**LAB ASSIGNMENT # 3**

**  
Attock Campus**

**Submitted By:**

Neha Zainab  
(SP21-BCS-024)

**Submitted To:**

Sir Bilal Bukhari

**Subject:**

Compiler Construction

**Date:**

26 MAY 2024

# LAB TASK (GAME)

#include <stdio.h>

#include <stdbool.h>

#include <stdlib.h>

#include <time.h>

#include <unistd.h>

#include <termios.h>

#define SCREEN\_WIDTH 50

#define SCREEN\_HEIGHT 20

#define PLAYER\_SYMBOL 'A'

#define ENEMY\_SYMBOL 'V'

#define BULLET\_SYMBOL '^'

#define MAX\_ENEMIES 5

typedef struct {

int x, y;

} Entity;

typedef struct {

Entity player;

Entity enemies[MAX\_ENEMIES];

Entity bullet;

int score;

bool gameOver;

} Game;

// Function to set terminal to non-canonical mode

void setNonCanonicalMode() {

struct termios oldt, newt;

tcgetattr(STDIN\_FILENO, &oldt);

newt = oldt;

newt.c\_lflag &= ~(ICANON | ECHO);

tcsetattr(STDIN\_FILENO, TCSANOW, &newt);

}

// Function to restore terminal to canonical mode

void restoreCanonicalMode() {

struct termios oldt;

tcgetattr(STDIN\_FILENO, &oldt);

oldt.c\_lflag |= (ICANON | ECHO);

tcsetattr(STDIN\_FILENO, TCSANOW, &oldt);

}

void init(Game \*game) {

// Initialize player

game->player.x = SCREEN\_WIDTH / 2;

game->player.y = SCREEN\_HEIGHT - 2;

// Initialize enemies

for (int i = 0; i < MAX\_ENEMIES; ++i) {

game->enemies[i].x = rand() % SCREEN\_WIDTH;

game->enemies[i].y = 1;

}

// Initialize bullet

game->bullet.x = -1;

game->bullet.y = -1;

game->score = 0;

game->gameOver = false;

}

void draw(const Game \*game) {

system("clear");

// Draw player

printf("\033[%d;%dH%c", game->player.y, game->player.x, PLAYER\_SYMBOL);

// Draw enemies

for (int i = 0; i < MAX\_ENEMIES; ++i) {

printf("\033[%d;%dH%c", game->enemies[i].y, game->enemies[i].x, ENEMY\_SYMBOL);

}

// Draw bullet

if (game->bullet.y != -1) {

printf("\033[%d;%dH%c", game->bullet.y, game->bullet.x, BULLET\_SYMBOL);

}

// Print score

printf("\033[%d;0HScore: %d\n", SCREEN\_HEIGHT + 1, game->score);

}

void update(Game \*game) {

// Move bullet

if (game->bullet.y != -1) {

game->bullet.y--;

// Check bullet collision with enemies

for (int i = 0; i < MAX\_ENEMIES; ++i) {

if (game->bullet.y == game->enemies[i].y && game->bullet.x == game->enemies[i].x) {

game->score++;

game->bullet.y = -1;

game->enemies[i].y = -1;

game->enemies[i].x = -1;

}

}

}

// Move enemies

for (int i = 0; i < MAX\_ENEMIES; ++i) {

if (game->enemies[i].y != -1) {

game->enemies[i].y++;

// Check enemy collision with player

if (game->enemies[i].y == game->player.y && game->enemies[i].x == game->player.x) {

game->gameOver = true;

return;

}

// Check enemy reaching the bottom

if (game->enemies[i].y >= SCREEN\_HEIGHT) {

game->gameOver = true;

return;

}

}

}

}

void handleInput(Game \*game) {

// Move player

char input;

read(STDIN\_FILENO, &input, 1);

switch (input) {

case 'a':

if (game->player.x > 0)

game->player.x--;

break;

case 'd':

if (game->player.x < SCREEN\_WIDTH - 1)

game->player.x++;

break;

case ' ':

// Shoot bullet

if (game->bullet.y == -1) {

game->bullet.x = game->player.x;

game->bullet.y = game->player.y - 1;

}

break;

default:

break;

}

}

int main() {

srand(time(NULL));

Game game;

init(&game);

// Set terminal to non-canonical mode for non-blocking input

setNonCanonicalMode();

while (!game.gameOver) {

draw(&game);

handleInput(&game);

update(&game);

usleep(100000); // sleep for 100 milliseconds

}

// Restore terminal to canonical mode

restoreCanonicalMode();

printf("\033[%d;0HGame Over! Your score: %d\n", SCREEN\_HEIGHT + 2, game.score);

return 0;

}