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             К. А. Тимофеев

Проверил                          Н. Ю. Гриценко

Минск 2024

**СОДЕРЖАНИЕ**

1 Постановка задачи3

2 Краткие теоретические сведения4

3 Результаты выполнения лабораторной работы6

Выводы7

Краткие теоретические сведения8

Приложение А (обязательное) Листинг исходного кода 9

# **1 ПОСТАНОВКА ЗАДАЧИ**

Основной задачей данной лабораторной работы является изучение командной оболочки *shell*, ее возможностей и принципов написания скриптов для выполнения в данной командной оболочке. Для выполнения данной лабораторной работы необходимо написать скрипт для оболочки *bash*, который реализует небольшой циферблат из *ASCII*-символов, отображающий текущее время и случайно меняющий свое расположение с течением времени.

# **2 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

Командная оболочка в операционных системах представляет собой интерфейс между пользователем и ядром операционной системы. Она позволяет пользователю взаимодействовать с операционной системой путем ввода текстовых команд. Командная оболочка интерпретирует эти команды, запускает соответствующие программы или процессы, управляет ими и возвращает результат пользователю.

Все оболочки разделяют общие функции и возможности. К ним относятся:

– интерпретация команд командной строки;

– вывод результатов выполнения команд командной строки;

– обработка файлов;

– объявление переменных и доступ к ним;

– обработка операций ввода и вывода и т. д.

Скрипты *shell* – это текстовые файлы с командами *UNIX*, которые могут интерпретироваться командной оболочкой. Они используются для автоматизации задач в командной оболочке *shell*.[1]

Каждый *shell*-скрипт начинается со специальной комбинации символов, обозначающей начало скрипта. Это символы «#» и «*!*», после которых указывается абсолютный путь к конкретной оболочке, в которой будет выполнятся скрипт.[2]

Для выполнения данной лабораторной работы была использована командная оболочка *bash*. Из возможностей скриптов *shell* были использованы следующие сведения и концепции:

1 Глобальные переменные: В *shell*-скрипте определены две глобальные переменные «\_*ROWS*» и «\_*COLUMNS*», которые задают размеры выводимой рамки. Также определена глобальная переменная «\_*SEED*», которая используется в генерации случайных чисел.

2 Функции: логика программы разделена на функции. В функции «*printBorder*» находится логика отрисовки рамки. Функция «*printClocks*» рисует циферблат из символов *ASCII*, функция «*printArrows*» рисует стрелки на циферблате. Так же в программу была добавлена функция «*random*», которая генерирует случайное целое положительное число.

3 Циклы: в shell-скрипте используется бесконечный цикл «*while true*», который выполняет основную программу в течение неопределенного времени.

4 Арифметические операции: в *shell*-скрипте используются арифметические операции для генерации случайных координат «*x*» и «*y*»   
в пределах заданных размеров области, в которых будет нарисован циферблат. Так же при помощи математических операций в функции «*random*» вычисляется псевдослучайное число.

5 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.

6 Форматирование вывода: для форматирования строки времени используется команда «*date*» с опцией «*+%H*», чтобы получить текущее время в необходимом формате.

Таким образом, при использовании всех вышеперечисленных концепций был разработан *shell*-скрипт, который реализует небольшой циферблат из символов *ASCII*, меняющий свое расположение на ограниченной области терминала с периодичностью в одну секунду.

# **3 РЕЗУЛЬТАТЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ             РАБОТЫ**

В ходе выполнения лабораторной работы был разработан *bash*-скрипт, который реализует небольшой циферблат из символов *ASCII*, меняющий свое расположение на ограниченной рамкой области периодичностью в одну секунду. Результат работы *bash*-скрипта представлен на рисунке 3.1.

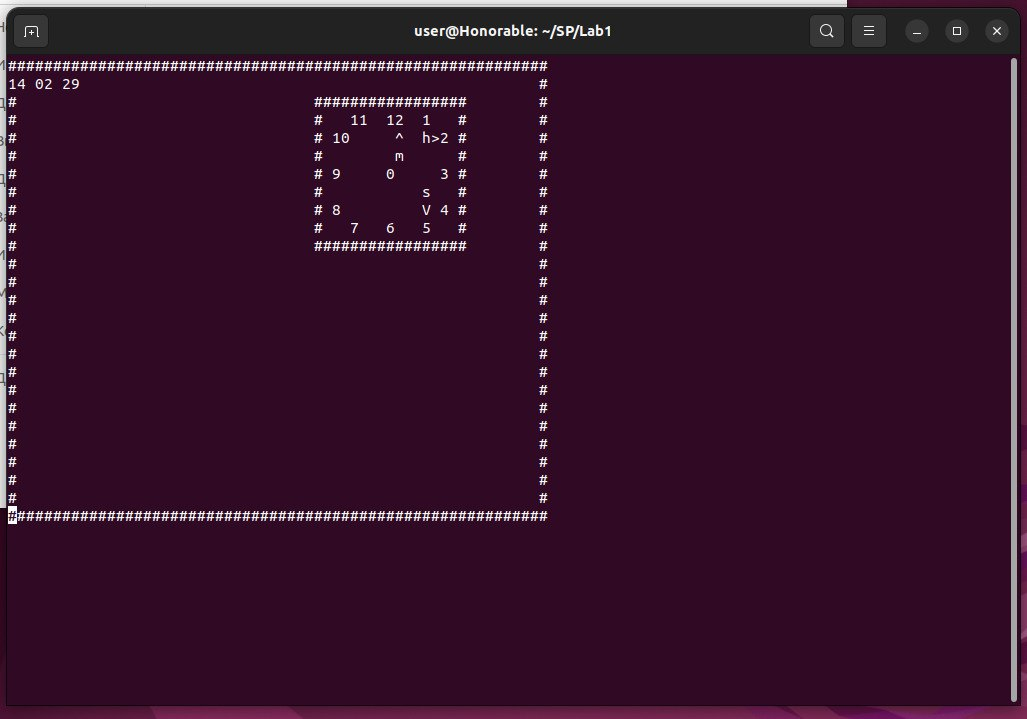


Рисунок 3.1 – Результат работы bash-скрипта

Размер области и интервал изменения расположения текстовых часов можно изменить только обращаясь к коду *bash*-скрипта.

В итоге, при вызове *bash*-скрипта появляется ограниченная область   
с выводом в ней циферблата с интервалом в одну секунду.

# **ВЫВОДЫ**

В ходе лабораторной работы были изучены основные элементы и возможности *shell*, а также принципы интеграции *Unix*-программ скриптами *shell*.   
Кроме того, в ходе лабораторной работы был разработан скрипт для командной оболочки *bash*, который реализует небольшой циферблат из символов *ASCII*, меняющий свое расположение на ограниченной области раз в секунду.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Командные оболочки (shells) [Электронный ресурс]. – Режим доступа: https://docs.altlinux.org/ru-RU/archive/2.2/html-single/master/install-html/ch06s04.html. – Дата доступа: 03.02.2024

[2] Bash-скрипты / Хабр [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/726316/. – Дата доступа: 04.02.2024

## **ПРИЛОЖЕНИЕ А**

## **(обязательное)**

## **Листинг исходного кода**

Листинг 1 – Программный код разработанного приложения

#!/bin/bash

\_COLUMNS=60

\_ROWS=25

\_SEED=1

function randomNumber(){

(( \_SEED = \_SEED \* 1103515245 + 12345 ))

local res=$(( (\_SEED / 65536) % 32768 ))

return $res

}

function smth(){

local val=$1; local letter=$2;

local indent=$3; local x=$4; local y=$5;

case $val in

1)

tput cup $((y+1+indent)) $((x+12))

printf "^"

tput cup $((y+2+indent)) $((x+12))

printf "$letter"

;;

2)

tput cup $((y+2)) $((x+14-indent))

printf ">"

tput cup $((y+2)) $((x+14-indent-1))

printf "$letter"

;;

3)

tput cup $((y+4)) $((x+14-indent))

printf ">"

tput cup $((y+4)) $((x+14-indent-1))

printf "$letter"

;;

4)

tput cup $((y+6)) $((x+14-indent))

printf ">"

tput cup $((y+6)) $((x+14-indent-1))

printf "$letter"

;;

5)

tput cup $((y+7 - indent)) $((x+12))

printf "V"

tput cup $((y+7-indent-1)) $((x+12))

printf "$letter"

;;

6)

tput cup $((y+7 - indent)) $((x+8))

printf "V"

tput cup $((y+7-indent-1)) $((x+8))

printf "$letter"

;;

7)

tput cup $((y+7 - indent)) $((x+4))

printf "V"

tput cup $((y+7-indent-1)) $((x+4))

printf "$letter"

;;

8)

tput cup $((y+6)) $((x+2+indent))

printf "<"

tput cup $((y+6)) $((x+2+indent+1))

printf "$letter"

;;

9)

tput cup $((y+4)) $((x+2+indent))

printf "<"

tput cup $((y+4)) $((x+2+indent+1))

printf "$letter"

;;

10)

tput cup $((y+2)) $((x+3+indent))

printf "<"

tput cup $((y+2)) $((x+3+indent+1))

printf "$letter"

;;

11)

tput cup $((y+1+indent)) $((x+5))

printf "^"

tput cup $((y+2+indent)) $((x+5))

printf "$letter"

;;

0)

tput cup $((y+1+indent)) $((x+9))

printf "^"

tput cup $((y+2+indent)) $((x+9))

printf "$letter"

;;

esac

tput cup 0 0

}

function printArrows(){

local x=$1; local y=$2;

local h=$[10#$3];

local m=$[10#$4];

local s=$[10#$5]

smth $(((m - m%5) / 5)) "m" 1 $x $y

smth $(((s - s%5) / 5)) "s" 1 $x $y

smth $((h%12)) "h" 1 $x $y

}

function printClocks(){

randomNumber

local rand=$?

# echo " random result is $rand "

local x=$((rand % (\_COLUMNS - 16)))

randomNumber

local rand2=$?

local y=$((rand2 % (\_ROWS - 8)))

# local x=15

# local y=8

tput cup $y $x

printf "#################"

tput cup $((y+1)) $x

printf "# 11 12 1 #"

tput cup $((y+2)) $x

printf "# 10 2 #"

tput cup $((y+3)) $x

printf "# #"

tput cup $((y+4)) $x

printf "# 9 0 3 #"

tput cup $((y+5)) $x

printf "# #"

tput cup $((y+6)) $x

printf "# 8 4 #"

tput cup $((y+7)) $x

printf "# 7 6 5 #"

tput cup $((y+8)) $x

printf "#################"

tput cup 0 0

printArrows $x $y $1 $2 $3

}

function printBorder(){

for ((i=0; i <$\_COLUMNS; ++i))

do

printf "#"

done

printf "\n"

for ((i=1; i < $\_ROWS; ++i))

do

printf "#"

for ((j=0; j<$\_COLUMNS-2;++j))

do

printf " "

done

printf "#\n"

done

for ((i=0; i <$\_COLUMNS; ++i))

do

printf "#"

done

}

function printTime(){

tput cup 1 0

printf "$1 $2 $3"

printClocks $1 $2 $3

}

\_SEED=$( date "+%s" )

while true;do

clear

tput cup 0 0

printBorder

hour=$(date "+%H")

minute=$(date "+%M")

second=$(date "+%S")

printTime $hour $minute $second

tput cup $\_ROWS 0

sleep 1

done