Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Операционные среды и системное программирование»

|  |  |
| --- | --- |
|  |  |

**ОТЧЕТ**

к лабораторной работе № 1

на тему «Скрипты shell»

Выполнил             А. Д. Филипеня

Проверил                          Н. Ю. Гриценко

Минск 2024

**СОДЕРЖАНИЕ**

[1 Постановка задачи 3](#_Toc157722973)

[2 Краткие теоретические сведения 4](#_Toc157722974)

[3 Результаты выполнения лабораторной работы 6](#_Toc157722975)

[Выводы 7](#_Toc157722976)

[Список использованных источников 8](#_Toc157722977)

[Приложение А (обязательное) Листинг исходного кода 9](#_Toc157722978)

# **1 ПОСТАНОВКА ЗАДАЧИ**

Целью выполнения данной лабораторной работы является изучение элементов и конструкций скриптов shell, а также принципов интеграции Unix-программ скриптами shell. В рамках лабораторной работы необходимо реализовать скрипт для оболочки bash, который реализует текстовые часы, меняющие свое расположение в пределах ограниченной области каждые две секунды.

# **2 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

Командный интерпретатор или оболочка – программа, обеспечивающая взаимодействие с пользователем посредством интерфейса командной строки. Примерами оболочки являются, например, оболочка bash операционных систем Unix/Linux или оболочка zsh, считающаяся основным командным интерпретатором в macOS и Kali Linux.[1]

Все оболочки имеют схожие функции и свойства, в соответствии с тем, что они должны выполнять команды пользователя и отображать результаты их выполнения. К функциям оболочки относятся:

– интерпретация командной строки;

– доступ к командам и результатам их выполнения;

– поддержка переменных;

– поддержка специальных символов;

– поддержка зарезервированных слов;

– обработка файлов;

– обработка операций ввода и вывода.

Shell рассматривается как интерпретатор специализированного языка. Это же название закрепилось и за самим языком. Соответственно, существуют различные варианты языка shell, свойственные разным командным интерпретаторам.

Скрипты shell – это текстовые файлы, содержащие команды Unix, которые интерпретируются оболочкой операционной системы.   
Они используются для автоматизации задач, управления системными ресурсами и выполнения различных операций.

Основное достоинство shell-скриптов заключается в том, что в одном скрипте можно объединить большое количество стандартных задач, что обеспечивает экономию времени и сокращение количества возможных ошибок при многократном выполнении однотипных задач.

Shell-скрипт начинается с комбинации символов, которая сообщает,   
что это shell-скрипт. Это символы «#» и «!», после которых указывается интерпретатор.[2]

Для выполнения данной лабораторной работы были использованы следующие сведения и концепции:

1 Скрипты bash: скрипты bash являются подмножеством скриптов shell.

2 Циклы: в shell-скрипте определены две переменные «screen\_cols»   
и «screen\_lines», которые определяют размеры области, в рамках которой появляются текстовые часы. При помощи циклов происходит отрисовка границ выделенной для часов области.

3 Функции: в shell-скрипте определена функция «get\_random\_number»,   
которая генерирует случайное число, в последующем используемое для координат часов, и функция «border», необходимая для отрисовки границ выделенной для часов области.

4 Циклы: в shell-скрипте используется бесконечный цикл «while true», который выполняет основную задачу, заключающуюся в перемещении часов в выделенной области, в течение неопределенного времени.

5 Арифметические операции: в shell-скрипте используются арифметические операции для функции генерации случайных координат «x» и «y» в пределах заданных размеров области, в которой появляются текстовые часы.

6 Встроенные команды: в shell-скрипте используются встроенные команды Unix, такие как «date», «clear», «tput», «sleep», «echo» для работы   
с датой и временем, очистки экрана, управления курсором терминала   
и задержки выполнения, а также для управления выводом соответственно.

7 Форматирование вывода: для форматирования строки времени используется команда «date» с опцией «%T», чтобы получить текущее время в необходимом формате. Также для генерации случайного числа используется команда «date» с опцией «%T», чтобы получить текущее время в секундах.

Таким образом, при использовании всех вышеперечисленных концепций был разработан shell-скрипт, который реализует текстовые часы, меняющие свое расположение в рамках ограниченной области терминала с периодичностью в две секунды.

# **3 РЕЗУЛЬТАТЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ             РАБОТЫ**

В ходе выполнения лабораторной работы был разработан небольшой bash-скрипт, который реализует текстовые часы, меняющие свое расположение в рамках ограниченной области с периодичностью в е секунду. Результат работы bash-скрипта представлен на рисунке 3.1.

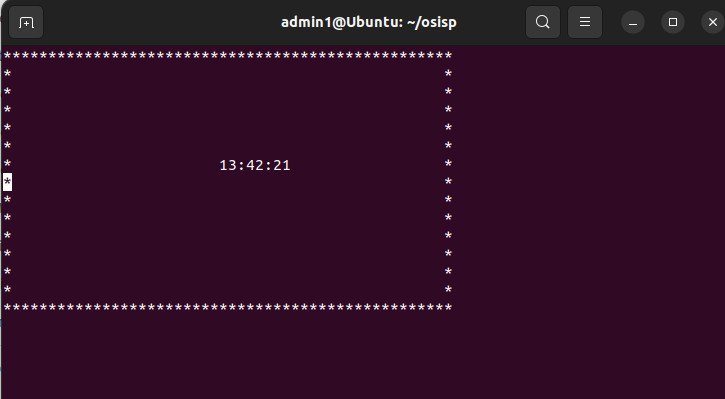


Рисунок 3.1 – Результат работы bash-скрипта

Периодичность изменения расположения часов и размер выделенной для них области можно изменить, обратившись к коду bash-скрипта.

Как итог, при вызове bash-скрипта появляется ограниченная область   
с выводом в ней текстовых часов, изменяющих свое положения в рамках этой область каждые две секунды.

# **ВЫВОДЫ**

В ходе лабораторной работы были изучены элементы и конструкции скриптов shell, а также принципы интеграции Unix-программ скриптами shell.   
Кроме того, в ходе лабораторной работы был разработан скрипт для оболочки bash, который реализует текстовые часы, изменяющие   
свое расположение в рамках ограниченной области через каждые две секунды.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Оболочка пользователя в Linux [Электронный ресурс]. – Режим доступа: https://white55.ru/shells.html. – Дата доступа: 30.01.2024.

[2] Скрипты shell [Электронный ресурс]. – Режим доступа: <https://support.apple.com/ru-ru/guide/terminal/>. – Дата доступа: 30.01.2024.

[3] BashTutorials [Электронный ресурс]. – Режим доступа:   
https://prog /bash-tutorials/. – Дата доступа: 30.01.2024.

## **ПРИЛОЖЕНИЕ А**

## **(обязательное)**

## **Листинг исходного кода**

Листинг 1 – Программный код разработанного приложения

#!/bin/bash

a=1664525

b=1013904223

m=4294967296 #2\*\*32

cols=$(tput cols)

screen\_cols=$((cols \* 50 / 100))

lines=$(tput lines)

screen\_lines=$((lines \* 60 / 100))

#(a\*seconds+b)modm

get\_random\_number()

{

seconds=$(date +%s)

random=$(( (a\*seconds+b)%m ))

echo $random

}

border()

{

for (( i=0; i<=$((screen\_cols+9-1)); i++)); do

echo -n "\*"

done

for (( i=0; i<=$((screen\_lines-1)); i++)); do

echo "\*"

done

for (( i=1; i<=$((screen\_cols+9)); i++)); do

echo -n "\*"

done

for (( i=0; i<=$((screen\_lines)); i++)); do

tput cup $i $((screen\_cols+9))

echo "\*"

done

}

while true; do

clear

x=$(( $(get\_random\_number)%screen\_cols ))

y=$(( $(get\_random\_number)%screen\_lines ))

border

tput cup $y $x

current\_time=$(date +%T)

echo $current\_time

sleep 2s

done