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

ОТЧЁТ

к лабораторной работе №1

на тему

**СКРИПТЫ SHELL.**

Выполнил: студент гр.253505 Снежко М.А.

Проверил: ассистент кафедры информатики Гриценко Н.Ю.

Минск 2025

СОДЕРЖАНИЕ

[1 Формулировка задачи 3](#_Toc179742261)

[2 Краткие теоритические сведения 4](#_Toc179742262)

[3 Описание функций программы 5](#_Toc179742263)

[Заключение 6](#_Toc179742264)

[Список использованных источников 7](#_Toc179742265)

[Приложение А (обязательное) 8](#_Toc179742266)

# 1 ФОРМУЛИРОВКА ЗАДАЧИ

Целью данной лабораторной работы является изучение методов создания скриптов для работы в консольной среде *Linux*, а также освоение принципов реализации динамических приложений. Основное внимание уделяется разработке консольного хранителя экрана, представляющего собой текстовые часы, которые периодически меняют своё положение на экране. Работа направлена на изучение алгоритмов генерации псевдослучайных чисел, управления выводом данных в консоли, а также обработки системных сигналов для управления работой приложения.

В процессе выполнения лабораторной работы необходимо реализовать скрипт, который отображает текстовые часы в формате часы минуты. Для изменения их положения на экране требуется использовать генератор псевдослучайных чисел, основанный на мультипликативном алгоритме. Генератор инициализируется текущим временем, а его значения применяются для расчёта координат текста. Расположение и цвет часов должны обновляться через фиксированные промежутки времени, задаваемые пользователем. Если интервал не указан, программа использует значение по умолчанию.

Реализация скрипта предполагает динамическое управление выводом данных на экран с учётом размеров окна терминала. Алгоритм генерации координат должен быть адаптирован к текущим параметрам консоли, чтобы текст всегда оставался в пределах видимой области. Цвет текста также изменяется случайным образом, что делает приложение более наглядным. В ходе работы скрипт должен демонстрировать устойчивость при длительном выполнении и стабильную работу в различных условиях.

Результаты поиска структурированы таким образом, чтобы обеспечить удобный анализ и возможность экспорта. В случае вывода данных в файл применяется структура *reg*-файлов, где сохраняются имена ключей, значения и их типы. Обработка поиска предполагает анализ всех поддеревьев заданного ключа с целью нахождения совпадений, а также их последующую обработку для формирования финального отчета.

Выполнение данной лабораторной работы направлено на освоение основ программирования на языке *Bash*, включая работу с текстовым выводом, реализацию алгоритмов псевдослучайной генерации чисел и обработку сигналов. Полученные знания и навыки могут быть использованы для создания более сложных приложений, таких как текстовые интерфейсы, интерактивные консольные утилиты или инструменты для автоматизации задач.

Таким образом, данная лабораторная работа направлена на освоение элементов и конструкций *shell*: переменных, параметров, ветвлений, циклов, вычислений, команд shell и вызовов внешних программ, а также принципов интеграций Unix-программ скриптами shell Полученные знания могут быть использованы для создания более сложных инструментов.

2 КРАТКИЕ ТЕОРИТИЧЕСКИЕ СВЕДЕНИЯ

*UNIX* – многопользовательская, многозадачная операционная система (ОС), разработанная для гибкости и адаптивности. Первоначально разработанная в 1970-х годах, *Unix* была одной из первых ОС, написанных на языке программирования *C*. С момента своего появления операционная система *Unix* и ее ответвления оказали глубокое влияние на компьютерную и электронную промышленность, предлагая переносимость, стабильность и совместимость в различных гетерогенных средах и типах устройств. [1]

Хранитель экрана (*screensaver*) – это программа, которая запускается в случае бездействия пользователя, обычно для предотвращения выгорания экрана или создания визуальных эффектов. В данном случае создается консольный хранитель экрана, представляющий собой текстовые часы, которые динамически перемещаются по экрану. Для реализации перемещений используется генератор псевдослучайных чисел, создающий координаты отображения часов. Основой для генератора может служить мультипликативный алгоритм, работающий с целыми числами. Это простой и эффективный метод генерации псевдослучайных чисел, который позволяет получать последовательности на основе текущего времени, задаваемого при инициализации генератора.

В операционных системах семейства *Unix/Linux* скрипты *shell* позволяют реализовывать сложные задачи за счет использования встроенных инструментов и утилит. Такие утилиты, как *awk*, *sed*, *tput* и другие, обеспечивают гибкость в обработке данных, управлении выводом и взаимодействии с пользователем. Например, команда *tput* используется для определения размеров терминала и управления курсором, что особенно важно для динамического отображения текста в консоли [2].

Основная среда взаимодействия с *UNIX* – командная строка. Суть её в том, что каждая строка, передаваемая пользователем системе, – это команда, которую та должна выполнить. Пока не нажата клавиша *Enter*, строку можно редактировать, затем она отсылается системе. Команды интерпретируются и выполняются специальной программой – командной оболочкой или *shell*. Через командную оболочку производится управление пользовательскими процессами – для этого используются средства межпроцессного обмена. Командная оболочка непосредственно связана с терминалом, через который осуществляется передача управляющих последовательностей и текста. [3]

Программирование на *shell* позволяет эффективно интегрировать различные команды *Unix* для выполнения задач, связанных с обработкой данных, автоматизацией процессов и управлением визуальными элементами. В данном случае скрипт демонстрирует возможности создания интерактивных приложений, которые сочетают динамическое отображение данных и управление с использованием базовых конструкций *shell*, таких как циклы, функции и обработка сигналов.

3 ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ

Разработанная программа представляет собой консольный хранитель экрана, который отображает текстовые часы, динамически меняющие свое положение на экране. Основной задачей программы является генерация случайных координат для отображения часов и их периодическое обновление. Для этого в программе используется мультипликативный алгоритм генерации псевдослучайных чисел, инициализируемый текущим временем. Программа также изменяет цвет текста часов, делая отображение более разнообразным.

Инициализация программы начинается с задания начальных параметров, таких как интервал обновления. Если пользователь при запуске не указывает интервал, используется значение по умолчанию. Основой для генератора случайных чисел является алгоритм, который перемножает текущий *seed* на фиксированное значение, а результат вычисляется по модулю. Этот метод позволяет получать новые координаты для отображения часов в пределах размеров терминала.

После генерации новых координат программа проверяет их корректность, чтобы текстовые часы всегда находились в видимой области экрана. С помощью команды *tput* считываются текущие размеры терминала, и координаты ограничиваются допустимыми значениями. Для отображения часов используется команда *tput* *cup*, которая перемещает курсор в заданные координаты, а цвет текста определяется случайным образом с помощью сгенерированного значения. Результат работы программы представлен на рисунке 3.1.

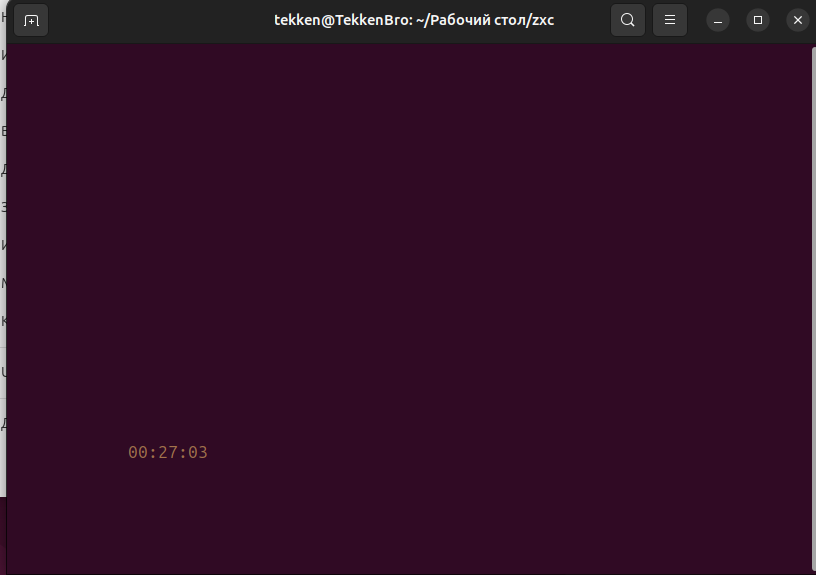


Рисунок 3.1 – Результат работы программы

Таким образом, программа демонстрирует использование конструкций *shell*-скриптов для решения практической задачи. Она позволяет изучить применение циклов, функций, работы с системными сигналами и управления выводом в консоли. Реализация скрипта иллюстрирует возможности интеграции инструментов *Linux*, таких как *tput* и команды обработки времени, для создания динамических приложений.

# ЗАКЛЮЧЕНИЕ

Таким образом, в рамках данной лабораторной работы была разработана программа – консольный хранитель экрана, отображающий текстовые часы, которые динамически меняют своё положение на экране. Основная цель работы заключалась в изучении конструкций и механизмов работы скриптов *shell*, таких как генерация случайных чисел, управление выводом на экран, обработка сигналов и циклические конструкции. Реализация программы позволила на практике изучить методы создания интерактивных приложений для консольной среды и принципы работы с системными данными, такими как размеры терминала.

Разработанный скрипт представляет собой универсальное решение, которое демонстрирует генерацию случайных координат, изменение цвета текста и управление положением часов на экране. Основной алгоритм основан на мультипликативной генерации псевдослучайных чисел, что позволило эффективно реализовать динамическое отображение. Программа использует данные о текущем времени для инициализации генератора, благодаря чему обеспечивается уникальность результатов при каждом запуске. Гибкость реализации позволяет пользователю задавать интервал обновления позиции часов, что делает приложение легко адаптируемым под различные сценарии.

Для корректной работы программы и удобства пользователя была реализована обработка системных сигналов. При завершении работы с помощью *Ctrl+C* программа очищает экран и восстанавливает начальные настройки терминала, исключая возможные проблемы с выводом. Это обеспечивает стабильность и безопасность работы приложения. Кроме того, программа учитывает размеры терминала, чтобы часы всегда оставались в видимой области экрана, что делает её универсальной для использования в различных средах. В процессе выполнения лабораторной работы исследовались параметры, влияющие на производительность и стабильность приложения. Это включало тестирование на различных размерах терминала, проверку работы алгоритма генерации случайных чисел и оценку корректности обработки сигналов. Результаты работы продемонстрировали эффективность и удобство программы, а также её устойчивость при непрерывной работе в течение длительного времени.

Таким образом, лабораторная работа позволила не только изучить основные конструкции скриптов shell, но и освоить практические аспекты их применения для решения реальных задач. Реализация программы дала возможность углубить знания о работе с терминалом, системными данными и сигналами, а также продемонстрировала важность корректного и безопасного взаимодействия с ресурсами операционной системы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Thecode "Обзор UNIX" – Электронный ресурс. Режим доступа: https://thecode.media/unix.

[2] Gnu.org "Возможности Bash" – Электронный ресурс. Режим доступа: https://www.gnu.org/software/bash/manual/bash.html.

[3] Heap.altlinux "Коммандная строка UNIX" – Электронный ресурс. Режим доступа: http://heap.altlinux.org/modules/unix\_base.dralex/ch02.html.

ПРИЛОЖЕНИЕ А

(обязательное)

**Исходный код программы**

#!/bin/bash

declare -i seed current\_x current\_y color\_code

handle\_sigint() {

clear

exit 1

}

initialize\_application() {

echo -ne "\e[?25l"

trap 'echo -ne "\e[?25h"' EXIT

trap handle\_sigint SIGINT

local interval=${1:-1}

if ! [[ "$interval" =~ ^[0-9]+$ ]] || (( interval < 1 )); then

echo "Error: Interval must be a positive integer."

exit 1

fi

seed=$(date +%s)

}

initialize\_rng() {

local a=16807

local m=2147483647

seed=$(( (a \* seed) % m ))

}

generate\_new\_position() {

local term\_lines=$(tput lines)

local term\_cols=$(tput cols)

max\_x=$(( term\_lines - 1 ))

(( max\_x < 0 )) && max\_x=0

max\_y=$(( term\_cols - 8 ))

(( max\_y < 0 )) && max\_y=0

initialize\_rng

current\_x=$(( seed % (max\_x + 1) ))

initialize\_rng

current\_y=$(( seed % (max\_y + 1) ))

}

generate\_new\_color() {

initialize\_rng

color\_code=$(( 31 + (seed % 7) ))

}

validate\_positions() {

local term\_lines=$(tput lines)

local term\_cols=$(tput cols)

(( current\_x > term\_lines - 1 )) && current\_x=$(( term\_lines - 1 ))

(( current\_x < 0 )) && current\_x=0

(( current\_y > term\_cols - 8 )) && current\_y=$(( term\_cols - 8 ))

(( current\_y < 0 )) && current\_y=0

}

print\_clock() {

clear

tput cup "$current\_x" "$current\_y"

echo -ne "\e[${color\_code}m$(date +%T)\e[0m"

}

main\_loop() {

local interval=$1

((interval == 0)) && interval=1

local counter=0

while true; do

if (( counter == 0 )); then

generate\_new\_position

generate\_new\_color

validate\_positions

fi

print\_clock

sleep 1

counter=$(( (counter + 1) % interval ))

done

}

initialize\_application "$@"

main\_loop "$1"