FINAL PROJECT

REPORT 4

Tri Bui

828135536

COMPE271

Professor Kenneth Arnold

March 20, 2023

1. Initial Design and Description

A math related video game with X-axis moving character, the other objects move down on Y-axis with value that have math ( + x / - with a number) to win the game get those objects to get the biggest number to kill an random number amount of enemies.

1. How it work

- Switching to another way of printing map. Using an Array as the map for the game and using Rand to make math equation

- Rand generated number of enemies that gradually increase overtime

1. Function of the code

- print\_board: to print game board

- generate\_random\_number: generate random number

- generate\_math\_equation: generate math equation

- handle\_player\_input: handling input of moving left and right, and others

- kbhit() : implement of kbhit from conio.h on Windows because the library is not available on Linux

1. Progress:

Currently trying to work on kbhit, having problem with ncurses library.

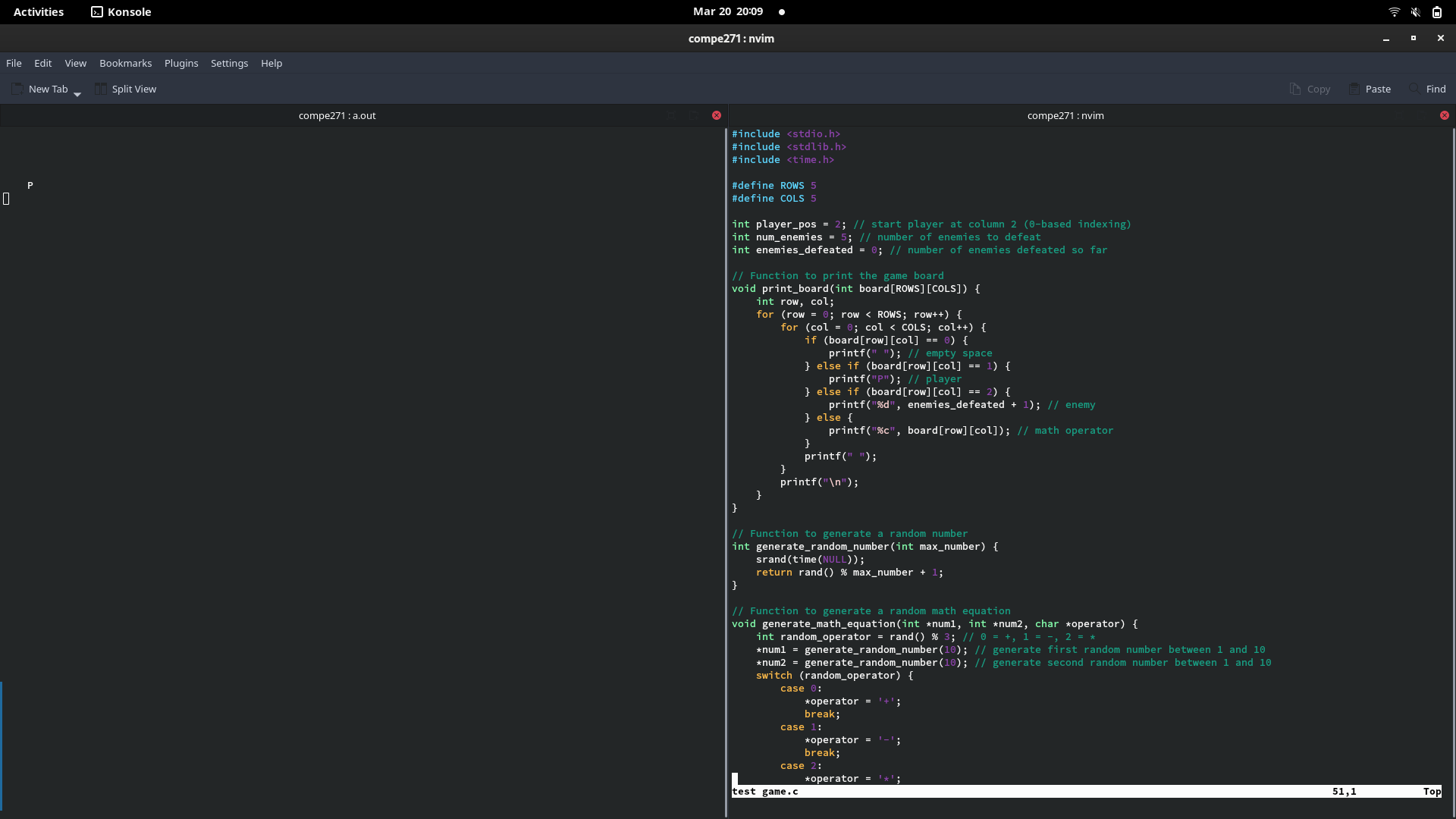
Recently changed the whole program into another way, but currently having problem with setting up input (currently have to press enter for every input which make the game play not smooth)

Moving left and right working, generate math and other element currently not working.

Currently working on the screen refresh and the input.

Will work on MIPS soon after getting these fixed.

1. Screenshot:



1. Code:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#define ROWS 5

#define COLS 5

int player\_pos = 2; // start player at column 2 (0-based indexing)

int num\_enemies = 5; // number of enemies to defeat

int enemies\_defeated = 0; // number of enemies defeated so far

// Function to print the game board

void print\_board(int board[ROWS][COLS]) {

int row, col;

for (row = 0; row < ROWS; row++) {

for (col = 0; col < COLS; col++) {

if (board[row][col] == 0) {

printf(" "); // empty space

} else if (board[row][col] == 1) {

printf("P"); // player

} else if (board[row][col] == 2) {

printf("%d", enemies\_defeated + 1); // enemy

} else {

printf("%c", board[row][col]); // math operator

}

printf(" ");

}

printf("\n");

}

}

// Function to generate a random number

int generate\_random\_number(int max\_number) {

srand(time(NULL));

return rand() % max\_number + 1;

}

// Function to generate a random math equation

void generate\_math\_equation(int \*num1, int \*num2, char \*operator) {

int random\_operator = rand() % 3; // 0 = +, 1 = -, 2 = \*

\*num1 = generate\_random\_number(10); // generate first random number between 1 and 10

\*num2 = generate\_random\_number(10); // generate second random number between 1 and 10

switch (random\_operator) {

case 0:

\*operator = '+';

break;

case 1:

\*operator = '-';

break;

case 2:

\*operator = '\*';

break;

}

}

// Function to get player input and update game state

void handle\_player\_input(int board[ROWS][COLS]) {

char input;

scanf(" %c", &input);

if (input == 'a') { // move left

if (player\_pos > 0) {

board[ROWS-1][player\_pos] = 0; // clear current player position

player\_pos--;

board[ROWS-1][player\_pos] = 1; // set new player position

}

} else if (input == 'd') { // move right

if (player\_pos < COLS-1) {

board[ROWS-1][player\_pos] = 0; // clear current player position

player\_pos++;

board[ROWS-1][player\_pos] = 1; // set new player position

}

} else if (input == '1' || input == '2' || input == '3') { // select math equation

int selected\_equation = input - '0'; // convert char input to int

int num1, num2;

char operator;

generate\_math\_equation(&num1, &num2, &operator); // generate new math equation

int result;

switch (operator) { // calculate result of math equation

case '+':

result = num1 + num2;

break;

case '-':

result = num1 - num2;

case '\*':

result = num1 \* num2;

break;

}

if (selected\_equation == (enemies\_defeated + 1) && result == selected\_equation) { // player selected correct equation

enemies\_defeated++;

board[ROWS-1][player\_pos] = 0; // clear current player position

player\_pos = 2; // set player position back to center

board[ROWS-1][player\_pos] = 1; // set new player position

if (enemies\_defeated == num\_enemies) {

printf("You win!\n");

exit(0); // exit program

}

} else { // player selected incorrect equation

printf("Incorrect equation selected. Try again.\n");

}

}

}

int main() {

int board[ROWS][COLS] = { // initialize game board

{0, 0, ' ', 0, 0},

{0, 0, ' ', 0, 0},

{0, 0, ' ', 0, 0},

{0, 0, ' ', 0, 0},

{0, 0, 1, 0, 0}

};

while (1) { // game loop

system("clear"); // clear console

print\_board(board);

handle\_player\_input(board);

}

}