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             В. Д. Ключинский

Проверил                          Н. Ю. Гриценко

Минск 2024

**СОДЕРЖАНИЕ**

[1 Цель работы 3](#_Toc158376840)

[2 Краткие теоретические сведения 4](#_Toc158376841)

[3 Полученные результаты 5](#_Toc158376842)

[Выводы 6](#_Toc158376843)

[Список использованных источников 7](#_Toc158376844)

[Приложение А (обязательное) Листинг исходного кода 8](#_Toc158376845)

**1 ЦЕЛЬ РАБОТЫ**

Целью выполнения лабораторной работы является изучение элементов и конструкций скриптов *shell*: переменных, параметров, ветвлений, циклов, вычислений, команд *shell* и вызовов внешних программ (*shell*, *sed*, *awk*, *wget*, различные фильтры и т.д.) для решения достаточно сложной задачи, имеющей практическое значение, а также принципов интеграции *Unix*-программ скриптами *shell* на примере создания программы «*screen saver*» для консоли.

**2 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

*Shell* – это интерфейс командной строки, который позволяет запускать команды, программы и сценарии оболочки. Оболочки бывают разных видов. Каждая оболочка имеет свой набор хорошо известных команд и функций.

*Bash* – это особый тип оболочки, а оболочка – это общий термин, применимый к интерфейсу командной строки. *Bash* является оболочкой, установленной по умолчанию в большинстве систем *Linux*, хотя доступно множество различных типов оболочек. *Bash* обладает большим количеством функций и возможностей, чем другие оболочки, что делает его популярным выбором среди разработчиков и опытных пользователей.[1]

Разница между *bash* и *shell* заключается в том, что *bash* является расширенным набором *shell*, что означает, что он поддерживает функциональность *shell*, а также добавляет дополнительные расширения. Большинство команд работают так же, как и в *shell*, тогда как оболочка представляет собой интерфейс пользователя к операционной системе, который позволяет пользователю взаимодействовать с компьютером.[2]

Программирование *bash*-скриптов относится к написанию командных скриптов для интерпретатора командной оболочки *UNIX*-подобных систем, таких как *Linux* и *MacOS*. *Bash* является одним из наиболее популярных интерпретаторов командной оболочки в этих системах.

Синтаксис *bash*-скриптов очень похож на синтаксис командной строки *UNIX*. Основные элементы включают в себя команды, переменные, условные операторы, циклы и функции. Функции позволяют группировать набор команд для выполнения определенной задачи. Нельзя не отметить, они могут быть вызваны из разных частей скрипта. *Bash*-скрипты также могут принимать аргументы из командной строки при запуске, что позволяет передавать параметры скрипту.[3]

Для выполнения поставленного задания были использованы следующие инструменты:

1 Команда echo для вывода текста в консоль *Linux*.

2 Циклы *for* и *while* для обеспечения непрерывной генерации изображения таймера через фиксированный временной интервал.

3 Функция генерации псевдослучайных натуральных чисел для определения новой позиции часов на экране консоли, а также функция, которая очищает консоль.

Таким образом, можно понять, что для реализации поставленного задания достаточно изучения основ создания *bash*-скриптов.

# **3 ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ**

Результаты работы скрипта представлены на рисунке 3.1 и рисунке 3.2

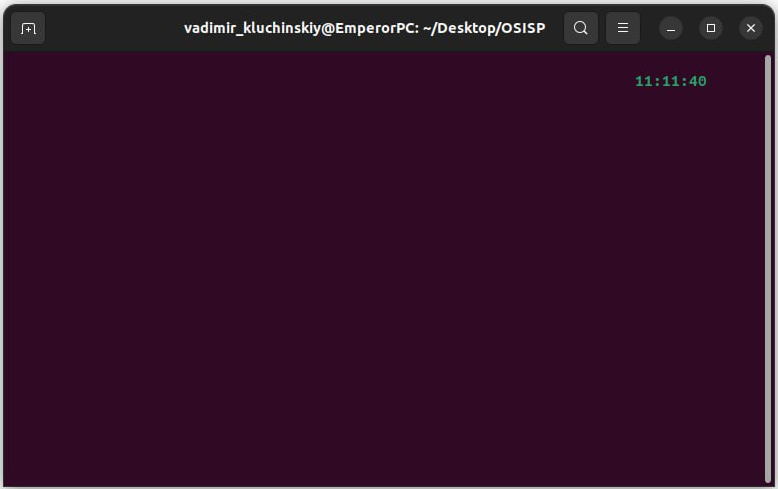


Рисунок 3.1 – Результат работы скрипта

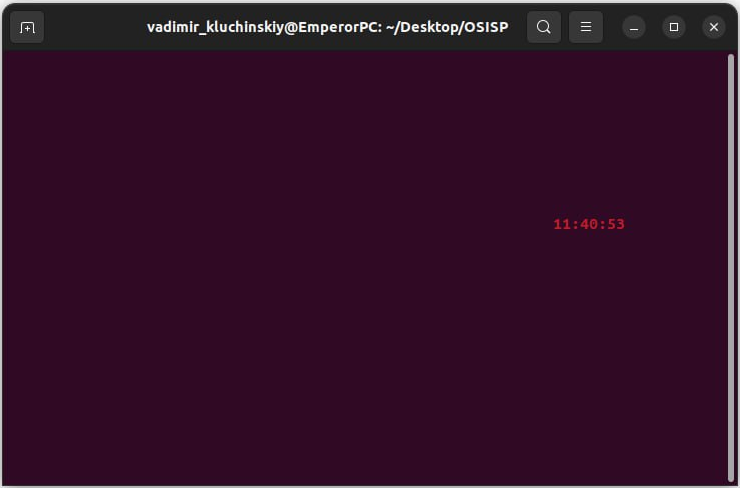


Рисунок 3.2 – Результат работы скрипта

Таким образом, созданный скрипт реализует поставленную задачу: изменяет положение часов через равные временные интервалы случайным образом.

# **ВЫВОДЫ**

В ходе выполнения данной лабораторный работы был разработан полноценный «*screen saver*», который меняет положение и цвет на экране консоли случайным образом. Были освоены важные инструменты скриптов shell: команды, переменные, ветвления, циклы, функции.

Для редактирования кода был выбран редактор текста «*VSCode*».

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Тейнсли, Д. Linux и другие UNIX-подобные операционные системы: программирование в Shеll. / Д. Тейнсли. – СПб.: БВХ-Петербург, 2006. – 1056 с: ил.

[2] Щупаков, Ю. Руководство по командам и shell-программированию, операционная система Linux. / Ю. Щупаков. – СПб: Питер, 2008. – 592 с.

[3] BashTutorials [Электронный ресурс]. – Режим доступа: https://prog /bash-tutorials/. – Дата доступа: 01.02.2024.

# **ПРИЛОЖЕНИЕ А**

# **(обязательное)**

# **Листинг исходного кода**

Листинг 1 – Программный код разработанного приложения:

#!/bin/bash

cleanup() {

# Reset terminal to normal state

stty echo # Restore echo

tput cnorm # Restore cursor

tput sgr0 # Reset terminal text attributes

clear # Clear screen

exit 0 # Exit script

}

trap cleanup SIGINT

RANDOM=1

Multiplier=12791

A=27259

Modulus=43717

generate\_random() {

next=$(((RANDOM\*Multiplier+A) % Modulus));

}

print\_current\_time() {

tput cup $1 $2

echo -e "${3}$(date +"%T")"

tput sgr0

}

generate\_color() {

colors=("30" "31" "32" "33" "34" "35" "36" "37")

rand\_index=$((RANDOM % ${#colors[@]}))

echo -e "\033[1;${colors[$rand\_index]}m"

}

clear

tput civis

stty -echo

# Main program loop

seconds\_interval=3

current\_color=$(generate\_color)

while true; do

generate\_random

rows=$(tput lines)

cols=$(tput cols)

row=$((RANDOM % (rows-3) + 1))

col=$((RANDOM % (cols-10) + 1))

clear

for ((i = 0; i < seconds\_interval; i++)); do

print\_current\_time $row $col $current\_color

sleep 1

done

current\_color=$(generate\_color)

done