

Домашнее задание

Семинар наставника

Тема 4. Работа с оболочкой Bash для решения задач в области инженерии данных

Все исполняемые файлы находятся в этой же директории.

Задание 1.

Функционал Bash

- Задание: Напишите Bash-скрипт, который выполняет следующие действия:
 1. Создаёт список всех файлов в текущей директории, указывая их тип (файл, каталог и т.д.).
 2. Проверяет наличие определённого файла, переданного как аргумент скрипта, и выводит сообщение о его наличии или отсутствии.
 3. Использует цикл for для вывода информации о каждом файле: его имя и права доступа.

```
# 1. Список всех файлов и их типов ---
echo "Список файлов и их типов в текущей директории:"
for item in *; do
    if [ -d "$item" ]; then
        echo "$item — каталог"
    elif [ -f "$item" ]; then
        echo "$item — файл"
    elif [ -L "$item" ]; then
        echo "$item — символическая ссылка"
    else
        echo "$item — другой тип"
    fi
done

echo

# Проверка наличия файла, переданного как аргумент
if [ -z "$1" ]; then
    echo "Ошибка: не указан файл для проверки."
    echo "Использование: $0 имя_файла"
    exit 1
fi
```

```

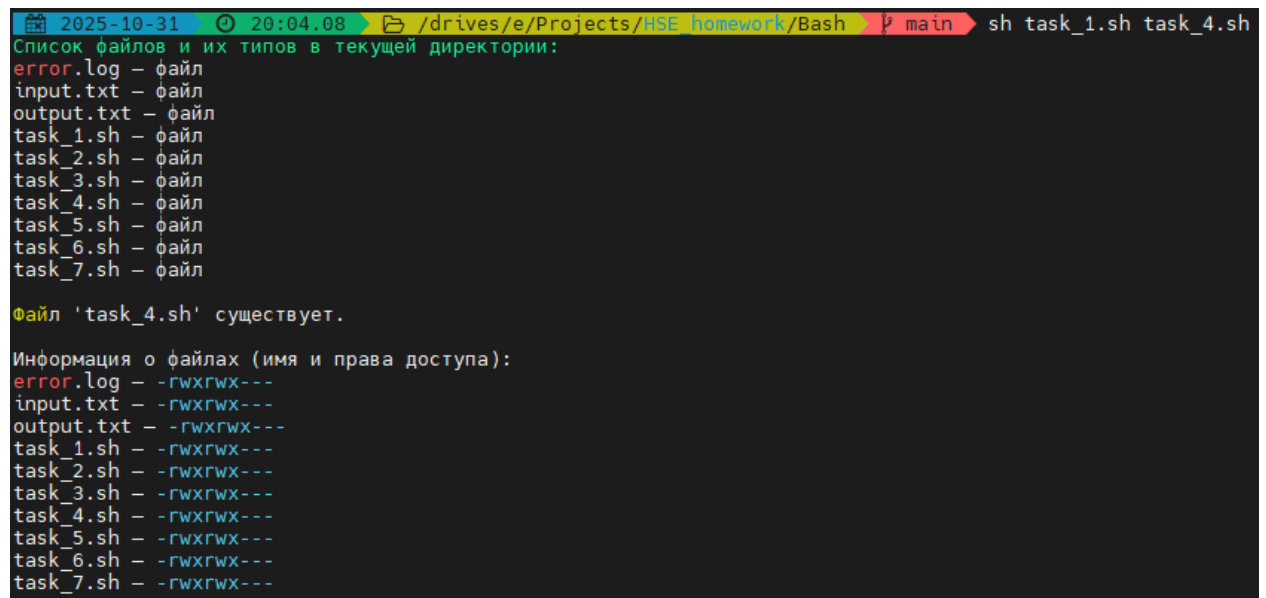
if [ -e "$1" ]; then
    echo "Файл '$1' существует."
else
    echo "Файл '$1' не найден."
fi

echo

# Вывод имени и прав доступа каждого файла
echo "Информация о файлах (имя и права доступа):"
for item in *; do
    if [ -e "$item" ]; then
        perms=$(ls -ld "$item" | awk '{print $1}')
        echo "$item - $perms"
    fi
done

```

Результат выполнения:



```

2025-10-31 20:04.08 /drives/e/Projects/HSE_homework/Bash main sh task_1.sh task_4.sh
Список файлов и их типов в текущей директории:
error.log - файл
input.txt - файл
output.txt - файл
task_1.sh - файл
task_2.sh - файл
task_3.sh - файл
task_4.sh - файл
task_5.sh - файл
task_6.sh - файл
task_7.sh - файл

Файл 'task_4.sh' существует.

Информация о файлах (имя и права доступа):
error.log - -rwxrwx---
input.txt - -rwxrwx---
output.txt - -rwxrwx---
task_1.sh - -rwxrwx---
task_2.sh - -rwxrwx---
task_3.sh - -rwxrwx---
task_4.sh - -rwxrwx---
task_5.sh - -rwxrwx---
task_6.sh - -rwxrwx---
task_7.sh - -rwxrwx---

```

Задание 2.

Переменная PATH

- Задание:

1. Напишите скрипт, который выводит текущее значение переменной PATH и добавляет в неё новую директорию, переданную в качестве аргумента.
2. Объясните, почему изменения переменной PATH, сделанные через терминал, временные, и предложите способ сделать их постоянными. Добавьте команду в файл .bashrc и

продемонстрируйте, как перезапустить терминал для применения изменений.

```
# Проверяем, что передан аргумент (директория)
if [ -z "$1" ]; then
    echo "Использование: $0 путь_к_директории"
    exit 1
fi

# 1. Выводим текущее значение PATH
echo "Текущее значение PATH:"
echo "$PATH"
echo

# 2. Добавляем новый путь, если его ещё нет
if [[ ":$PATH:" != *":$1:"* ]]; then
    export PATH="$PATH:$1"
    echo "Добавлена новая директория: $1"
else
    echo "Директория уже присутствует в PATH."
fi

# 3. Выводим новое значение PATH
echo
echo "Новое значение PATH:"
echo "$PATH"
```

Результат работы:

```
2025-10-31 20:04:24 /drives/e/Projects/HSE_homework/Bash main sh task_2.sh /task_1.sh
Текущее значение PATH:
/usr/bin:/bin:/usr/bin:/drives/c/WINDOWS:/drives/c/WINDOWS/system32:/drives/c/Program Files/Eclipse Adoptium/jdk-21.0.8.9-hotspot/bin:/drives/c/WINDOWS/system32:/drives/c/WINDOWS:/drives/c/WINDOWS/System32/Wbem:/drives/c/WINDOWS/System32/WindowsPowerShell/v1.0:/drives/c/Program Files/NVIDIA Corporation/NVIDIA App/NvDLISR:/drives/c/Program Files/Git/cmd:/drives/c/Program Files/Docker/Docker/resources/bin:/drives/c/Program Files (x86)/NVIDIA Corporation/PhysX/Common:/drives/c/Users/obliw/AppData/Local/Microsoft/WindowsApps:/drives/c/Users/obliw/AppData/Local/Programs/Microsoft VS Code/bin:/drives/c/Users/obliw/AppData/Local/Programs/cursor/resources/app/bin:/drives/c/Program Files/Je
tBrains/PyCharm 2025.2.1/bin:/drives/c/Program Files/JetBrains/IntelliJ IDEA Community Edition 2025.2.2/bin:/drives/c/Users/obliw/anaconda3:/drives/c/Users/obliw/anaconda3/Scripts:/drives/c/Users/obliw/anaconda3/Library/bin:/drives/c/WINDOWS/sysnative
Добавлена новая директория: /task_1.sh

Новое значение PATH:
/usr/bin:/bin:/usr/bin:/drives/c/WINDOWS:/drives/c/WINDOWS/system32:/drives/c/Program Files/Eclipse Adoptium/jdk-21.0.8.9-hotspot/bin:/drives/c/WINDOWS/system32:/drives/c/WINDOWS:/drives/c/WINDOWS/System32/Wbem:/drives/c/WINDOWS/System32/WindowsPowerShell/v1.0:/drives/c/Program Files/NVIDIA Corporation/NVIDIA App/NvDLISR:/drives/c/Program Files/Git/cmd:/drives/c/Program Files/Docker/Docker/resources/bin:/drives/c/Program Files (x86)/NVIDIA Corporation/PhysX/Common:/drives/c/Users/obliw/AppData/Local/Microsoft/WindowsApps:/drives/c/Users/obliw/AppData/Local/Programs/Microsoft VS Code/bin:/drives/c/Users/obliw/AppData/Local/Programs/cursor/resources/app/bin:/drives/c/Program Files/Je
tBrains/PyCharm 2025.2.1/bin:/drives/c/Program Files/JetBrains/IntelliJ IDEA Community Edition 2025.2.2/bin:/drives/c/Users/obliw/anaconda3:/drives/c/Users/obliw/anaconda3/Scripts:/drives/c/Users/obliw/anaconda3/Library/bin:/drives/c/WINDOWS/sysnative:/task_1.sh
task_2.sh: line 48: =====
```

Изменение действует только в текущем сеансе Bash. Когда терминал закрывается, процесс bash завершается. При новом запуске bash создаётся

новый процесс, который инициализируется из файлов настроек (.bashrc, /etc/profile). Поэтому добавленный путь исчезает.

Чтобы путь добавлялся автоматически при каждом запуске терминала, нужно добавить команду export в файл .bashrc.

Открываем файл .bashrc:

```
nano ~/.bashrc
```

Добавляем строку в конец:

```
export PATH="$PATH: E:/Projects/HSE_homework/Bash/task_2.sh"
```

Сохраняем и закрываем.

Применяем изменения:

```
source ~/.bashrc
```

Задание 3.

Управляющие конструкции (условия и циклы)

- Задание: Напишите скрипт, который запрашивает у пользователя ввод числа и затем:
 - Использует if, чтобы проверить, является ли число положительным, отрицательным или нулем, и выводит соответствующее сообщение.
 - Использует while для подсчёта от 1 до введенного числа (если оно положительное).

Скрипт:

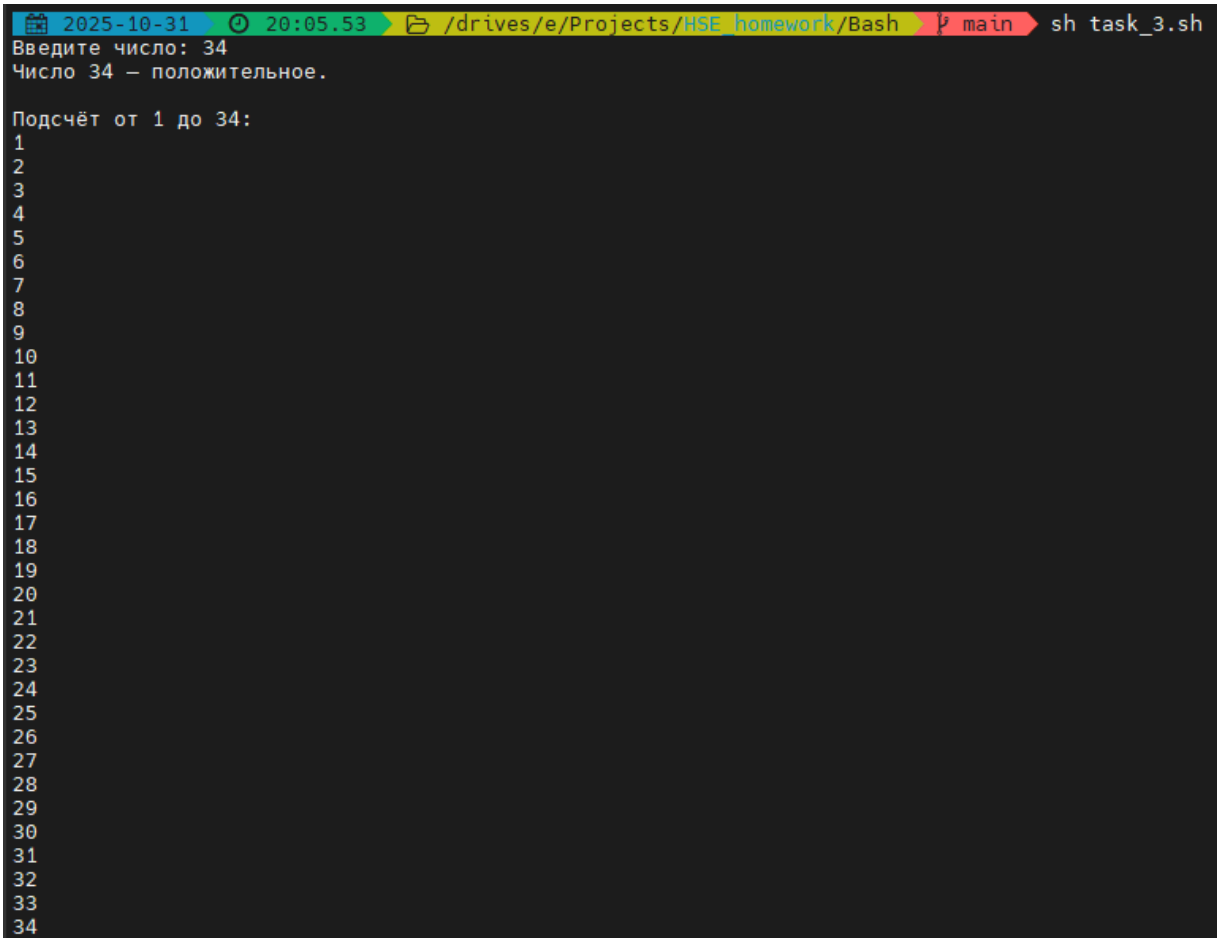
```
# Запрашиваем ввод числа у пользователя
read -p "Введите число: " num

# Проверяем, положительное, отрицательное или ноль
if [ "$num" -gt 0 ]; then
    echo "Число $num — положительное."
elif [ "$num" -lt 0 ]; then
    echo "Число $num — отрицательное."
else
    echo "Число равно нулю."
fi
```

```
echo

# Если число положительное, считаем от 1 до этого числа
if [ "$num" -gt 0 ]; then
    echo "Подсчёт от 1 до $num:"
    i=1
    while [ $i -le $num ]; do
        echo "$i"
        ((i++)) # то же самое, что i=$((i+1))
    done
else
    echo "Подсчёт выполняется только для положительных чисел."
fi
```

Результат выполнения:



```
2025-10-31 20:05.53 /drives/e/Projects/HSE_homework/Bash main sh task_3.sh
Введите число: 34
Число 34 — положительное.

Подсчёт от 1 до 34:
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
```

Задание 4.

Работа с функциями

- Задание: Создайте скрипт с функцией, которая принимает в качестве аргумента строку и выводит её с префиксом "Hello, ". Напишите ещё одну функцию, которая принимает два числа и возвращает их сумму. Вызовите обе функции в скрипте и продемонстрируйте результат.

Скрипт:

```
# Вывод строки с префиксом "Hello, "
say_hello() {
    name="$1"
    echo "Hello, $name!"
}

# Возвращает сумму двух чисел
sum_numbers() {
    a="$1"
    b="$2"
    result=$((a + b))
    echo "$result"
}

echo "====="
echo

# Вызов функции say_hello
say_hello "World"
say_hello "Gleb"

echo

# Вызов функции sum_numbers
num1=5
num2=7
sum=$(sum_numbers "$num1" "$num2")
echo "Сумма чисел $num1 и $num2 равна: $sum"
```

Результат выполнения:

```
2025-10-31 20:11.42 /drives/e/Projects/HSE_homework/Bash main sh task_4.sh
=====
Hello, World!
Hello, Gleb!

Сумма чисел 5 и 7 равна: 12
```

Задание 5.

Управление процессами и фоновое выполнение

- Задание: Напишите скрипт, который запускает три команды `sleep` с разными временами в фоновом режиме. Используйте команды `jobs`, `fg`, `bg`, чтобы продемонстрировать управление этими задачами. Опишите, что вы наблюдали.

Скрипт, который запускает 3 фоновые задачи:

```
echo "Запуск трёх фоновых задач..."

# Запускаем три команды sleep с разными временами
sleep 5 &
sleep 10 &
sleep 15 &

# Показываем список запущенных задач
echo
echo "Активные фоновые задачи:"
jobs
```

Работа с командами `jobs`, `fg`, `bg`:

```
2025-10-31 20:41.05 /drives/e/Projects/HSE_homework/Bash main source task_5.sh
Запуск трёх фоновых задач...

Активные фоновые задачи:
[1] Running      _tob sleep 5 &
[2]- Running      _tob sleep 10 &
[3]+ Running      _tob sleep 15 &

2025-10-31 20:41.07 /drives/e/Projects/HSE_homework/Bash main fg %3
_tob sleep 15

[3]+ Stopped      _tob sleep 15

2025-10-31 20:41.12 /drives/e/Projects/HSE_homework/Bash main bg %3
[3]+ _tob sleep 15 &

[1] Done          _tob sleep 5
[3]+ Exit 1        _tob sleep 15

2025-10-31 20:41.14 /drives/e/Projects/HSE_homework/Bash main jobs
[2]+ Done          _tob sleep 10

2025-10-31 20:41.18 /drives/e/Projects/HSE_homework/Bash main
```

При выполнении команды `fg %3` процесс `sleep 15` был переведён на передний план, и терминал ожидал его завершения.

После нажатия `Ctrl+Z` процесс был приостановлен (статус `Stopped`).

Команда `bg %3` вернула задачу в фоновый режим, и она продолжила выполняться. Команда `jobs` показывала текущее состояние процессов.

После завершения задач `Bash` вывел:

```
[1]+ Done sleep 5
```

```
[2]+ Done sleep 10
```

```
[3]+ Exit 1 sleep 15
```

Код "Exit 1" у третьей задачи означает, что процесс завершился с ошибкой (возможно, из-за потери доступа к `stdin` после перевода между `foreground` и `background`).

Задание 6.

Ввод/вывод и перенаправление

- Задание: Создайте скрипт, который выполняет следующие действия:
 1. Читает данные из файла `input.txt`.
 2. Перенаправляет вывод команды `wc -l` (подсчет строк) в файл `output.txt`.
 3. Перенаправляет ошибки выполнения команды `ls` для несуществующего файла в файл `error.log`.

```
# 1. Чтение данных из файла input.txt
echo "Содержимое файла input.txt:"
cat input.txt
echo

# 2. Подсчет количества строк и запись результата в output.txt
wc -l < input.txt > output.txt
echo "Результат подсчета строк записан в output.txt"
echo

# 3. Попытка вызвать ls для несуществующего файла и запись ошибки в error.log
ls nonexistent_file 2> error.log
echo "Ошибки (если были) записаны в error.log"
```


Результат выполнения:

```
2025-10-31 20:13.44 /drives/e/Projects/HSE_homework/Bash main sh task_6.sh
Содержимое файла input.txt:
g
g
g

Результат подсчета строк записан в output.txt
Ошибки (если были) записаны в error.log
```

Файл input.txt:

```
HSE_homework > Bash > input.txt
1 g
2 g
3 g
4 |
```

Файл output.txt:

```
HSE_homework > Bash > output.txt
1 3
2 |
```

Файл error.log:

```
HSE_homework > Bash > error.log
1 ls: nonexistent_file: No such file or directory
2 |
```

Задание 7.

Использование alias и автодополнение

- Задание: Создайте alias для команды `ls -la` и назовите его `ll`. Напишите команду, чтобы сделать alias постоянным, и объясните, где она должна быть добавлена. Продемонстрируйте использование автодополнения на примере команды `cd`.

Выполнение:

Создание временного alias:

```
alias ll='ls -la'
```

Добавление alias в `.bashrc` для постоянного использования:

```
echo "alias ll='ls -la'" >> ~/.bashrc
```

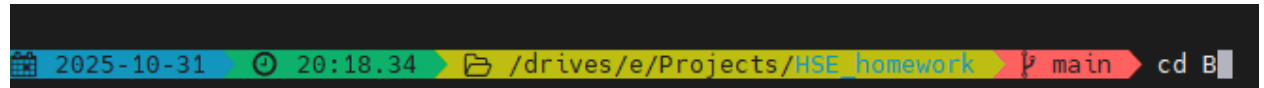
Применение изменений:

```
source ~/.bashrc
```

Автодополнение работает так:

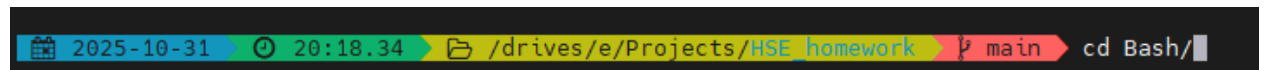
Пример:

Ввожу: cd B

A terminal window with a dark background. The top bar shows the date 2025-10-31, time 20:18.34, and the current directory /drives/e/Projects/HSE_homework on the main branch. The command prompt shows 'cd B' with a cursor at the end.

Нажимаю Tab → автодополнение подставляет:

cd Bash/

A terminal window with a dark background. The top bar shows the date 2025-10-31, time 20:18.34, and the current directory /drives/e/Projects/HSE_homework on the main branch. The command prompt shows 'cd Bash/' with a cursor at the end.