/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This program is a simple tic-tac-toe game.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* PROTOTYPES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

char\*\* createboard();

void print(char\*\* Board);

int isdraw(char\*\* Board);

char winningmove(char\*\* Board, int i, int j);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* MAIN FUNCTION

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main() {

char\*\* Board = createboard();

char winner = '\0';

char row;

char col;

char turn = 'X';

while(!winner && !isdraw(Board)) {

print(Board);

// Read Move

printf("Player %c, make your move: ", turn);

fflush(stdout);

scanf(" %c %c", &row, &col);

// Make move if square is free

int rowind = row - 'A';

int colind = col - '1';

if (Board[rowind][colind] == ' ') {

Board[rowind][colind] = turn;

if (turn == 'X') {

turn = 'O';

} else {

turn = 'X';

}

winner = winningmove(Board, rowind, colind);

} else {

printf("Square occupied; try again.\n");

}

}

// Game over - print board & declare finish

print(Board);

if (winner == 'X' || winner == 'O') {

printf("Congratulations %c!\n", winner);

} else {

printf("Game ends in a draw.\n");

}

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* FUNCTION DEFINITIONS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Creates the board with all squares init to ' '

char\*\* createboard() {

char\*\* B = calloc(3, sizeof(char\*));

for(int i = 0; i < 3; ++i) {

B[i] = calloc(3, sizeof(char));

}

for(int j=0; j < 3; ++j) {

for(int k=0; k < 3; ++k) {

B[j][k] = ' ';

}

}

return B;

}

// Prints the board

void print(char\*\* Board) {

printf(" |1|2|3|\n");

for(int i = 0; i < 3; ++i) {

printf("%c|", 'A' + i);

for(int j = 0; j < 3; ++j) {

printf("%c|", Board[i][j]);

}

printf("\n");

}

}

// Returns true if the game is a draw

int isdraw(char\*\* Board) {

for(int i = 0; i < 3; ++i) {

for(int j = 0; j < 3; ++j) {

if (Board[i][j] == ' ') {

// empty square, so game ain't over yet

return 0;

}

}

}

// no empty squares, so it's a draw

return 1;

}

// Returns 'X' if (i,j) was a winning move for X

// Returns 'Y' if (i,j) was a winning move for Y

// Retruns ASCII value 0 otherwise

char winningmove(char\*\* Board, int i, int j) {

if (Board[i][j] == Board[i][(j+1)%3]

&& Board[i][j] == Board[i][(j+2)%3])

{

// got a column

return Board[i][j];

}

else if (Board[i][j] == Board[(i+1)%3][j]

&& Board[i][j] == Board[(i+2)%3][j])

{

// got a row

return Board[i][j];

}

else if (i == j && Board[i][j] == Board[(i+1)%3][(j+1)%3]

&& Board[i][j] == Board[(i+2)%3][(j+2)%3])

{

// got the forward diagonal

return Board[i][j];

}

else if (i+j == 2 && Board[i][j] == Board[(i+2)%3][(j+1)%3]

&& Board[i][j] == Board[(i+1)%3][(j+2)%3])

{

// got the reverse diagonal

return Board[i][j];

}

else {

// got nothing

return 0;

}

}