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:
студент гр. 153503
Татаринов В.В.

Проверил:
Гриценко Н.Ю.

Минск 2024

СОДЕРЖАНИЕ

1 ЦЕЛЬ РАБОТЫ

Изучить элементы и конструкции скриптов *shell*: переменные, параметры, ветвления, циклы, вычисления, команды *shell* и вызовы внешних программ для решения достаточно сложной задачи, имеющей практическое значение, а также принципы интеграции *Unix*-программ скриптами *shell*. Написать скрипт для оболочки *shell,* который представляет собой реализацию карточной игры «Мемо».

2 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Shell – интерпретатор командной строки, представляющий собой программу, которая принимает команды от пользователя и исполняет их. К ключевым функциям *shell* относятся такие операции, как взаимодействие с пользователем, редактирование командной строки, история команд, обработка шаблонов имен, перенаправление потоков ввода/вывода команд, управление заданиями.

Кроме того, *shell* – это специализированный язык программирования, в котором есть переменные, конструкции, циклы, ветвления, функции [1].

Shell работает подобно оболочке, которая окружает ядро операционной системы и предоставляет пользователям доступ к различным функциям и сервисам. Она позволяет запускать сторонние программы, создавать и запускать скрипты и автоматизировать различные задачи. Эту оболочку также можно использовать для управления удаленными серверами через протоколы командной строки [2].

Ввод и вывод распределяется между тремя стандартными потоками:

1 *Stdin* – стандартный входной поток. Он обычно переносит данные от пользователя к программе. Программы, которые предполагают стандартный ввод, обычно получают входные данные от устройства типа клавиатура. Стандартный ввод прекращается по достижении *EOF*, который указывает на то, что данных для чтения больше нет. Примером команды стандартного ввода является *cat. Cat* отправляет полученные входные данные на дисплей терминала в качестве стандартного вывода и останавливается после того, как получает *EOF*.

2 *Stdout* – стандартный выходной поток. Он записывает данные, сгенерированные программой. Когда стандартный выходной поток не перенаправляется в какой-либо файл, он выводит текст на дисплей терминала. В качестве примера можно привести команду *echo.* По умолчанию эта команда выводит на экран любой аргумент, который передается ему в командной строке. При выполнении *echo* без каких-либо аргументов, выводится пустая строка.

3 *Stderr* – стандартный поток ошибок. Он записывает ошибки, возникающие в ходе исполнения программы. Как и в случае стандартного вывода, по умолчанию этот поток выводится на терминал дисплея. В качестве примера можно запустить команду *ls*, указав в качестве аргумента имя несуществующего каталога. Так как такого каталога не существует, на дисплей терминала будет выведен текст стандартной ошибки [3].

3 ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

В результате выполнения лабораторной работы был написан скрипт для оболочки *shell,* который представляет собой реализацию карточной игры «Мемо».

Результат работы скрипта представляет собой таблицу, каждой ячейке которой соответствует определенная карточка, информацию о совершенном игроком ходе, очки игроков (рисунок 1).

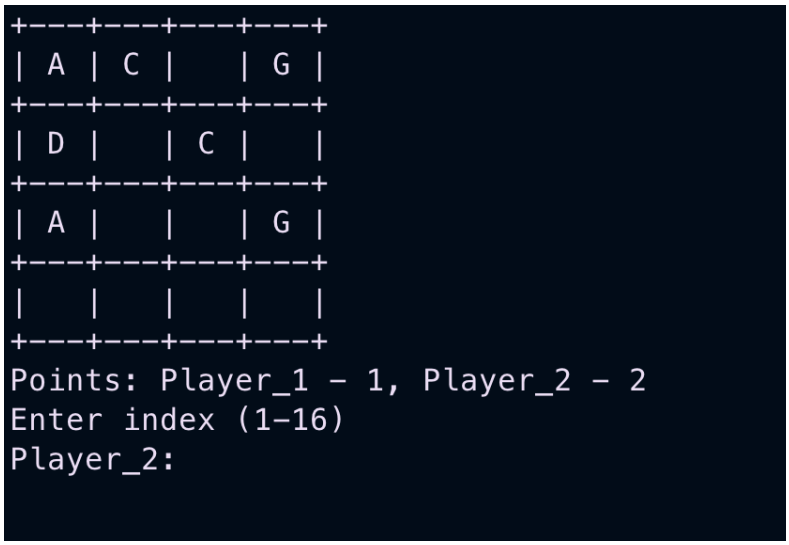

Рисунок 1 – Реализация карточной игры «Мемо»

ВЫВОДЫ

В результате выполнения лабораторной работы были изучены элементы и конструкции скриптов *shell*: переменные, параметры, ветвления, циклы, вычисления, команды. Был написан скрипт для оболочки *shell,* который представляет собой реализацию карточной игры «Мемо».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Что такое shell и зачем он нужен [Электронный ресурс]. – Режим доступа: https://www.inp.nsk.su/~bolkhov/teach/inpunix/shell.ru.html.

[2] Shell операционная система [Электронный ресурс]. – Режим доступа: https://uchet-jkh.ru/i/cto-takoe-shell-operacionnaya-sistema-prostymi-slovami/.

[3] Перенаправление ввода/вывода [Электронный ресурс]. – Режим доступа: https://selectel.ru/blog/tutorials/linux-redirection/.

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг кода

Листинг 1 – Файл *memo.sh*:

```
#!/bin/zsh

function read_results {
    local log_file="$(dirname "$0")/log.txt"
    if [ -f "$log_file" ]; then
        echo "Game history:"
        cat "$log_file"
    else
        echo "File $log_file does not exist"
    fi
}

function log_results {
    local result="$1"
    local log_file="$(dirname "$0")/log.txt"
    local timestamp=$(date +"%Y-%m-%d %H:%M:%S")
    echo "[$timestamp] $result" >> "$log_file"
}

function shuffle_cards {
    local i j temp
    for ((i = 16 - 1; i > 0; i--)); do
        j=$((RANDOM % i + 1))
        temp=${cards[$i]}
        cards[$i]=${cards[$j]}
        cards[$j]=$temp
    done
}

board=(
    " " " " " " " " " " " " " "
    " " " " " " " " " " " " " "
    " " " " " " " " " " " " " "
    " " " " " " " " " " " " " "
)
cards=(
    "A" "A" "B" "B"
    "C" "C" "D" "D"
    "E" "E" "F" "F"
```

```
    "G" "G" "H" "H"
)

player1_pairs=0
player2_pairs=0

function display_board {
    clear
    echo "+---+---+---+---+"
    echo "| ${board[1]} | ${board[2]} | ${board[3]} | ${board[4]} |"
    echo "+---+---+---+---+"
    echo "| ${board[5]} | ${board[6]} | ${board[7]} | ${board[8]} |"
    echo "+---+---+---+---+"
    echo "| ${board[9]} | ${board[10]} | ${board[11]} | ${board[12]} |"
    echo "+---+---+---+---+"
    echo "| ${board[13]} | ${board[14]} | ${board[15]} | ${board[16]} |"
    echo "+---+---+---+---+"
}

function check_winner {
    local matched_count=0
    for cell in "${board[@]}"; do
        if [[ "$cell" == " " ]]; then
            return
        fi
    done

    echo "The game is over! All the cards are open."

    if (( player1_pairs > player2_pairs )); then
        echo "Player_1 wins! $player1_pairs - $player2_pairs"
        log_results "Player_1 wins! $player1_pairs - $player2_pairs"
    elif (( player2_pairs > player1_pairs )); then
        echo "Player_2 wins! $player2_pairs - $player1_pairs"
        log_results "Player_2 wins! $player2_pairs - $player1_pairs"
    else
        echo "Draw! $player1_pairs - $player2_pairs"
        log_results "Draw! $player1_pairs - $player2_pairs"
    fi

    read_results

    exit 0
```

```
}

function main {
    local current_player=1
    local selected=(
    shuffle_cards
    while true; do
        display_board

        if (( ${#selected[@]} == 2 )); then
            if [[ "${cards[selected[1]]}" == "${cards[selected[2]]}" ]]; then
                echo "Player_$current_player found a matching pair!"
                if ((current_player == 1)); then
                    ((player1_pairs++))
                else
                    ((player2_pairs++))
                fi
            else
                board[selected[1]]=" "
                board[selected[2]]=" "
                echo "The cards didn't match."
                ((current_player == 1)) && current_player=2 ||
current_player=1
            fi
            selected=()
        fi
        check_winner
        echo "Points: Player_1 - $player1_pairs, Player_2 - $player2_pairs"
        echo "Enter index (1-16)"
        echo -n "Player_$current_player: "
        read index

        if [[ "$index" =~ ^[1-9]$ || "$index" =~ ^1[0-6]$ ]] && [[
"${board[index]}" == " " ]]; then
            selected+=($index)
            board[index]=${cards[index]}
        else
            echo "Invalid index or card has already been matched."
            sleep 2
        fi
    done
}


main
```