"

found\_txt\_files=true

fi

done

# Проверка, были ли найдены файлы \*.txt

if [ "$found\_txt\_files" = false ]; then

echo "В указанной директории нет файлов с расширением \*.txt."

fi