#include <iostream>

#include <conio.h>

#include "Header.h"

using namespace std;

void all\_game(char field[][12], char white, char black, int size, int queen\_white, int queen\_black, int color)//Процесс игры

{

int new\_poz\_hor = 0, new\_poz\_ver = 0, poz\_hor = 0, poz\_ver = 0, col\_vo\_white = 12, col\_vo\_blakc = 12, turn\_of\_the\_more = 0, AIexort = 0, AIturn = 0;

bool exit\_the\_str\_cucle = false;

int pawn\_beat\_hor[12], pawn\_beat\_ver[12], col\_vo\_voz\_beat = 0, draw = 0;

//new\_poz\_hor - новая позиция пешки по горизонтали

//new\_poz\_ver - новая позиция пешки по вертикали

//poz\_hor - начальная позиция пешки по горизонтали

//poz\_ver - начальная позиция пешки по вертикали

//col\_vo\_white - количество белых фигур

//col\_vo\_blakc - количество чёрных фигур

//furn\_of\_the\_more - очередь хода

//exit\_the\_str\_cucle - выход из циклов проверки

//pawn\_beat\_hor - позиция пешки по горизонтали, которая может бить

//pawn\_beat\_ver - позиция пешки по вертикали, которая может бить

//col\_vo\_voz\_beat - количество пешек, которые могут бить

//AIexort - бот атакует

//AIturn - ход бота

//draw - ничья, если количество фигур, которые не могут походить равно количеству фигур на поле

do

{

if (turn\_of\_the\_more % 2 == 0)//Подсказка очереди хода

{

cout << "\n";

spaceFIELD();

cout << "Ходять белые";

if (color == 1)

{

AIturn = 0;

}

else if (color == 2)

{

AIturn = 1;

}

}

else

{

cout << "\n";

spaceFIELD();

cout << "Ходят чёрные";

if (color == 1)

{

AIturn = 1;

}

else if (color == 2)

{

AIturn = 0;

}

}

for (int i = 0; i < size; i++)//Проверка есть ли пешки способные бить

{

for (int j = 0; j < size; j++)

{

if (turn\_of\_the\_more % 2 == 0)

{

if (field[i][j] == white)

{

if (pawn\_black\_and\_white\_beating(field, j, i, black, queen\_black) && color == 2)//AI поиск цели

{

pawn\_beat\_hor[col\_vo\_voz\_beat] = i;

pawn\_beat\_ver[col\_vo\_voz\_beat] = j;

poz\_hor = pawn\_beat\_hor[col\_vo\_voz\_beat];

poz\_ver = pawn\_beat\_ver[col\_vo\_voz\_beat];

col\_vo\_voz\_beat++;

if ((field[i - 1][j - 1] == 'B' || field[i - 1][j - 1] == 'D') && field[i - 2][j - 2] == ' ')

{

new\_poz\_ver = j - 1;

new\_poz\_hor = i - 1;

}

else if ((field[i + 1][j - 1] == 'B' || field[i + 1][j - 1] == 'D') && field[i + 2][j - 2] == ' ')

{

new\_poz\_ver = j - 1;

new\_poz\_hor = i + 1;

}

else if ((field[i + 1][j + 1] == 'B' || field[i + 1][j + 1] == 'D') && field[i + 2][j + 2] == ' ')

{

new\_poz\_ver = j + 1;

new\_poz\_hor = i + 1;

}

else if ((field[i - 1][j + 1] == 'B' || field[i - 1][j + 1] == 'D') && field[i - 2][j + 2] == ' ')

{

new\_poz\_ver = j + 1;

new\_poz\_hor = i - 1;

}

AIexort = 1;

}

else if (pawn\_black\_and\_white\_beating(field, j, i, black, queen\_black) && AIturn != 1)

{

cout << "\n";

spaceFIELD();

cout << "Белая пешка на позиции " << j - 1 << " ; " << i - 1 << " может бить\n";

pawn\_beat\_hor[col\_vo\_voz\_beat] = i;

pawn\_beat\_ver[col\_vo\_voz\_beat] = j;

col\_vo\_voz\_beat++;

}

if (check\_poz\_pavn\_white\_or\_black(field, i, j, white, queen\_white, black, queen\_black, 1, 2) )

{

draw++;

}

}

else if (field[i][j] == queen\_white)

{

can\_a\_queen\_beat(i, j, size, field, white, queen\_white, black, queen\_black, pawn\_beat\_hor, pawn\_beat\_ver, col\_vo\_voz\_beat);

if (check\_poz\_queen\_white\_or\_black(field, i, j, white, queen\_white, black, queen\_black, 1, 2))

{

draw++;

}

}

}

else

{

if (field[i][j] == black)

{

if (pawn\_black\_and\_white\_beating(field, j, i, white, queen\_white) && color == 1)//AI поиск цели

{

pawn\_beat\_hor[col\_vo\_voz\_beat] = i;

pawn\_beat\_ver[col\_vo\_voz\_beat] = j;

poz\_hor = pawn\_beat\_hor[col\_vo\_voz\_beat];

poz\_ver = pawn\_beat\_ver[col\_vo\_voz\_beat];

col\_vo\_voz\_beat++;

if ((field[i - 1][j - 1] == 'A' || field[i - 1][j - 1] == 'C') && field[i - 2][j - 2] == ' ')

{

new\_poz\_ver = j - 1;

new\_poz\_hor = i - 1;

}

else if ((field[i + 1][j - 1] == 'A' || field[i + 1][j - 1] == 'C') && field[i + 2][j - 2] == ' ')

{

new\_poz\_ver = j - 1;

new\_poz\_hor = i + 1;

}

else if ((field[i + 1][j + 1] == 'A' || field[i + 1][j + 1] == 'C') && field[i + 2][j + 2] == ' ')

{

new\_poz\_ver = j + 1;

new\_poz\_hor = i + 1;

}

else if ((field[i - 1][j + 1] == 'A' || field[i - 1][j + 1] == 'C') && field[i - 2][j + 2] == ' ')

{

new\_poz\_ver = j + 1;

new\_poz\_hor = i - 1;

}

AIexort = 1;

}

else if (pawn\_black\_and\_white\_beating(field, j, i, white, queen\_white) && AIturn != 1)

{

cout << "\n";

spaceFIELD();

cout << "Чёрная пешка на позиции " << j - 1 << " ; " << i - 1 << " может бить\n";

pawn\_beat\_hor[col\_vo\_voz\_beat] = i;

pawn\_beat\_ver[col\_vo\_voz\_beat] = j;

col\_vo\_voz\_beat++;

}

if (check\_poz\_pavn\_white\_or\_black(field, i, j, black, queen\_black, white, queen\_white, 1, 2))

{

draw++;

}

}

else if (field[i][j] == queen\_black)

{

can\_a\_queen\_beat(i, j, size, field, black, queen\_black, white, queen\_white, pawn\_beat\_hor, pawn\_beat\_ver, col\_vo\_voz\_beat);

if (check\_poz\_queen\_white\_or\_black(field, i, j, black, queen\_black, white, queen\_white, 1, 2))

{

draw++;

}

}

}

}

}

if (turn\_of\_the\_more % 2 == 0 ? draw == col\_vo\_white : draw == col\_vo\_blakc)

{

exit\_the\_str\_cucle = true;

break;

}

draw = 0;

if (AIexort == 0 && AIturn == 1 && (color == 1 || color == 2))//Выбор шашки AI

{

poz\_hor = rand() % 10;

poz\_ver = rand() % 10;

}

else if (AIturn == 0)

{

cout << "\n";

spaceFIELD();

cout << "Сделайте ход\n";

spaceFIELD();

cout << "Для этого сперва напишите координаты пешки, которую хотите передвинуть, а потом куда её передвинуть\n";

spaceFIELD();

cout << "Выберите пешку сперва написав её позицию по вертикали, а потом по горизонтали\n";

cin >> poz\_ver >> poz\_hor;//Ввод позиции пешки

poz\_hor++;//увеличение на 1 для того, чтобы правильно определить позицию пешки

poz\_ver++;

}

if (col\_vo\_voz\_beat == 0)//проверка позиции если нет пешек способных бить

{

check\_poz\_pawn(field, turn\_of\_the\_more, poz\_hor, poz\_ver, white, black, queen\_white, queen\_black, AIturn);

}

else//если есть пешки способные бить

{

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

{

for (int i = 0; i < col\_vo\_voz\_beat; i++)

{

if (poz\_ver == pawn\_beat\_ver[i] && poz\_hor == pawn\_beat\_hor[i])

{

check\_poz\_pawn(field, turn\_of\_the\_more, poz\_hor, poz\_ver, white, black, queen\_white, queen\_black, AIturn);

exit\_the\_str\_cucle = true;

break;

}

}

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

{

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

}

}

}

col\_vo\_voz\_beat = 0;

if (AIexort == 0 && AIturn == 1 && (color == 1 || color == 2))//Выбор шашки AI

{

new\_poz\_hor = rand() % 10;

new\_poz\_ver = rand() % 10;

}

else if (AIturn == 0)

{

cout << "\n";

spaceFIELD();

cout << "Введите координаты куда вы её хотите передвинуть (сперва вертикаль, потом горизонталь)\n";

cin >> new\_poz\_ver >> new\_poz\_hor;//Ввод новой позиции пешки

new\_poz\_hor++;//увеличение на 1 для того, чтобы правильно определить новую позицию пешки

new\_poz\_ver++;

}

if (exit\_the\_str\_cucle != false)

{

if (turn\_of\_the\_more % 2 == 0 ? field[new\_poz\_hor][new\_poz\_ver] == black || field[new\_poz\_hor][new\_poz\_ver] == queen\_black : field[new\_poz\_hor][new\_poz\_ver] == white || field[new\_poz\_hor][new\_poz\_ver] == queen\_white)

{

check\_a\_new\_position(exit\_the\_str\_cucle, new\_poz\_ver, new\_poz\_hor, poz\_ver, poz\_hor, field, black, queen\_black, white, queen\_white, AIturn);//проверка новой позиции

}

else

{

exit\_the\_str\_cucle = false;

if (turn\_of\_the\_more % 2 == 0)

{

while (field[new\_poz\_hor][new\_poz\_ver] != black && field[new\_poz\_hor][new\_poz\_ver] != queen\_black)

{

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

check\_a\_new\_position(exit\_the\_str\_cucle, new\_poz\_ver, new\_poz\_hor, poz\_ver, poz\_hor, field, black, queen\_black, white, queen\_white, AIturn);//проверка новой позиции

}

}

else

{

while (field[new\_poz\_hor][new\_poz\_ver] != white && field[new\_poz\_hor][new\_poz\_ver] != queen\_white)

{

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

check\_a\_new\_position(exit\_the\_str\_cucle, new\_poz\_ver, new\_poz\_hor, poz\_ver, poz\_hor, field, black, queen\_black, white, queen\_white, AIturn);//проверка новой позици

}

}

}

}

exit\_the\_str\_cucle = false;

while (exit\_the\_str\_cucle == false)//ход пешки

{

check\_a\_new\_position(exit\_the\_str\_cucle, new\_poz\_ver, new\_poz\_hor, poz\_ver, poz\_hor, field, black, queen\_black, white, queen\_white, AIturn);//проверка новой позиции

if (field[new\_poz\_hor][new\_poz\_ver] == ' ')//Перемещение пешки если на новой позиции нет других пешек

{

field[new\_poz\_hor][new\_poz\_ver] = field[poz\_hor][poz\_ver];

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

exit\_the\_str\_cucle = true;

}

else//Перемещение пешки если на новой позиции есть другая пешка

{

if (turn\_of\_the\_more % 2 == 0 && field[new\_poz\_hor][new\_poz\_ver] == black || field[new\_poz\_hor][new\_poz\_ver] == queen\_black)//если ходили белые и бить надо чёрных

{

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

{

if (new\_poz\_ver > poz\_ver)//определяет в какую сторону по вертикали надо бить

{

if (new\_poz\_hor > poz\_hor)//определяет в какую сторону по горизонтали надо бить

{

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

{

field[new\_poz\_hor + 1][new\_poz\_ver + 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_blakc--;

new\_poz\_hor++;

new\_poz\_ver++;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, black, queen\_black);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

else

{

if (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

{

field[new\_poz\_hor - 1][new\_poz\_ver + 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_blakc--;

new\_poz\_hor--;

new\_poz\_ver++;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, black, queen\_black);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

}

else

{

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

{

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

{

field[new\_poz\_hor + 1][new\_poz\_ver - 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_blakc--;

new\_poz\_hor++;

new\_poz\_ver--;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, black, queen\_black);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

else

{

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

{

field[new\_poz\_hor - 1][new\_poz\_ver - 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_blakc--;

new\_poz\_hor--;

new\_poz\_ver--;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, black, queen\_black);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

}

}

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

{

if (new\_poz\_ver > poz\_ver)//определяет в какую сторону по вертикали надо бить

{

if (new\_poz\_hor > poz\_hor)//определяет в какую сторону по горизонтали надо бить

{

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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, 1, 1);

}

else

{

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

}

}

else

{

if (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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, -1, 1);

}

else

{

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

}

}

}

else

{

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

{

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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, 1, -1);

}

else

{

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

}

}

else

{

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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, -1, -1);

}

else

{

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

}

}

}

}

}

else if (turn\_of\_the\_more % 2 != 0 && (field[new\_poz\_hor][new\_poz\_ver] == white || field[new\_poz\_hor][new\_poz\_ver] == queen\_white))//если ходили чёрные и бить надо белых

{

if (field[poz\_hor][poz\_ver] == black)

{

if (new\_poz\_ver > poz\_ver)//определяет в какую сторону по вертикали надо бить

{

if (new\_poz\_hor < poz\_hor)//определяет в какую сторону по горизонтали надо бить

{

if (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

{

field[new\_poz\_hor - 1][new\_poz\_ver + 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_white--;

new\_poz\_hor--;

new\_poz\_ver++;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, white, queen\_white);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

else

{

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

{

field[new\_poz\_hor + 1][new\_poz\_ver + 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_white--;

new\_poz\_hor++;

new\_poz\_ver++;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, white, queen\_white);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

}

else

{

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

{

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

{

field[new\_poz\_hor - 1][new\_poz\_ver - 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_white--;

new\_poz\_hor--;

new\_poz\_ver--;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, white, queen\_white);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

else

{

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

{

field[new\_poz\_hor + 1][new\_poz\_ver - 1] = field[poz\_hor][poz\_ver];

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

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

col\_vo\_white--;

new\_poz\_hor++;

new\_poz\_ver--;

check\_the\_ability\_to\_beat\_again(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, size, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, AIturn, white, queen\_white);

exit\_the\_str\_cucle = true;

}

else

{

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

}

}

}

}

else if (field[poz\_hor][poz\_ver] == queen\_black)

{

if (new\_poz\_ver > poz\_ver)//определяет в какую сторону по вертикали надо бить

{

if (new\_poz\_hor > poz\_hor)//определяет в какую сторону по горизонтали надо бить

{

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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, 1, 1);

}

else

{

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

}

}

else

{

if (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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, -1, 1);

}

else

{

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

}

}

}

else

{

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

{

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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, 1, -1);

}

else

{

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

}

}

else

{

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

{

queen\_beat(field, new\_poz\_hor, new\_poz\_ver, poz\_hor, poz\_ver, turn\_of\_the\_more, col\_vo\_white, col\_vo\_blakc, exit\_the\_str\_cucle, AIturn, -1, -1);

}

else

{

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

}

}

}

}

}

else

{

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

}

}

}

turning\_a\_pawn(new\_poz\_hor, new\_poz\_ver, field, queen\_white, queen\_black, turn\_of\_the\_more);

turn\_of\_the\_more++;//следующий ход

exit\_the\_str\_cucle = false;

system("cls");//Очистка поля после хода

filed\_out\_durig\_the\_game(field, size);//Вывод поля после хода игрока

cout << "\n";

spaceFIELD();

cout << "Осталось " << col\_vo\_white << " белых фигур и " << col\_vo\_blakc << " чёрных фигур\n";

AIexort = 0;

} while (col\_vo\_blakc != 0 && col\_vo\_white != 0);

if (col\_vo\_blakc == 0)//Объявление результата раунда

{

spaceFIELD();

cout << "\n\nПобедили белые\n";

}

else if (col\_vo\_white == 0)

{

spaceFIELD();

cout << "\n\nПобедили чёрные\n";

}

else if (exit\_the\_str\_cucle == true)

{

spaceFIELD();

cout << "\n\nНичья\n";

}

}

void who\_needs\_to\_take\_away\_figure(int turn\_of\_the\_more, int& col\_vo\_black, int& col\_vo\_white)//кому отнаять фигуру

{

if (turn\_of\_the\_more % 2 == 0)

{

col\_vo\_black--;

}

else

{

col\_vo\_white--;

}

}

void pawn\_pos\_inp\_error(int& ver, int& hor, int AIturn)//При вводе позиции пешки допущена ошибка

{

if (AIturn == 1)

{

hor = rand() % 8 + 2;

ver = rand() % 8 + 2;

//cout << ver - 1 << " ";

//cout << hor - 1 << "\n";

}

else

{

spaceFIELD();

cout << "Вы ввели неверные координаты, введите их ещё раз\n";

cin >> ver >> hor;

hor++;

ver++;

}

}

void turning\_a\_pawn(int hor, int ver, char field[][12], char queen\_white, char queen\_black, int turn\_of\_the\_more)//Превращение пешки в дамку если она дошла до границы

{

if (hor == 9 && turn\_of\_the\_more % 2 == 0)

{

field[hor][ver] = queen\_white;

}

else if (hor == 2 && turn\_of\_the\_more % 2 != 0)

{

field[hor][ver] = queen\_black;

}

}