Отчет по работе с Bash-скриптами

Выполнил: Александр Ошаров
Дата: 15.11.2025
Тема: Bash-скрипты с использованием условных конструкций, циклов, функций и внешних файлов

Введение

В рамках данной работы были разработаны и протестированы 5 Bash-скриптов, демонстрирующих использование ключевых конструкций языка Bash: условных операторов `if`, циклов `while`, внутренних функций и взаимодействия между файлами. Все скрипты успешно протестированы в среде WSL (Windows Subsystem for Linux).

Среда выполнения

- **ОС:** Windows 10/11
- **Среда выполнения:** WSL2 (Ubuntu)
- **Типы файлов:** `.sh`

Список скриптов

1. `condition_loop.sh`

Содержание конструкций:

- `if` — условная проверка числа
- `while` — цикл для подсчета

Описание:
Скрипт определяет функцию `check_number`, которая принимает число и проверяет, больше ли оно 10, меньше или равно. Затем в цикле `while` вызывает эту функцию для чисел от 1 до 5.

Результат выполнения:

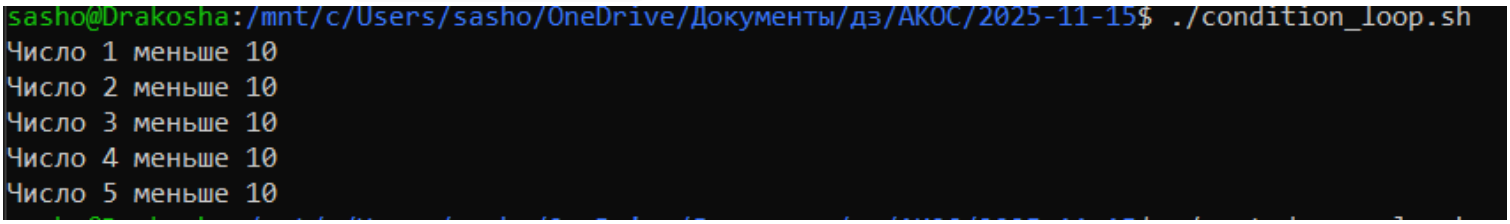
```
Число 1 меньше 10
Число 2 меньше 10
Число 3 меньше 10
```

```
Число 4 меньше 10
Число 5 меньше 10
```

Код:

```bash
#!/bin/bash

# Функция проверки числа
check_number() {
    local num=$1
    if [ $num -gt 10 ]; then
        echo "Число $num больше 10"
    elif [ $num -lt 10 ]; then
        echo "Число $num меньше 10"
    else
        echo "Число $num равно 10"
    fi
}

# Цикл while для подсчета
counter=1
while [ $counter -le 5 ]; do
    check_number $counter
    ((counter++))
done
```

Скриншот выполнения `condition_loop.sh`:



---

2. `nested_functions.sh`

Содержание конструкций:

- `if` — внутри функции `calculate` (в `case`)
- Использование функций, вызываемых из других функций

Описание:
Скрипт демонстрирует вложенность функций. Функция `process_data` вызывает `calculate`, которая выполняет сложение или умножение в зависимости от параметра.

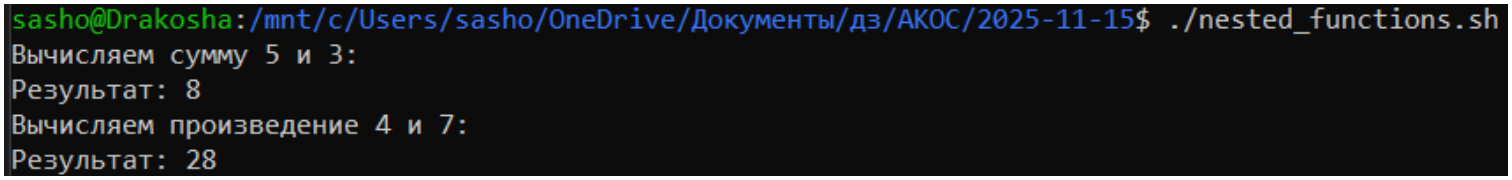
Результат выполнения:

```
Вычисляем сумму 5 и 3:
Результат: 8
Вычисляем произведение 4 и 7:
Результат: 28
```

Код:

```bash
#!/bin/bash

# Внутренняя функция для вычисления
calculate() {
    local operation=$1
    local num1=$2
    local num2=$3
    local result

    case $operation in
        "add")
            result=$((num1 + num2))
            ;;
        "multiply")
            result=$((num1 * num2))
            ;;
        *)
            result=0
            ;;
    esac

    echo $result
}

# Внешняя функция, вызывающая внутреннюю
process_data() {
    echo "Вычисляем сумму 5 и 3:"
    local sum=$(calculate "add" 5 3)
    echo "Результат: $sum"

    echo "Вычисляем произведение 4 и 7:"
    local product=$(calculate "multiply" 4 7)
    echo "Результат: $product"
}

# Вызов основной функции
process_data
```

Скриншот выполнения `nested_functions.sh`:



---

3. `utils.sh`

Содержание конструкций:

- `if` — в функции `is_even` и `perform_calculation`
- Функции: `display_message`, `perform_calculation`, `is_even`

Описание:
Файл с вспомогательными функциями, которые используются в других скриптах. Содержит логику для вывода сообщений, выполнения простых вычислений и проверки четности числа.

Код:

```bash
#!/bin/bash

# Функция для вывода сообщения
display_message() {
    local msg=$1
    echo "[INFO] $msg"
}

# Функция для выполнения вычислений
perform_calculation() {
    local a=$1
    local b=$2

    if [ $a -gt $b ]; then
        echo "Разница: $((a - b))"
    else
        echo "Сумма: $((a + b))"
    fi
}

# Функция для проверки четности
is_even() {
    local num=$1
    if [ $((num % 2)) -eq 0 ]; then
        return 0
    else
        return 1
```

```
    fi
}
```

---

4. `main_script.sh`

Содержание конструкций:

- Скрипт, вызывающий скрипты из других файлов (через `source ./utils.sh`)

Описание:
Основной скрипт, который подключает `utils.sh` с помощью команды `source`. Затем вызывает функции из внешнего файла.

Результат выполнения:

```
Запуск основного скрипта
[INFO] Привет из основного скрипта!
Разница: 5
```

Код:

```bash
#!/bin/bash

# Загрузка вспомогательного скрипта
source ./utils.sh

# Основная логика
echo "Запуск основного скрипта"
display_message "Привет из основного скрипта!"
perform_calculation 10 5
```

Скриншот выполнения `main_script.sh`:

```
sasho@Drakosha:/mnt/c/Users/sasho/OneDrive/Документы/дз/АКОС/2025-11-15$ ./main_script.sh
Запуск основного скрипта
[INFO] Привет из основного скрипта!
Разница: 5
```

---

5. `complex_example.sh`

Содержание конструкций:

- `if` — в функциях `process_numbers` и `analyze_files`
- `while` — в функции `process_numbers`

- Использование функций, вызываемых из других функций
- `for` — в функции `analyze_files`

Описание:
Комплексный скрипт, объединяющий все рассмотренные элементы. Обрабатывает числа в цикле, анализирует файлы в текущей директории, использует вложенные функции.

Результат выполнения:

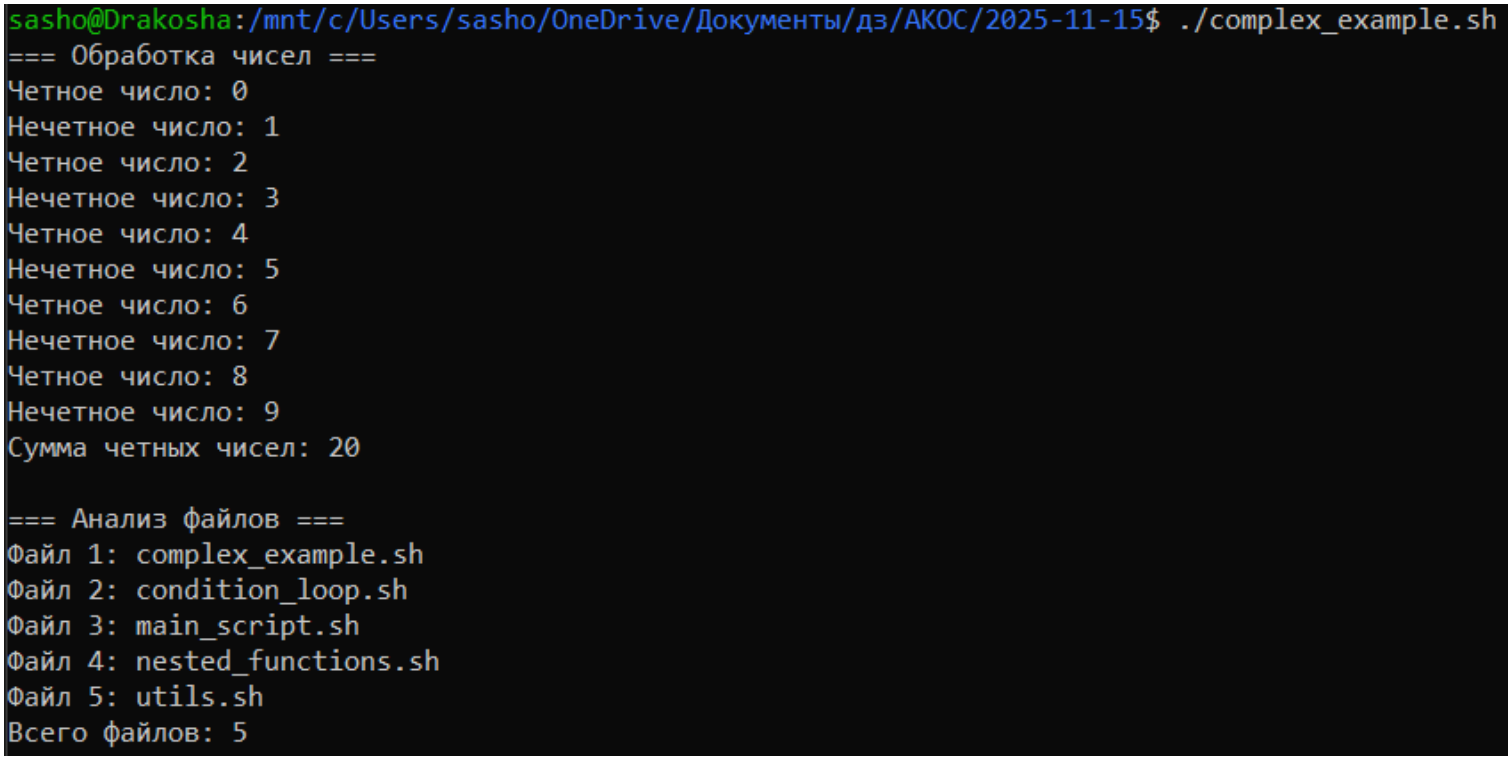
```
=== Обработка чисел ===
Четное число: 0
Нечетное число: 1
Четное число: 2
Нечетное число: 3
Четное число: 4
Нечетное число: 5
Четное число: 6
Нечетное число: 7
Четное число: 8
Нечетное число: 9
Сумма четных чисел: 20

=== Анализ файлов ===
Файл 1: condition_loop.sh
Файл 2: nested_functions.sh
Файл 3: main_script.sh
Файл 4: utils.sh
Файл 5: complex_example.sh
Всего файлов: 5
```

Код:

```bash
#!/bin/bash

# Функция обработки чисел
process_numbers() {
    local limit=$1
    local count=0
    local sum=0

    while [ $count -lt $limit ]; do
        if [ $((count % 2)) -eq 0 ]; then
            echo "Четное число: $count"
            sum=$((sum + count))
        else
            echo "Нечетное число: $count"
        fi
        ((count++))
```

```
    done

    echo "Сумма четных чисел: $sum"
    return $sum
}

# Функция анализа файлов
analyze_files() {
    local dir_path=$1
    local file_count=0

    if [ -d "$dir_path" ]; then
        for file in "$dir_path"/*; do
            if [ -f "$file" ]; then
                ((file_count++))
                echo "Файл $file_count: $(basename "$file")"
            fi
        done
    else
        echo "Директория $dir_path не существует"
    fi

    echo "Всего файлов: $file_count"
}

# Основная функция
main() {
    echo "=== Обработка чисел ==="
    process_numbers 10

    echo -e "\n=== Анализ файлов ==="
    analyze_files "."
}

# Вызов основной функции
main
```

Скриншот выполнения `complex_example.sh`:



Заключение

Все пять скриптов были успешно разработаны, протестированы и запущены в среде WSL. Каждый из них демонстрирует конкретные элементы языка Bash:

- Условные операторы `if` использованы в скриптах 1, 2, 4, 5
- Циклы `while` использованы в скриптах 1, 5
- Использование функций, вызываемых из других функций, реализовано в скриптах 2, 5
- Вызов скриптов из других файлов реализован в скриптах 3 и 4

Файлы приложены к отчету и находятся в работоспособном состоянии.

Приложения

- `condition_loop.sh`
- `nested_functions.sh`
- `utils.sh`
- `main_script.sh`
- `complex_example.sh`