TASK 1 Write a program that calculates the Fibonacci sequence.

//

// LAB ARRAYS TASK 1

// 24/11/20

// Sidi Liang

//

#include <stdio.h>

void generateFibonacci(int finbonacci[], int n);//The function to generate finbonacci

void displayFibonacci(int finbonacci[], int n);//Display the array

int main(){

int finbonacci[100] = {0, 1};//Declare the array

//n = 10

generateFibonacci(finbonacci, 10);

displayFibonacci(finbonacci, 10);

puts("");

//n = 20

generateFibonacci(finbonacci, 20);

displayFibonacci(finbonacci, 20);

puts("");

//n = 50, the number is too large for int to store

generateFibonacci(finbonacci, 50);

displayFibonacci(finbonacci, 50);

return 0;

}

void generateFibonacci(int finbonacci[], int n){

for(int i = 2; i <= n; i++){

finbonacci[i] = finbonacci[i - 2] + finbonacci[i - 1]; //Finbonacci

}

}

void displayFibonacci(int finbonacci[], int n){

for(int i = 0; i <= n; i++){

printf("%d ", finbonacci[i]);

}

puts("");

}

Code for TASK 1

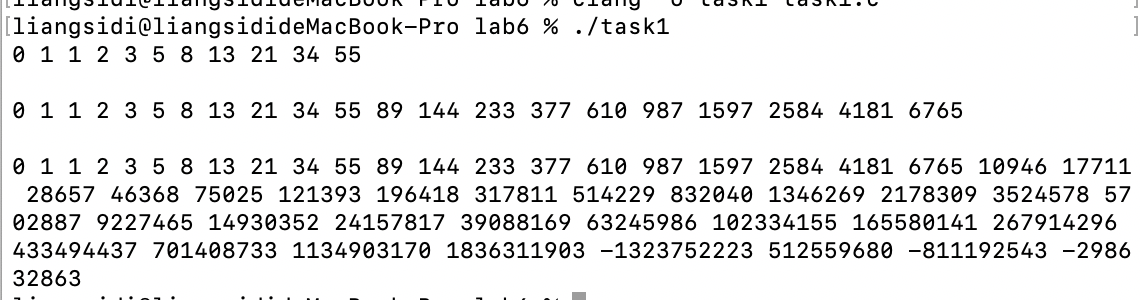


Figure 1: Output of Code in Task 1.

TASK 2 Write a program which creates a "game board".

//

// LAB ARRAYS TASK 2

// 24/11/20

// Sidi Liang

//

#include <stdio.h>

void drawBoard(int gameBoard[3][3]);//Draw the game board

int gameIsEnded(int gameBoard[3][3]);//Judge if the game is ended

int main(){

int gameBoard[3][3] = {0};

drawBoard(gameBoard);

while(!gameIsEnded(gameBoard)){

int userControl, \*gameBoardPt = &gameBoard[0][0];

scanf("%d", &userControl);//User input

gameBoardPt += userControl - 1;//Point the pointer to the place to change value

\*gameBoardPt = 1; //change the value

//remove board from the screen

for(int i = 0; i <= 2; i++){

puts("");

}

drawBoard(gameBoard);

}

return 0;

}

void drawBoard(int gameBoard[3][3]){

for(int i = 2; i >= 0; i--){

for(int j = 0; j <= 2; j++){

if(!gameBoard[i][j]) printf("%s ", ".");//If the value is 0, print .

else printf("%s ", "X");//If the value is 1, print X

}

puts("");

}

puts("");

}

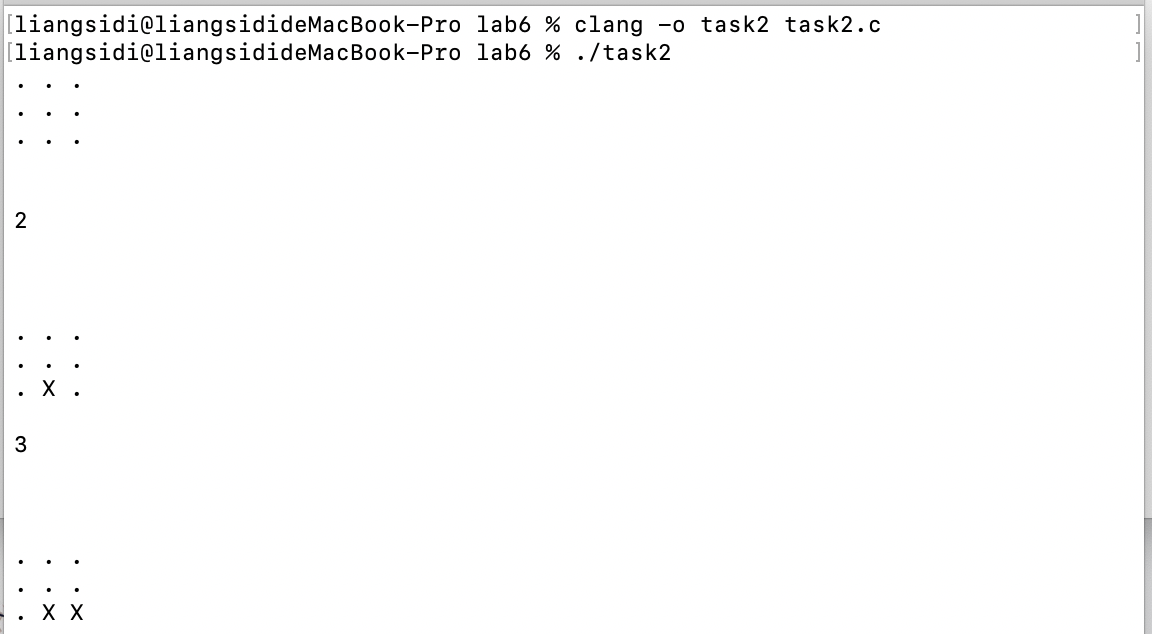


Figure 2: Output of Code in Task 2.

int gameIsEnded(int gameBoard[3][3]){

for(int i = 0; i <= 2; i++){

for(int j = 0; j <= 2; j++){

if(!gameBoard[i][j]) return 0; //If there's still dots in the borad, game is not ended

}

}

return 1;

}

Code for TASK 2

TASK 3 Write a tic-tac-toe game

//

// LAB ARRAYS TASK 3

// 24/11/20

// Sidi Liang

//

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

void drawBoard(int gameBoard[3][3]);//Draw the game board

int gameIsEnded(int gameBoard[3][3]);//Judge if the game is ended

int computerTurn(int gameBoard[3][3]);

int userTurn(int gameBoard[3][3]);

int judgeHorizontal(int gameBoard[3][3]);

int judgeVertical(int gameBoard[3][3]);

int judgeX(int gameBoard[3][3]);

int main(){

srand((unsigned)time(NULL)\*10);

int lastTurn = 2, gameBoard[3][3] = {0};

drawBoard(gameBoard);

while(!gameIsEnded(gameBoard)){

if(lastTurn == 1){

lastTurn = computerTurn(gameBoard);

}else if(lastTurn == 2){

lastTurn = userTurn(gameBoard);

}

//remove board from the screen

for(int i = 0; i <= 2; i++){

puts("");

}

drawBoard(gameBoard);

}

if(lastTurn == 1){

puts("Player won");

}else if(lastTurn == 2){

puts("Computer won");

}

return 0;

}

void drawBoard(int gameBoard[3][3]){

for(int i = 2; i >= 0; i--){

for(int j = 0; j <= 2; j++){

if(!gameBoard[i][j]) printf("%s ", ".");//If the value is 0, print .

else if(gameBoard[i][j] == 1) printf("%s ", "X");//If the value is 1, print X

else if(gameBoard[i][j] == 2) printf("%s ", "O");//If the value is 2, print O

}

puts("");

}

puts("");

}

int judgeHorizontal(int gameBoard[3][3]){

for(int i = 0; i <= 2; i++){

if(gameBoard[i][0] == gameBoard[i][1] && gameBoard[i][1] == gameBoard[i][2] && gameBoard[i][0] != 0) return 1; //Three same pieces horizontally

}

return 0;

}

int judgeVertical(int gameBoard[3][3]){

for(int i = 0; i <= 2; i++){

if(gameBoard[0][i] == gameBoard[1][i] && gameBoard[1][i] == gameBoard[2][i] && gameBoard[0][i] != 0) return 1; //Three same pieces vertically

}

return 0;

}

int judgeX(int gameBoard[3][3]){

if(gameBoard[0][0] == gameBoard[1][1] && gameBoard[1][1] == gameBoard[2][2] && gameBoard[0][0] != 0) return 1; //Three same pieces in diagonal line

if(gameBoard[0][2] == gameBoard[1][1] && gameBoard[1][1] == gameBoard[2][0] && gameBoard[0][2] != 0) return 1; //Three same pieces in diagonal line

return 0;

}

int gameIsEnded(int gameBoard[3][3]){

if(judgeX(gameBoard) || judgeVertical(gameBoard) || judgeHorizontal(gameBoard)) return 1; //Game ended

for(int i = 0; i <= 2; i++){

for(int j = 0; j <= 2; j++){

if(!gameBoard[i][j]) return 0; //If there's still dots in the borad, game is not ended

}

}

return 1;

}

int userTurn(int gameBoard[3][3]){

int userControl, \*gameBoardPt = &gameBoard[0][0];;

scanf("%d", &userControl);//User input

gameBoardPt += userControl - 1;//Point the pointer to the place to change value

if(\*gameBoardPt != 0){

puts("ilegal input!");

return 2;

}

\*gameBoardPt = 1; //change the value

return 1;

}

int computerTurn(int gameBoard[3][3]){

int computerControl, \*gameBoardPt = &gameBoard[0][0];

do{

gameBoardPt = &gameBoard[0][0];

computerControl = rand() % 9;

gameBoardPt += computerControl;//Point the pointer to the place to change value

}while(\*gameBoardPt != 0);//Only put piece in empty slots

\*gameBoardPt = 2; //change the value

return 2;

}

Code for TASK 3

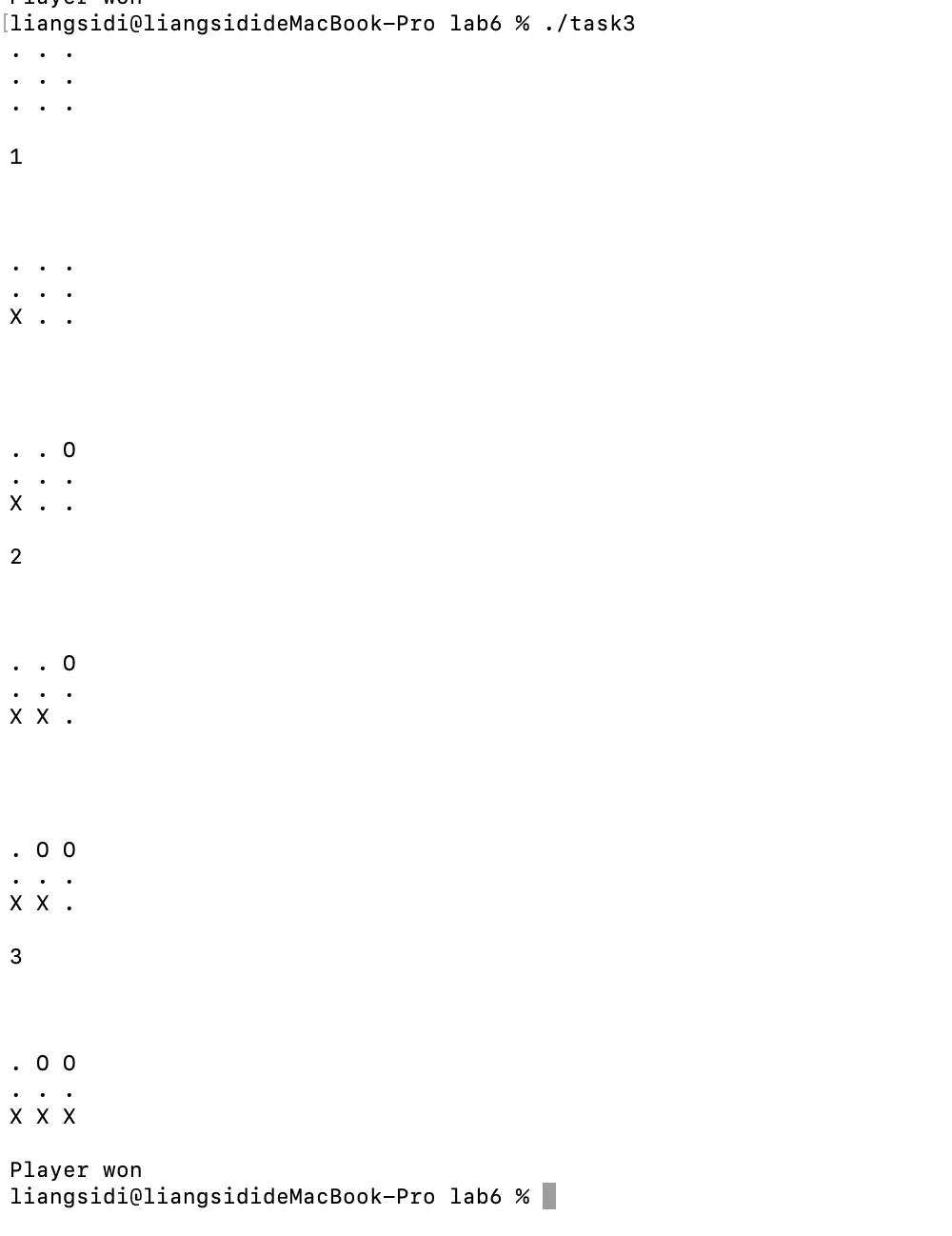


Figure 3: Output of Code in Task 3.