#include <iostream>

#include <conio.h>

#include "Header.h"

using namespace std;

void check\_poz\_pawn(char filed[][12], int turn\_of\_the\_more, int& poz\_hor, int& poz\_ver, char white, char black, char queen\_white, char queen\_black, int AIturn)//проверка позиции пешки

{

bool exit = false;

if (turn\_of\_the\_more % 2 == 0)//Проверка правильности вввода позиции пешки в зависимости от очереди хода, с проверкой не допустить хода назад

{

while (exit == false)

{

if (filed[poz\_hor][poz\_ver] == white)//если это белая пешка

{

if (check\_poz\_pavn\_white\_or\_black(filed, poz\_ver, poz\_hor, white, queen\_white, black, queen\_black, 1, 2))//Ошибка при вводе позиции у белых

{

pawn\_pos\_inp\_error(poz\_ver, poz\_hor, AIturn);

}

else

{

exit = true;

}

}

else if (filed[poz\_hor][poz\_ver] == queen\_white)

{

if (check\_poz\_queen\_white\_or\_black(filed, poz\_ver, poz\_hor, white, queen\_white, black, queen\_black, 1, 2))//Ошибка при вводе позиции у белых

{

pawn\_pos\_inp\_error(poz\_ver, poz\_hor, AIturn);

}

else

{

exit = true;

}

}

else

{

pawn\_pos\_inp\_error(poz\_ver, poz\_hor, AIturn);

}

}

}

else

{

while (exit == false)

{

if (filed[poz\_hor][poz\_ver] == black)//если это черная пешка

{

if (check\_poz\_pavn\_white\_or\_black(filed, poz\_ver, poz\_hor, black, queen\_black, white, queen\_white, -1, -2))//Ошибка при вводе позиции у чёрных

{

pawn\_pos\_inp\_error(poz\_ver, poz\_hor, AIturn);

}

else

{

exit = true;

}

}

else if (filed[poz\_hor][poz\_ver] == queen\_black)//если это чёрная дамка

{

if (check\_poz\_queen\_white\_or\_black(filed, poz\_ver, poz\_hor, black, queen\_black, white, queen\_white, -1, -2))//Ошибка при вводе позиции у чёрных

{

pawn\_pos\_inp\_error(poz\_ver, poz\_hor, AIturn);

}

else

{

exit = true;

}

}

else

{

pawn\_pos\_inp\_error(poz\_ver, poz\_hor, AIturn);

}

}

}

}

void check\_the\_ability\_to\_beat\_again(char field[][12], int& new\_poz\_hor, int& new\_poz\_ver, int& poz\_hor, int& poz\_ver, int size, int turn\_of\_the\_more, int& col\_vo\_white, int& col\_vo\_black, int AIturn, char evil\_pawn, char evil\_queen)//может ли пешка бить еще раз

{

poz\_ver = new\_poz\_ver;

poz\_hor = new\_poz\_hor;

while (turn\_of\_the\_more % 2 == 0 ? pawn\_black\_and\_white\_beating(field, poz\_ver, poz\_hor, 'B', 'D') : pawn\_black\_and\_white\_beating(field, poz\_ver, poz\_hor, 'A', 'C'))//цикл проверки может ли пешка побить ещё раз, заканчивается если возможности нет

{

system("cls");

filed\_out\_durig\_the\_game(field, size);

if (AIturn == 1)

{

if ((field[new\_poz\_hor - 1][new\_poz\_ver - 1] == 'A' || field[new\_poz\_hor - 1][new\_poz\_ver - 1] == 'C') && field[new\_poz\_hor - 2][new\_poz\_ver - 2] == ' ')

{

new\_poz\_ver = new\_poz\_ver - 1;

new\_poz\_hor = new\_poz\_hor - 1;

}

else if ((field[new\_poz\_hor + 1][new\_poz\_ver - 1] == 'A' || field[new\_poz\_hor + 1][new\_poz\_ver - 1] == 'C') && field[new\_poz\_hor + 2][new\_poz\_ver - 2] == ' ')

{

new\_poz\_ver = new\_poz\_ver - 1;

new\_poz\_hor = new\_poz\_hor + 1;

}

else if ((field[new\_poz\_hor + 1][new\_poz\_ver + 1] == 'A' || field[new\_poz\_hor + 1][new\_poz\_ver + 1] == 'C') && field[new\_poz\_hor + 2][new\_poz\_ver + 2] == ' ')

{

new\_poz\_ver = new\_poz\_ver + 1;

new\_poz\_hor = new\_poz\_hor + 1;

}

else if ((field[new\_poz\_hor - 1][new\_poz\_ver + 1] == 'A' || field[new\_poz\_hor - 1][new\_poz\_ver + 1] == 'C') && field[new\_poz\_hor - 2][new\_poz\_ver + 2] == ' ')

{

new\_poz\_ver = new\_poz\_ver + 1;

new\_poz\_hor = new\_poz\_hor - 1;

}

}

else

{

spaceFIELD();

cout << "У пешки есть возможность побить, бейте\n";

cin >> new\_poz\_ver >> new\_poz\_hor;

new\_poz\_ver++;

new\_poz\_hor++;

}

while (turn\_of\_the\_more % 2 == 0 ? pawn\_black\_and\_white\_beating(field, poz\_ver, poz\_hor, 'B', 'D') : pawn\_black\_and\_white\_beating(field, poz\_ver, poz\_hor, 'A', 'C'))

{

if (new\_poz\_ver - poz\_ver == 1 && new\_poz\_hor - poz\_hor == 1)//если вражеская пешка выше по горизонтали и вертикали

{

if ((field[new\_poz\_hor][new\_poz\_ver] == evil\_pawn || field[new\_poz\_hor][new\_poz\_ver] == evil\_queen) && field[new\_poz\_hor + 1][new\_poz\_ver + 1] == ' ' && new\_poz\_hor + 1 <= 9 && new\_poz\_hor + 1 <= 9)//проверка есть ли за пешкой другая пешка

{

process\_of\_eating\_pawn(field, poz\_hor, poz\_ver, new\_poz\_hor, new\_poz\_ver, turn\_of\_the\_more, col\_vo\_black, col\_vo\_white, 1, 1);

break;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else if (new\_poz\_ver - poz\_ver == -1 && new\_poz\_hor - poz\_hor == 1)//если вражеская пешка ниже по вертикали, но выше по горизонтали

{

if ((field[new\_poz\_hor][new\_poz\_ver] == evil\_pawn || field[new\_poz\_hor][new\_poz\_ver] == evil\_queen) && field[new\_poz\_hor + 1][new\_poz\_ver - 1] == ' ' && new\_poz\_ver + 1 <= 9 && new\_poz\_hor - 1 >= 2)//проверка есть ли за пешкой другая пешка

{

process\_of\_eating\_pawn(field, poz\_hor, poz\_ver, new\_poz\_hor, new\_poz\_ver, turn\_of\_the\_more, col\_vo\_black, col\_vo\_white, 1, -1);

break;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else if (new\_poz\_ver - poz\_ver == 1 && new\_poz\_hor - poz\_hor == -1)//если вражеская пешка выше по вертикали, но ниже по горизонтали

{

if ((field[new\_poz\_hor][new\_poz\_ver] == evil\_pawn || field[new\_poz\_hor][new\_poz\_ver] == evil\_queen) && field[new\_poz\_hor - 1][new\_poz\_ver + 1] == ' ' && new\_poz\_hor - 1 >= 2 && new\_poz\_ver + 1 <= 9)//проверка есть ли за пешкой другая пешка

{

process\_of\_eating\_pawn(field, poz\_hor, poz\_ver, new\_poz\_hor, new\_poz\_ver, turn\_of\_the\_more, col\_vo\_black, col\_vo\_white, -1, 1);

break;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else if (new\_poz\_ver - poz\_ver == -1 && new\_poz\_hor - poz\_hor == -1)//если вражеская пешка ниже по горизонтали и вертикали

{

if ((field[new\_poz\_hor][new\_poz\_ver] == evil\_pawn || field[new\_poz\_hor][new\_poz\_ver] == evil\_queen) && field[new\_poz\_hor - 1][new\_poz\_ver - 1] == ' ' && new\_poz\_ver - 1 >= 2 && new\_poz\_ver - 1 >= 2)//проверка есть ли за пешкой другая пешка

{

process\_of\_eating\_pawn(field, poz\_hor, poz\_ver, new\_poz\_hor, new\_poz\_ver, turn\_of\_the\_more, col\_vo\_black, col\_vo\_white, -1, -1);

break;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

}

}

void process\_of\_eating\_pawn(char field[][12], int& hor, int& ver, int& new\_hor, int& new\_ver, int turn\_of\_the\_more, int& col\_vo\_black, int& col\_vo\_white, int i, int j)//процесс бития пешки

{

field[new\_hor + i][new\_ver + j] = field[hor][ver];

field[new\_hor][new\_ver] = ' ';

field[hor][ver] = ' ';

who\_needs\_to\_take\_away\_figure(turn\_of\_the\_more, col\_vo\_black, col\_vo\_white);

new\_hor += i;

new\_ver += j;

ver = new\_ver;

hor = new\_hor;

}

void check\_a\_new\_position(bool& exit\_the\_str\_cucle, int& new\_poz\_ver, int& new\_poz\_hor, int& poz\_ver, int& poz\_hor, char field[][12], char black, char queen\_black, char white, char queen\_white, int AIturn)//проверка новой позиции

{

while (exit\_the\_str\_cucle == false)

{

if (Check\_the\_req\_cel(new\_poz\_ver, new\_poz\_hor) || (new\_poz\_hor <= 9 && new\_poz\_hor >= 2 && new\_poz\_ver <= 9 && new\_poz\_ver >= 2))

{

if (field[poz\_hor][poz\_ver] == black || field[poz\_hor][poz\_ver] == queen\_black)//Проверка правильности вввода новой позиции пешки в зависимости от очереди хода, с проверкой не допустить хода назад

{

if (field[poz\_hor][poz\_ver] == black)//Если это черная пешка

{

if (new\_poz\_ver > poz\_ver)

{

if (new\_poz\_hor < poz\_hor)

{

if (new\_poz\_hor - poz\_hor == -1 && new\_poz\_ver - poz\_ver == 1)//Ошибка при вводе позиции у чёрных

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

if (field[new\_poz\_hor][new\_poz\_ver] == white || field[new\_poz\_hor][new\_poz\_ver] == queen\_white)

{

if (new\_poz\_ver - poz\_ver == 1 && new\_poz\_hor - poz\_hor == 1)//Ошибка при вводе позиции у чёрных

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

}

else

{

if (new\_poz\_hor < poz\_hor)

{

if (new\_poz\_hor - poz\_hor == -1 && new\_poz\_ver - poz\_ver == -1)//Ошибка при вводе позиции у чёрных

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

if (field[new\_poz\_hor][new\_poz\_ver] == white || field[new\_poz\_hor][new\_poz\_ver] == queen\_white)

{

if (new\_poz\_ver - poz\_ver == -1 && new\_poz\_hor - poz\_hor == +1)//Ошибка при вводе позиции у чёрных

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

}

}

else if (field[poz\_hor][poz\_ver] == queen\_black)//Если это чёрная дамка

{

the\_queens\_move( new\_poz\_ver, new\_poz\_hor, poz\_ver, poz\_hor, exit\_the\_str\_cucle, field, AIturn);

}

}

else if (field[poz\_hor][poz\_ver] == white || field[poz\_hor][poz\_ver] == queen\_white)

{

if (field[poz\_hor][poz\_ver] == white)//Если это белая пешка

{

if (new\_poz\_ver > poz\_ver)

{

if (new\_poz\_hor > poz\_hor)

{

if (new\_poz\_hor - poz\_hor == 1 && new\_poz\_ver - poz\_ver == 1)//Ошибка при вводе позиции у белых

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

if (field[new\_poz\_hor][new\_poz\_ver] == black || field[new\_poz\_hor][new\_poz\_ver] == queen\_black)

{

if (new\_poz\_hor - poz\_hor == -1 && new\_poz\_ver - poz\_ver == 1)//Ошибка при вводе позиции у белых

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

}

else

{

if (new\_poz\_hor > poz\_hor)

{

if (new\_poz\_hor - poz\_hor == 1 && new\_poz\_ver - poz\_ver == -1)//Ошибка при вводе позиции у белых

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

if (field[new\_poz\_hor][new\_poz\_ver] == black || field[new\_poz\_hor][new\_poz\_ver] == queen\_black)

{

if (new\_poz\_hor - poz\_hor == -1 && new\_poz\_ver - poz\_ver == -1)//Ошибка при вводе позиции у белых

{

exit\_the\_str\_cucle = true;

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

}

}

else if (field[poz\_hor][poz\_ver] == queen\_white)//Если это белая дамка

{

the\_queens\_move( new\_poz\_ver, new\_poz\_hor, poz\_ver, poz\_hor, exit\_the\_str\_cucle, field, AIturn);

}

}

}

else

{

pawn\_pos\_inp\_error(new\_poz\_ver, new\_poz\_hor, AIturn);

}

}

exit\_the\_str\_cucle = false;

}