**Лабораторна робота Bash**

РОЗРОБКА ГРИ НА МОВІ BASH

**Мета:** використовуючи команди bash створити гру.

**Хід роботи:**

Обрано для створення гру змійка. Правила гри наступні: у гравця є поле, в межах якого він може рухати змійку. Коли гравець входить в край поля з одної сторони, голова змійки переноситься на протилежну сторону. На полі з’являються яблука, які змійка має «їсти» аби збільшуватися. Якщо голова змійки стикається з частиною її тіла – гра закінчується поразкою. Мета гри – отримання якнайвищого рахунку, який зберігається між ігровими сесіями.

Оскільки гра буде консольною, то програмування почато зі створення функцій взаємодії з консоллю, в тому числі друк символа в певному місці та завершення виконання програми після натиснення будь-якої кнопки.

Лістинг файлу console.sh:

#!/bin/bash

# Перенесення курсора на задану позицію

move\_cursor\_to\_position() {

    local row="$1"

    local col="$2"

    # Move cursor to the specified row and column

    echo -e -n "\e[${row};${col}H"

}

# Запис символу на задану позицію

write\_char\_at\_position() {

    local row="$1"

    local col="$2"

    local char="$3"

    move\_cursor\_to\_position "$row" "$col"

    echo -n "$char"

}

# Відображення границь вікна

draw\_window\_borders() {

    local height="$1"

    local width="$2"

    local char="$3"

    # Малювання вертикальних границь

    for ((i=0; i < height; i++)); do

        write\_char\_at\_position "$i" 0 "$char"

        write\_char\_at\_position "$i" "$((width - 1))" "$char"

    done

    # Малювання горизонтальних границь

    for ((i=0; i < width; i++)); do

        write\_char\_at\_position 0 "$i" "$char"

        write\_char\_at\_position "$((height - 1))" "$i" "$char"

    done

}

# Завершення гри

exit\_on\_input() {

    local row="$1"

    move\_cursor\_to\_position "$row" 0

    # Відновлення видимості курсора

    tput cnorm

    # Відображення повідомлення про завершення гри

    read -n 1 -s -r -p "Press any key to exit..."

    clear

    move\_cursor\_to\_position 0 0

}

Наступним кроком створено файл variables.sh для локалізації усіх змінних, значення яких можуть бути змінені з часом, як розміри поля, символи поля, змійки та яблука і таке інше.

Через особливості bash та відсутність вбудованих структур даних, обрано реалізацію збереження координат у рядку. Для операцій додання, пошуку та видалення координат створено ряд функцій у файлі coordinates\_functions.sh.

Лістинг файлу coordinates\_functions.sh:

#!/bin/bash

# Обмежена реалізація структури Черга (FIFO) на основі рядка, оскільки можливості мови bash обмежені

# Додавання координати в кінець списку координат

coords\_append() {

    local x=$1

    local y=$2

    coordinates+="($x,$y) "

}

# Перевірка чи координати містяться в списку координат

coords\_contain() {

    local x="$1"

    local y="$2"

    local element\_to\_check="($x,$y)"

    # echo "Ele: $element\_to\_check"

    # Перевірка чи координати містяться в списку координат за допомогою регулярного виразу

    if [[ $coordinates == \*"$element\_to\_check"\* ]]; then

        return 0

    else

        return 1

    fi

}

# Видалення найстарішої координати зі списку координат

coords\_pop() {

    # Отримання найстарішої координати

    local temp="${coordinates%% \*}"

    # Видалення дужок з координати та занесення в глобальну змінну

    temp="${temp//(/}"

    last\_coords="${temp//)/}"

    # Видалення найстарішої координати зі списку координат

    coordinates="${coordinates#\* }"

}

Створено файл game.sh з циклом гри, що опрацьовує ввід користувача, виведення інформації, створення яблук, процесу їх «поїдання», переміщення змійки та завершення гри.

Лістинг файлу game.sh:

#!/bin/bash

set -o allexport

source console.sh

source variables.sh

source coordinates\_functions.sh

set +o allexport

# Оголошення глобальних змінних

export coordinates="" last\_coords=""

apple\_x=0

apple\_y=0

score=0

highscore=0

# Отримання рахунку з файлу

if [ -f "$HIGHSCORE\_FILE" ] ; then

    highscore=$(cat "$HIGHSCORE\_FILE")

fi

# Створення яблука

spawn\_apple() {

    apple\_x=$((RANDOM % (WINDOW\_HEIGHT-3) + 2))

    apple\_y=$((RANDOM % (WINDOW\_WIDTH-3) + 2))

    # Якщо координати яблука збігаються з координатами гравця, то генеруємо нові координати

    while coords\_contain "$apple\_x" "$apple\_y" ; do

        apple\_x=$((RANDOM % (WINDOW\_HEIGHT-3) + 2))

        apple\_y=$((RANDOM % (WINDOW\_WIDTH-3) + 2))

    done

    # Відображення яблука

    write\_char\_at\_position "$apple\_x" "$apple\_y" "$APPLE\_CHAR"

}

# Головний цикл гри

run\_game\_loop() {

    # Встановлення початкових координат гравця в центрі вікна

    local x=$((WINDOW\_HEIGHT/2))

    local y=$((WINDOW\_WIDTH/2))

    # Створення гравця

    coords\_append "$x" "$y"

    write\_char\_at\_position "$x" "$y" "$PLAYER\_CHAR"

    # Створення яблука

    spawn\_apple

    # Відображення рекорду

    write\_char\_at\_position $((WINDOW\_HEIGHT/2 + 1)) $((WINDOW\_WIDTH+5)) "Highest score: $highscore"

    # Цикл гри

    while true; do

        # Зчитування вводу з клавіатури з таймаутом

        read -r -t "$GAME\_TICK\_LENGTH" -n 1 -s holder && input="$holder"

        # Якщо ввід не є прийнятним, він ігнорується

        if [[ ! $input =~ [^wasde] ]]; then

            direction="$input"

        fi

        # В залежності від напрямку руху гравця, змінюються його координати.

        # Також тут реалізовано вихід з гри та ігронування вводу доки він не буде прийнятним

        case "$direction" in

            w) ((x--));;

            s) ((x++));;

            a) ((y--));;

            d) ((y++));;

            e) break;;

            \*) continue

                # \*)

        esac

        # Якщо відлагоджувальний режим включений, то відображається додаткова інформація

        if [ "$DEBUG" -eq 0 ] ; then

            write\_char\_at\_position $((WINDOW\_HEIGHT+1)) "$WINDOW\_WIDTH" "LC: $last\_coords"

            write\_char\_at\_position $((WINDOW\_HEIGHT+2)) "$WINDOW\_WIDTH" "RX: $rx"

            write\_char\_at\_position $((WINDOW\_HEIGHT+3)) "$WINDOW\_WIDTH" "RY: $ry"

            write\_char\_at\_position $((WINDOW\_HEIGHT+4)) "$WINDOW\_WIDTH" "Coords: $coordinates"

        fi

        # Відображення рахунку

        write\_char\_at\_position $((WINDOW\_HEIGHT/2)) $((WINDOW\_WIDTH+5)) "Score: $score"

        # Якщо гравець не з'їв яблуко, то видаляється хвіст змії

        if [ "$x" -ne "$apple\_x" ] || [ "$y" -ne "$apple\_y" ] ; then

            coords\_pop

            IFS=',' read -r rx ry <<< "$last\_coords"

            write\_char\_at\_position "$rx" "$ry" " "

        fi

        # Перевірка чи гравець не вийшов за межі вікна та перенесення на протилежний бік вікна

        if [ "$x" -lt 2 ]; then

            x=$((WINDOW\_HEIGHT-2))

        fi

        if [ "$x" -gt $((WINDOW\_HEIGHT-2)) ]; then

            x=2

        fi

        if [ "$y" -lt 2 ]; then

            y=$((WINDOW\_WIDTH-2))

        fi

        if [ "$y" -gt $((WINDOW\_WIDTH-2)) ]; then

            y=2

        fi

        # Перевірка чи гравець не зіткнувся з собою (тобто програв)

        if coords\_contain "$x" "$y" ; then

            write\_char\_at\_position $((WINDOW\_HEIGHT/2)) $((WINDOW\_WIDTH/2-3)) "\*DEAD\*"

            break

        fi

        # Додавання нової координати гравця та відображення нової голови змії

        coords\_append "$x" "$y"

        write\_char\_at\_position "$x" "$y" "$PLAYER\_CHAR"

        # Якщо гравець з'їв яблуко, то генерується нове яблуко та збільшується рахунок

        if [ "$x" -eq "$apple\_x" ] && [ "$y" -eq "$apple\_y" ] ; then

            spawn\_apple

            score=$((score+1))

        fi

        # Видалення хвоста змії

        if ! coords\_contain "$rx" "$ry" ; then

            write\_char\_at\_position "$rx" "$ry" " "

        fi

    done

    if [ "$highscore" -lt "$score" ] ; then

        echo "$score" > "$HIGHSCORE\_FILE"

    fi

}

Створено файл main.sh, який користувач буде запускати для початку гри. В ньому курсор в консолі встановлюється прозорим, консоль очищується та друкується ігрове поле, після чого запускається ігровий цикл. Коли ігровий цикл закінчується, виконання файлу завершується викликом функції, що повертає видимість курсору та повідомляє користувача, що виконання програми завершиться після того, як він натисне будь-яку кнопку, після чого консоль очищується, а курсор переноситься на початок консолі.

Лістинг файлу main.sh:

#!/bin/bash

# Імпортування функцій та змінних з інших файлів

set -o allexport

source console.sh

source variables.sh

source game.sh

set +o allexport

# Встановлення прозорого курсора

tput civis

clear

# Встановлення ігрового розмірів вікна

draw\_window\_borders "$WINDOW\_HEIGHT" "$WINDOW\_WIDTH" "$WINDOW\_CHAR"

# Запуск головного циклу гри

run\_game\_loop

# Завершення гри

exit\_on\_input "$WINDOW\_HEIGHT"

Після написання застосунок протестовано. Рисунки процесу гри:

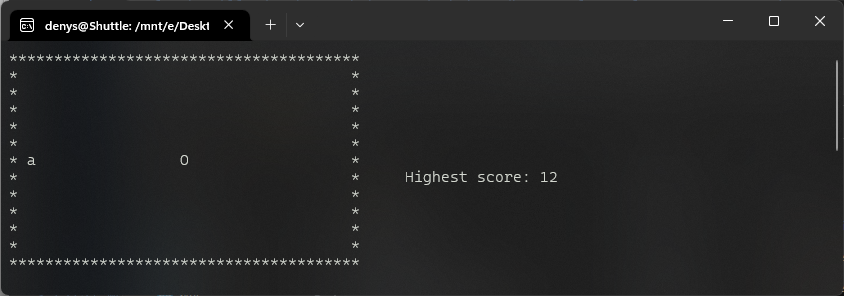


Рисунок 1. Початковий екран гри.

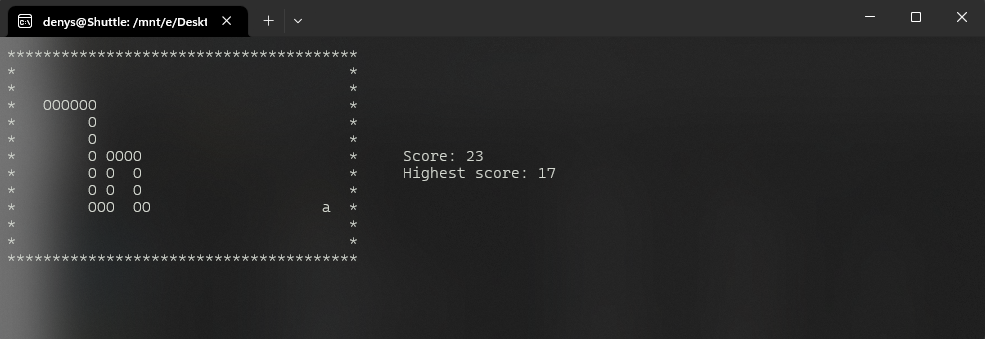


Рисунок 2. Процес гри.

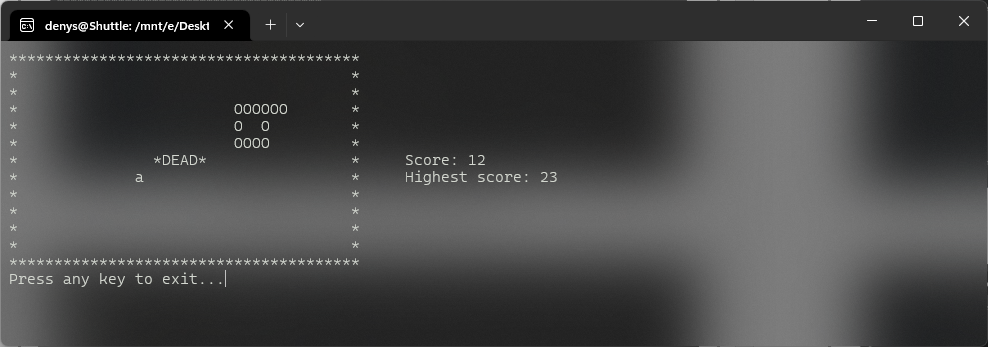


Рисунок 3. Програш.

Файли програми викладено на Gihub за посиланням: <https://github.com/DeNice-r/bash-snake>

**Висновок:** під час виконання даної лабораторної роботи я вивчив більшість синтаксичних особливостей bash, а також на початковому рівні навчився створювати програми, що включають в себе маніпуляції з даними різних типів та файлами.