ФГБОУ ВО «Ивановский государственный энергетический университет
имени В.И. Ленина»
Факультет информатики и вычислительной техники
Кафедра программного обеспечения компьютерных систем

Курсовая работа

на тему

Уголки

Выполнил:
студент группы 1-41 Иванов Д.С. _____ _____
(подпись) (дата)

Руководитель:
доцент каф. ПОКС Алыкова А.Л. _____ _____
(подпись) (дата)

Иваново 2021

Задание

Написать программу, реализующую игру в уголки. Данная программа должна обеспечивать:

- Диалоговый режим работы, диалог инициируется компьютером;
- Отображение игрового поля на экране;
- Выбор в игровом поле шашки и перемещение ее с помощью функциональной клавиатуры, помощь игроку по использованию функциональной клавиатуры;
- Возможность игры двух игроков или игрока с компьютером, возможность выбирать начинающего игру самостоятельно или случайно;
- Игра компьютера по некоторой стратегии;
- Оперативное отображение и контроль ситуации, определение победителя.

Описание игры

На шахматной доске в виде прямоугольников 4x3 расставлены шашки: в одном углу белые, в противоположном – черные. Одним ходом разрешается перемещать шашку на свободную в любом направлении по горизонтали или вертикали или “перепрыгивать” шашкой любое количество шашек независимо от цвета, если они стоят через клетку. Выигрывает тот, кто раньше переместит свои шашки в угол противника.

Анализ игры компьютера

В игре Уголки нет определенной стратегии, по которой мог бы играть компьютер. Поэтому расчет ходов будет производиться следующим образом: будут рассматриваться все возможные варианты ходов на 2 полу хода и будет выбираться лучший из них.

Математическая модель

Представим игровую доску в виде двумерного массива 8x8. Заполним верхний левый угол белыми (в моей программе для красоты красными) шашками, правый нижний угол черными (в моей программе для красоты синими) шашками. Соответствующий элемент массива белой шашке будет иметь значение 1, черной – 2.

Также используется дополнительный массив “маски”, в котором: 0 – обычная клетка поля; положение курсора – 1; выбранная шашка – 2; возможные ходы для выбранной шашки – 3 и 4 (3 – обычный возможный ход, 4 – возможный ход через шашку); при наведении курсором на возможный ход для выбранной шашки – 5 и 6 (5 – при наведении на обычный возможный ход, 6 – при наведении на возможный ход через шашку), при наведении курсором на выделенную шашку – 7 и 8;

Для перерисовки отдельных клеток поля при совершении игроком какого-либо действия используется массив предыдущей маски. После совершения какого-либо действия изменяется массив маски и вызывается функция перерисовки экрана, которая сверяет значения элементов массива маски и предыдущей маски, при каком-либо несовпадении клетка данного элемента перерисовывается в зависимости от нового значения. После работы функции перерисовки весь массив маски копируется в массив предыдущей маски.

Для оценки хода используется двумерный массив “score”, который хранит в себе вес каждой определенной клетки на поле. При переборе всех возможных ходов используется переменная суммы, которая изначально равна 0 и изменяется следующим образом: берется клетка поля и рассматриваются все возможные её ходы на 2 полу хода вперед, при каком-либо ходе берется значение элемента массива “score”, в который клетка попадет при данном ходе и из неё вычитается значение элемента этого же массива, из которого клетка начала свой ход. Текущая разница прибавляется к сумме. По ходу перебора выбирается лучшая сумма. Вместе с ней запоминаются 4 переменные: xn и yn – начальные координаты шашки, xk и yk – конечные координаты шашки. То есть мы берем шашку с координатами [yn][xn] и ставим её в клетку с координатами [yk][xk]. При ходе через шашку используется другая функция перебора с рекурсией, также принимающая, кроме суммы и 4-ёх переменных координат, вектор координат, в которых уже побывала шашка при переборе. Это сделано для того, чтобы избежать зацикливания.

Описание алгоритмов

Функция poisk1(int mas[8][8])

С данной функции начинается перебор всех возможных ходов. Функция принимает лишь массив текущего состояния игрового поля. В глобальных переменных находится 5 переменных: xn, yn, xk, yk, maxs, что и является её возвращаемым значением. Данная функция в последующем обращается к функциям poisk1ch() и poisk3().

Функция poisk1ch(int yn, int xn, int yv, int xv, vector <pair<int, int>> v, int mas[8][8], double sum)

Вторая “фаза” перебора, в которую попадают ходы, сделанную через шашку в функции poisk1(). Все возможные ходы через шашку в первом полу ходе, за исключением начальных, сделанных в функции poisk1(), рассматриваются рекурсивно в данной функции. Затем из этой функции значения ходов передаются в функцию poisk3().

Функция poisk3(int yn, int xn, int yk, int xk, int mas[8][8], double sum)

В данной функции начинается перебор возможных ходов уже второго полу хода. В параметры функции уже добавляется переменная sum из первого полу хода, в сложении с которой в дальнейшем рассматриваются наилучшие ходы. На этапе данной функции рассмотрение части ходов, за исключением ходов через шашку, завершается. Если же на данном этапе есть ходы через шашку, то они рассматриваются в функции poisk3ch().

Функция poisk3ch(int yn, int xn, int yk, int xk, int yv, int xv, vector <pair<int, int>> v, int mas[8][8], double sum)

Последняя стадия перебора всех возможных ходов, в которой рассматриваются ходы через шашку второго полу хода.

Функция draw()

Данная функция не принимает никаких параметров. Она отвечают за полную перерисовку игровой доски, включая её обводку. Возвращаемых значений так же нет.

Функция redraw()

Так же, как и функция draw(), не принимает никаких параметров, т.к. массивы маски и предыдущей маски вынесены в глобальные переменные. Данная функция перерисовывает отдельные ячейки доски с каждым действием пользователя, основываясь на двух массивах маски + массиве координат клеток. Возвращаемых значений нет.

Функция endgame()

Используется при нажатии Esc, выводит на экран надпись “Спасибо за игру!” и ждет нажатия любой клавиши, после чего вызывается return 0, программа закрывается.

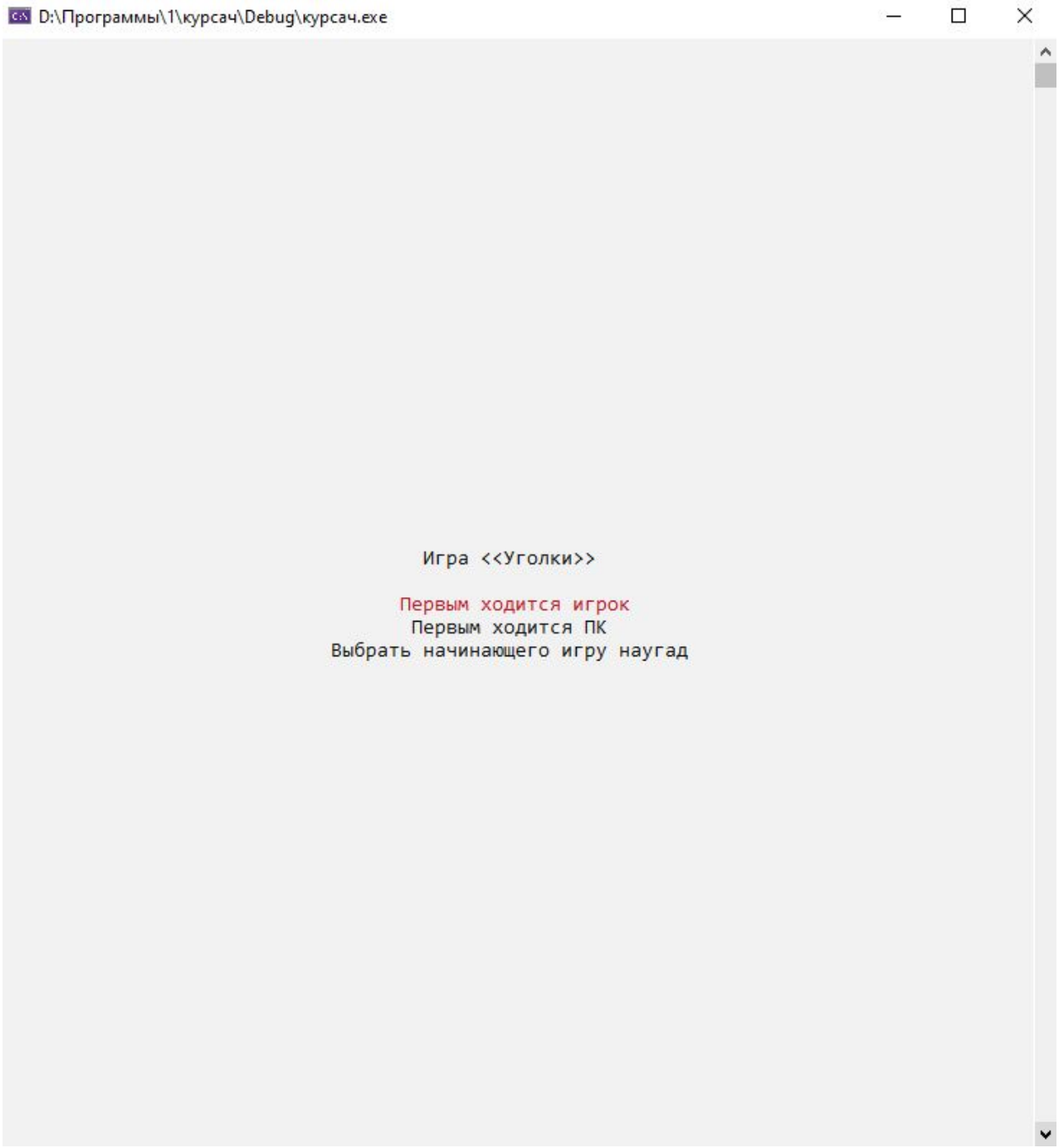
Алгоритмы, не вынесенные в отдельные функции

Во время хода человека считывание клавиш осуществляется с помощью _getch(), после чего, основываясь на int значении клавиши, нажатой на клавиатуре, выполняется проверка и происходят определенные операции, сопутствующие данной клавише.

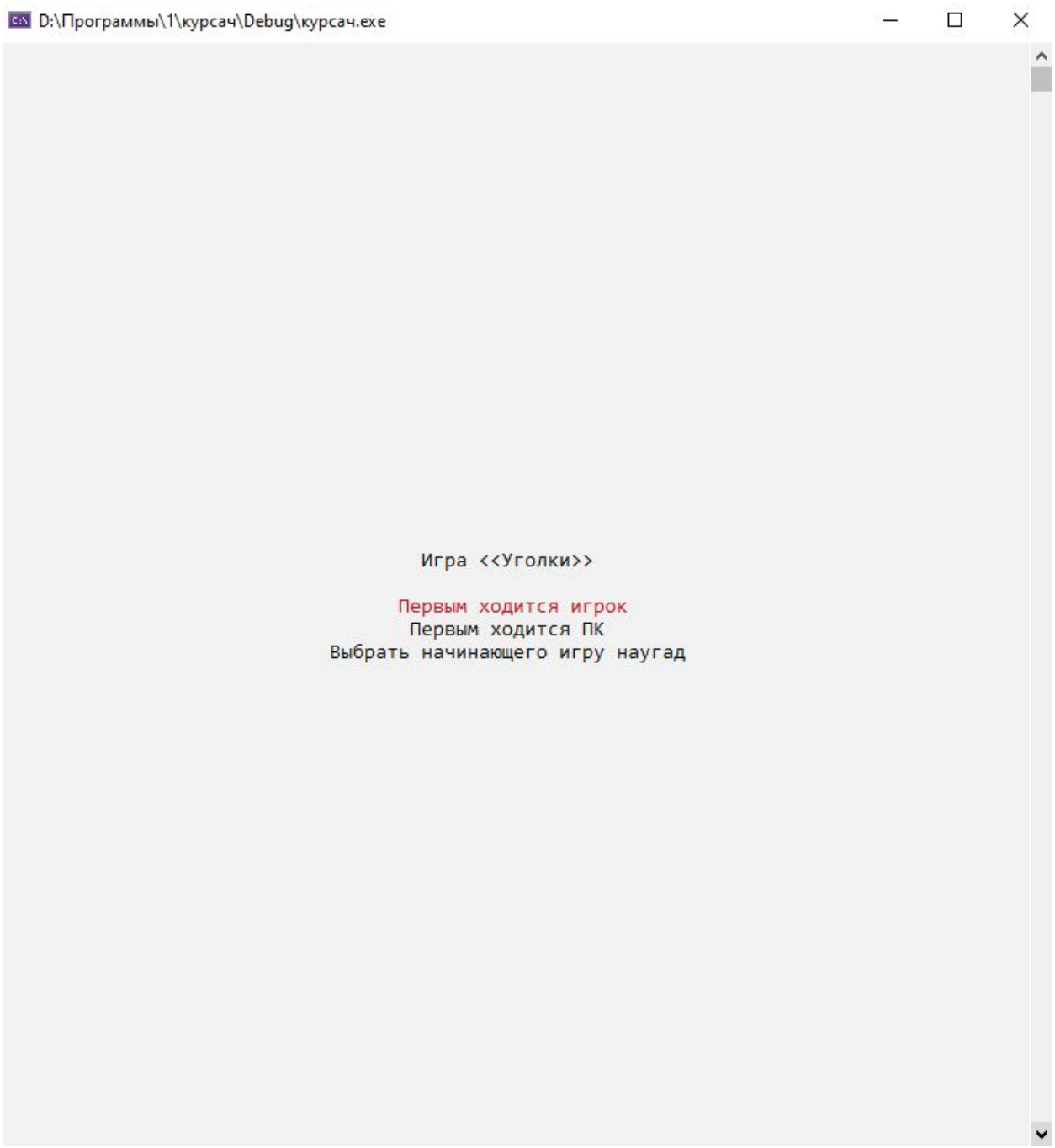
Ниже приедены блок схемы функций поиска наилучшего хода.

Руководство пользования и скриншоты программы

- При запуске программы пользователю нужно выбрать противника: Компьютер либо же другой человек (игра осуществляется на одном компьютере). Выбор осуществляется при помощи стрелочек “вверх” и “вниз”, выбранная строка выделяется красным цветом. Для подтверждения своего выбора нужно нажать клавишу Enter.

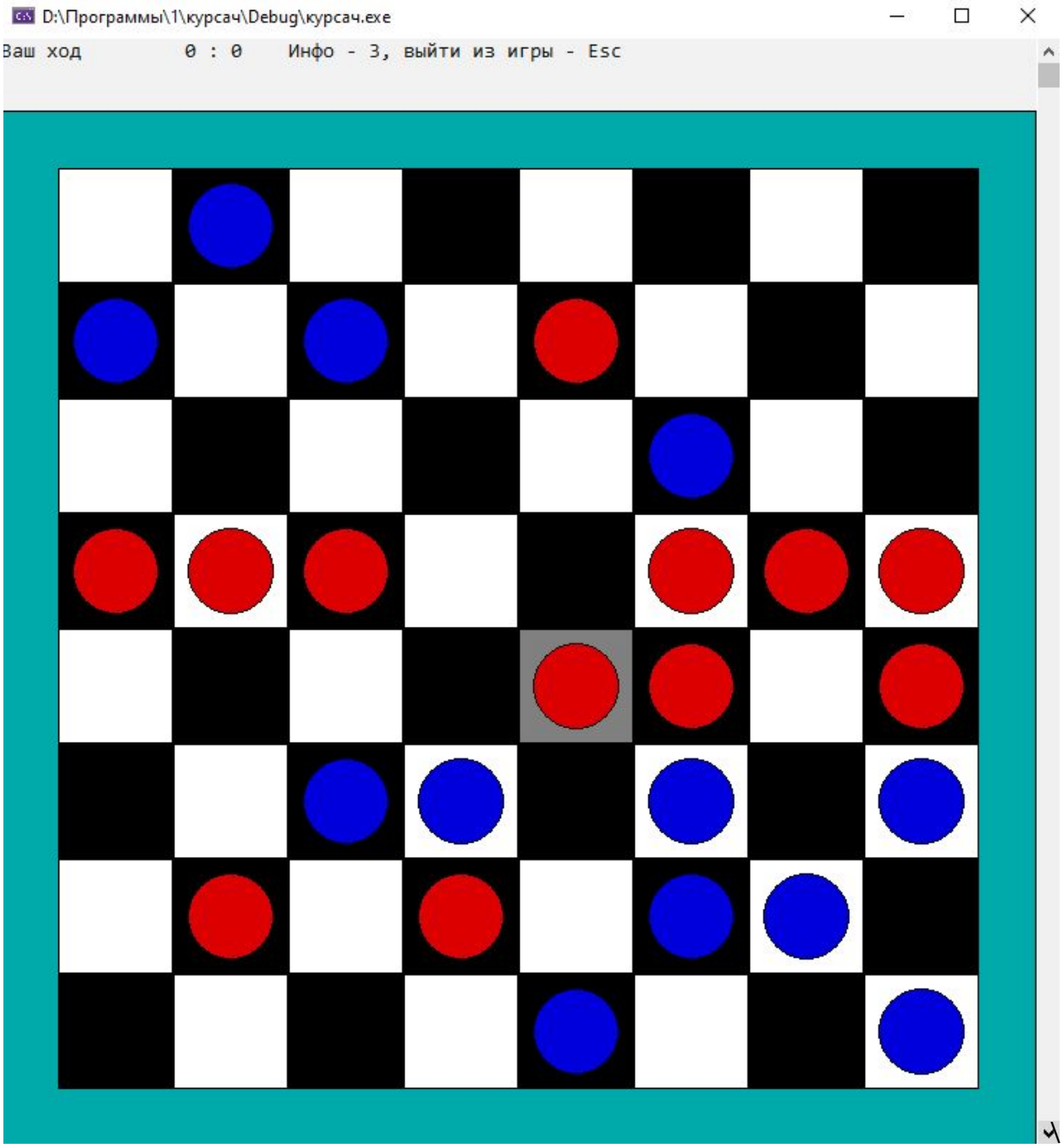


- После выбора противника, пользователю предоставляется выбор начинающего игру, также можно выбрать начинающего игру наугад. Интерфейс такой же, как и на предыдущем шаге, выбор варианта осуществляется с помощью стрелочек, подтверждение действия при нажатии клавиши Enter.

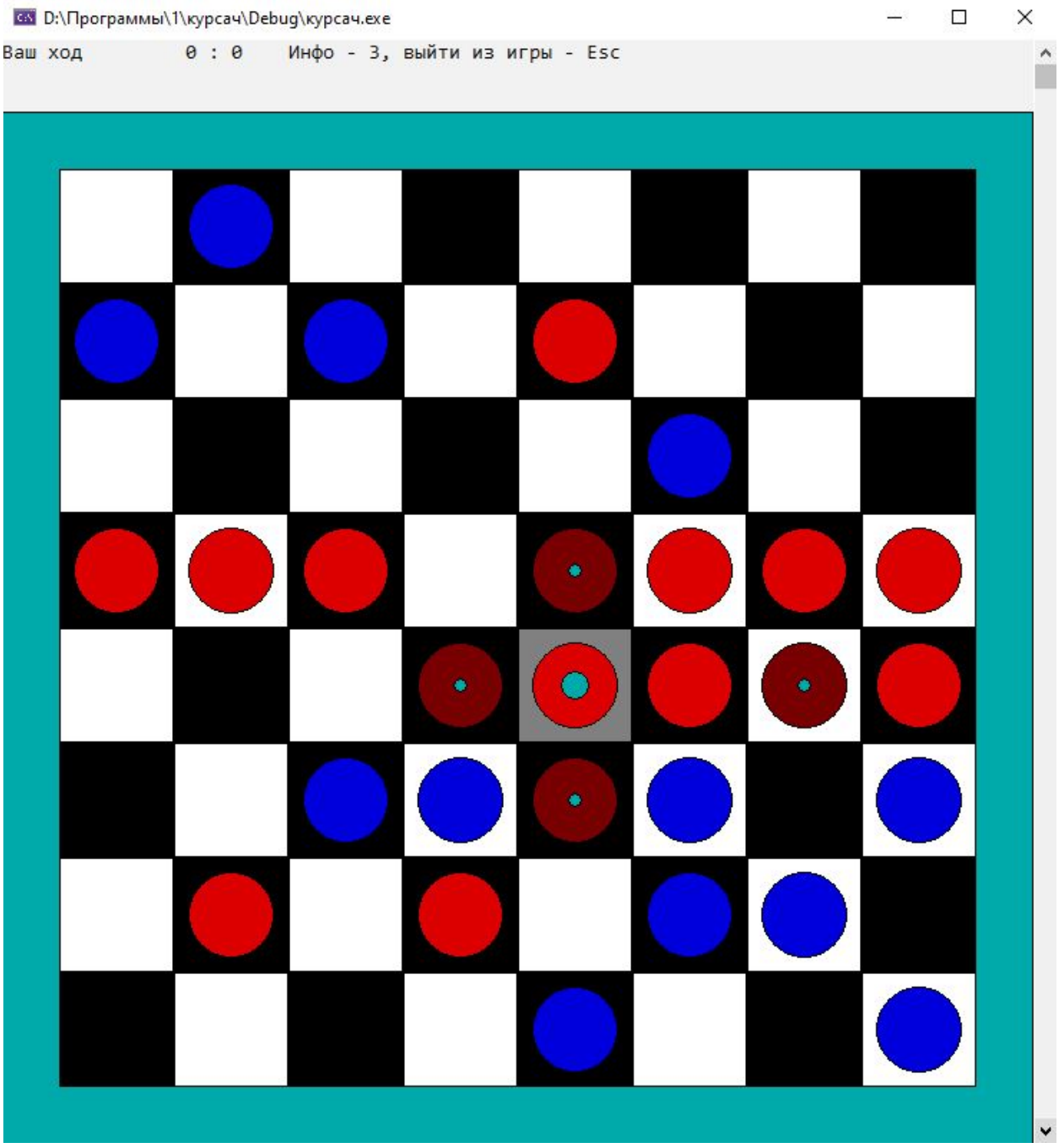


- Как только пользователь выбрал противника и начинающего игру, он попадает непосредственно на игровое поле. Над шахматной доской расположены: строка, отображающая чья сейчас очередь совершать ход, например “Ваш ход” (при игре против ПК), справа находится строка счета в формате “0 : 0”, где первое число – количество побед играющего за красные шашки (левый верхний угол), второе число – количество побед играющего за синие шашки (правый нижний угол). Справа от счета

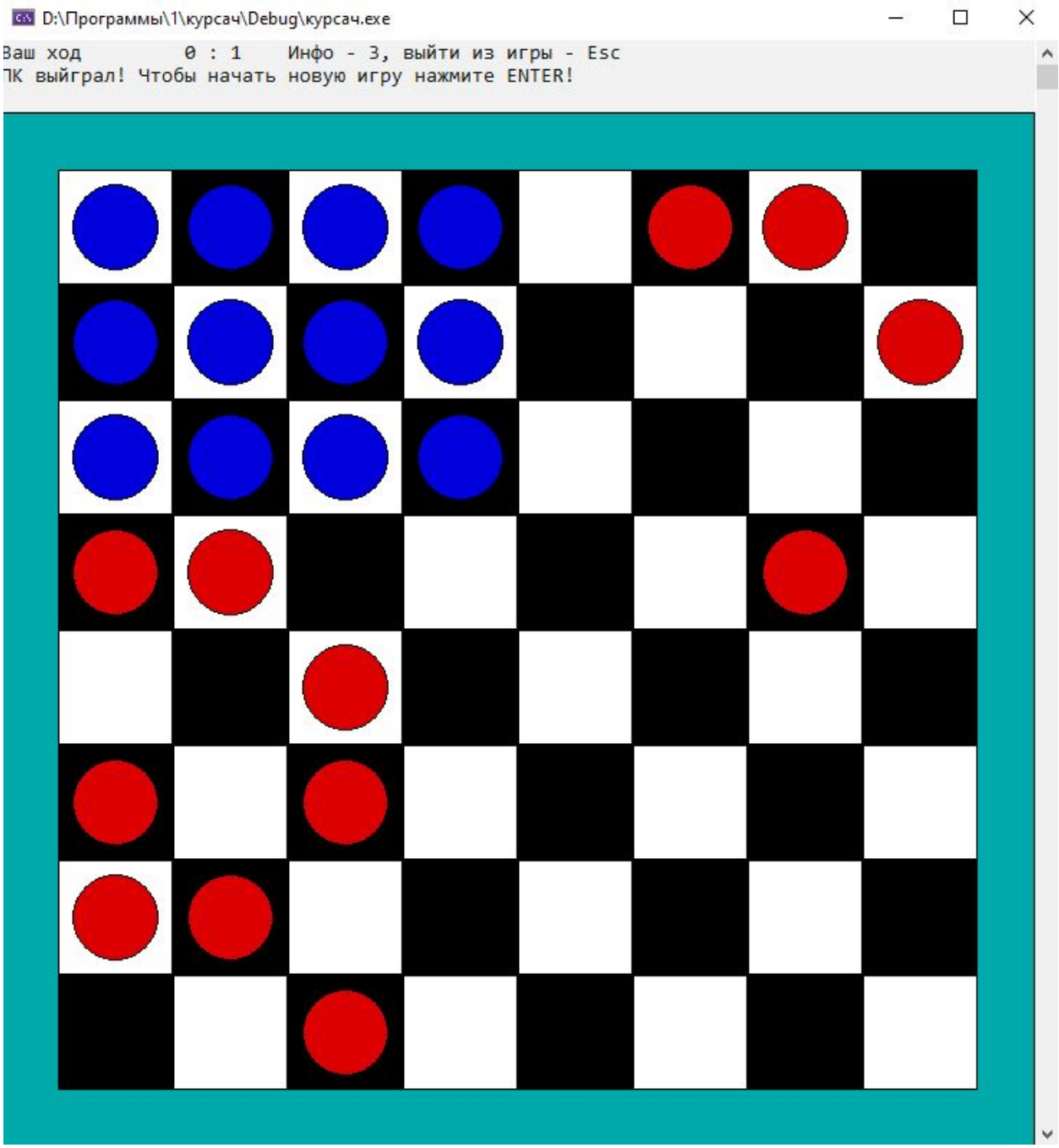
находится строка подсказки “Инфо – 3, выйти из игры - Esc”. Для отображения информации, содержащей информацию о кнопках управления и правилах игры нужно нажать на клавиатуре клавишу 3, для выхода из игры – Esc. На игровой доске курсор, указывающий на текущую клетку, обозначен серым цветом.



- При нажатии клавиши Enter на свою шашку, если ни одна другая шашка при этом еще не выделена, текущая шашка выделяется и выделяется большим кружком в её центре. После этого отображаются все возможные ходы данной шашкой (при включенных подсказках), выделенные кружком поменьше. Нажав на клетку возможного хода, вы перемещаете туда выделенную клетку. Если ход был сделан не через шашку, то право хода передается противнику, если ход был сделан через шашку вам будут предложены все возможные ходы через шашку из этой клетки, либо же право хода будет передано сопернику, если ходов больше нет. Также вы можете остаться в данной ситуации на месте и не продолжать свой ход, просто сняв выделение с шашки.



- При заполнении одним из игроков полностью поля противника своими шашками, будет выведен победитель, изменится текущий счет и будет предложено сыграть в новую игру.



- Дополнительные клавиши:
    1. Полная перерисовка игрового поля (при масштабировании окна графика сбивается) – клавиша 1.
    2. Пропуск хода – клавиша 0.
    3. Включение/отключение подсказок – 2.

Код основных алгоритмов

```
void poisk3ch(int yn, int xn, int yk, int xk, int yv, int xv, vector <pair<int, int>> v, int mas[8][8], double sum)
{
	if (yv > 1 && mas[yv - 1][xv] != 0 && mas[yv - 2][xv] == 0)
	{
		bool find = 0;

		for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv - 2, xv))
		{
			find = 1;
			break;
		}

		if (find == 0)
		{
			double sc = score[yv - 2][xv] - score[yv][xv];
			if (sum + sc > maxs)
			{
				maxs = sum + sc;
				xn1 = xn;
				yn1 = yn;
				xk1 = xk;
				yk1 = yk;
			}

			swap(mas[yv][xv], mas[yv - 2][xv]);
			v.push_back(make_pair(yv - 2, xv));
			poisk3ch(yn, xn, yk, xk, yv - 2, xv, v, mas, sum + sc);
			swap(mas[yv][xv], mas[yv - 2][xv]);
			v.pop_back();
		}

	}
```

```
if (yv < 6 && mas[yv + 1][xv] != 0 && mas[yv + 2][xv] == 0)
{
	bool find = 0;

	for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv + 2, xv))
	{
		find = 1;
		break;
	}

	if (find == 0)
	{
		double sc = score[yv + 2][xv] - score[yv][xv];
		if (sum + sc > maxs)
		{
			maxs = sum + sc;
			xn1 = xn;
			yn1 = yn;
			xk1 = xk;
			yk1 = yk;
		}

		swap(mas[yv][xv], mas[yv + 2][xv]);
		v.push_back(make_pair(yv + 2, xv));
		poisk3ch(yn, xn, yk, xk, yv + 2, xv, v, mas, sum + sc);
		swap(mas[yv][xv], mas[yv + 2][xv]);
		v.pop_back();
	}
}

if (xv > 1 && mas[yv][xv - 1] != 0 && mas[yv][xv - 2] == 0)
{
	bool find = 0;
```

```
		for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv, xv - 2))
		{
			find = 1;
			break;
		}

		if (find == 0)
		{
			double sc = score[yv][xv - 2] - score[yv][xv];
			if (sum + sc > maxs)
			{
				maxs = sum + sc;
				xn1 = xn;
				yn1 = yn;
				xk1 = xk;
				yk1 = yk;
			}

			swap(mas[yv][xv], mas[yv][xv - 2]);
			v.push_back(make_pair(yv, xv - 2));
			poisk3ch(yn, xn, yk, xk, yv, xv - 2, v, mas, sum + sc);
			swap(mas[yv][xv], mas[yv][xv - 2]);
			v.pop_back();
		}
	}

	if (xv < 6 && mas[yv][xv + 1] != 0 && mas[yv][xv + 2] == 0)
	{
		bool find = 0;

		for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv, xv + 2))
		{
			find = 1;
```

```
                break;
            }

            if (find == 0)
            {
                double sc = score[yv][xv + 2] - score[yv][xv];
                if (sum + sc > maxs)
                {
                    maxs = sum + sc;
                    xn1 = xn;
                    yn1 = yn;
                    xk1 = xk;
                    yk1 = yk;
                }

                swap(mas[yv][xv], mas[yv][xv + 2]);
                v.push_back(make_pair(yv, xv + 2));
                poisk3ch(yn, xn, yk, xk, yv, xv + 2, v, mas, sum + sc);
                swap(mas[yv][xv], mas[yv][xv + 2]);
                v.pop_back();
            }
        }
}

void poisk3(int yn, int xn, int yk, int xk, int mas[8][8], double sum)
{
        for (int i = 0; i < 8; i++)
        {
            for (int j = 0; j < 8; j++)
            {
                if (mas[i][j] == 2)
                {
                    if (i > 0 && mas[i - 1][j] == 0)
                    {
```

```
            double sc = score[i - 1][j] - score[i][j];
            if (sum + sc > maxs)
            {
                maxs = sum + sc;
                xn1 = xn;
                yn1 = yn;
                xk1 = xk;
                yk1 = yk;
            }
        }
        else if (i > 1 && mas[i - 2][j] == 0)
        {
            double sc = score[i - 2][j] - score[i][j];

            if (sum + sc > maxs)
            {
                maxs = sum + sc;
                xn1 = xn;
                yn1 = yn;
                xk1 = xk;
                yk1 = yk;
            }

            vector <pair<int, int>> n;
            n.push_back(make_pair(i, j));
            n.push_back(make_pair(i - 2, j));

            swap(mas[i][j], mas[i - 2][j]);
            poisk3ch(yn, xn, yk, xk, i - 2, j, n, mas, sum + sc);
            swap(mas[i][j], mas[i - 2][j]);
        }

        if (i < 7 && mas[i + 1][j] == 0)
        {
```

```
            double sc = score[i + 1][j] - score[i][j];
            if (sum + sc > maxs)
            {
                maxs = sum + sc;
                xn1 = xn;
                yn1 = yn;
                xk1 = xk;
                yk1 = yk;
            }
        }
        else if (i < 6 && mas[i + 2][j] == 0)
        {
            double sc = score[i + 2][j] - score[i][j];

            if (sum + sc > maxs)
            {
                maxs = sum + sc;
                xn1 = xn;
                yn1 = yn;
                xk1 = xk;
                yk1 = yk;
            }

            vector <pair<int, int>> n;
            n.push_back(make_pair(i, j));
            n.push_back(make_pair(i + 2, j));

            swap(mas[i][j], mas[i + 2][j]);
            poisk3ch(yn, xn, yk, xk, i + 2, j, n, mas, sum + sc);
            swap(mas[i][j], mas[i + 2][j]);
        }

        if (j > 0 && mas[i][j - 1] == 0)
        {
```

```
            double sc = score[i][j - 1] - score[i][j];
            if (sum + sc > maxs)
            {
                maxs = sum + sc;
                xn1 = xn;
                yn1 = yn;
                xk1 = xk;
                yk1 = yk;
            }
        }
        else if (j > 1 && mas[i][j - 2] == 0)
        {
            double sc = score[i][j - 2] - score[i][j];

            if (sum + sc > maxs)
            {
                maxs = sum + sc;
                xn1 = xn;
                yn1 = yn;
                xk1 = xk;
                yk1 = yk;
            }

            vector <pair<int, int>> n;
            n.push_back(make_pair(i, j));
            n.push_back(make_pair(i, j - 2));

            swap(mas[i][j], mas[i][j - 2]);
            poisk3ch(yn, xn, yk, xk, i, j - 2, n, mas, sum + sc);
            swap(mas[i][j], mas[i][j - 2]);
        }

        if (j < 7 && mas[i][j + 1] == 0)
        {
```

```
                        double sc = score[i][j + 1] - score[i][j];
                        if (sum + sc > maxs)
                        {
                                maxs = sum + sc;
                                xn1 = xn;
                                yn1 = yn;
                                xk1 = xk;
                                yk1 = yk;
                        }
                }
                else if (j < 6 && mas[i][j + 2] == 0)
                {
                        double sc = score[i][j + 2] - score[i][j];

                        if (sum + sc > maxs)
                        {
                                maxs = sum + sc;
                                xn1 = xn;
                                yn1 = yn;
                                xk1 = xk;
                                yk1 = yk;
                        }

                        vector <pair<int, int>> n;
                        n.push_back(make_pair(i, j));
                        n.push_back(make_pair(i, j + 2));

                        swap(mas[i][j], mas[i][j + 2]);
                        poisk3ch(yn, xn, yk, xk, i, j + 2, n, mas, sum + sc);
                        swap(mas[i][j], mas[i][j + 2]);
                }
        }
    }
}
```

```
}

void poisk1ch(int yn, int xn, int yv, int xv, vector <pair<int, int>> v, int mas[8][8], double sum)
{
	if (yv > 1 && mas[yv - 1][xv] != 0 && mas[yv - 2][xv] == 0)
	{
		bool find = 0;

		for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv - 2, xv))
		{
			find = 1;
			break;
		}

		if (find == 0)
		{
			double sc = score[yv - 2][xv] - score[yv][xv];
			if (sum + sc >= maxs)
			{
				maxs = sum + sc;
				yn1 = yn;
				xn1 = xn;
				yk1 = yv - 2;
				xk1 = xv;
			}

			swap(mas[yv][xv], mas[yv - 2][xv]);
			v.push_back(make_pair(yv - 2, xv));
			poisk3(yn, xn, yv - 2, xv, mas, sum + sc);
			poisk1ch(yn, xn, yv - 2, xv, v, mas, sum + sc);
			swap(mas[yv][xv], mas[yv - 2][xv]);
			v.pop_back();
		}
```

```
}

if (yv < 6 && mas[yv + 1][xv] != 0 && mas[yv + 2][xv] == 0)
{
	bool find = 0;

	for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv + 2, xv))
	{
		find = 1;
		break;
	}

	if (find == 0)
	{
		double sc = score[yv + 2][xv] - score[yv][xv];
		if (sum + sc >= maxs)
		{
			maxs = sum + sc;
			yn1 = yn;
			xn1 = xn;
			yk1 = yv + 2;
			xk1 = xv;
		}

		swap(mas[yv][xv], mas[yv + 2][xv]);
		v.push_back(make_pair(yv + 2, xv));
		poisk3(yn, xn, yv + 2, xv, mas, sum + sc);
		poisk1ch(yn, xn, yv + 2, xv, v, mas, sum + sc);
		swap(mas[yv][xv], mas[yv + 2][xv]);
		v.pop_back();
	}
}

if (xv > 1 && mas[yv][xv - 1] != 0 && mas[yv][xv - 2] == 0)
```

```
{
	bool find = 0;

	for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv, xv - 2))
	{
		find = 1;
		break;
	}

	if (find == 0)
	{
		double sc = score[yv][xv - 2] - score[yv][xv];
		if (sum + sc >= maxs)
		{
			maxs = sum + sc;
			yn1 = yn;
			xn1 = xn;
			yk1 = yv;
			xk1 = xv - 2;
		}

		swap(mas[yv][xv], mas[yv][xv - 2]);
		v.push_back(make_pair(yv, xv - 2));
		poisk3(yn, xn, yv, xv - 2, mas, sum + sc);
		poisk1ch(yn, xn, yv, xv - 2, v, mas, sum + sc);
		swap(mas[yv][xv], mas[yv][xv - 2]);
		v.pop_back();
	}
}

if (xv < 6 && mas[yv][xv + 1] != 0 && mas[yv][xv + 2] == 0)
{
	bool find = 0;
```

```
                for (int i = 0; i < v.size(); i++) if (v[i] == make_pair(yv, xv + 2))
                {
                        find = 1;
                        break;
                }

                if (find == 0)
                {
                        double sc = score[yv][xv + 2] - score[yv][xv];
                        if (sum + sc >= maxs)
                        {
                                maxs = sum + sc;
                                yn1 = yn;
                                xn1 = xn;
                                yk1 = yv;
                                xk1 = xv + 2;
                        }

                        swap(mas[yv][xv], mas[yv][xv + 2]);
                        v.push_back(make_pair(yv, xv + 2));
                        poisk3(yn, xn, yv, xv + 2, mas, sum + sc);
                        poisk1ch(yn, xn, yv, xv + 2, v, mas, sum + sc);
                        swap(mas[yv][xv], mas[yv][xv + 2]);
                        v.pop_back();
                }
        }
}

void poisk1(int mas[8][8])
{
        for (int i = 0; i < 8; i++)
        {
                for (int j = 0; j < 8; j++)
                {
```

```
if (mas[i][j] == 2)
{
	if (i > 0 && mas[i - 1][j] == 0)
	{
		double sc = score[i - 1][j] - score[i][j];
		if (sc >= maxs)
		{
			maxs = sc;
			yn1 = i;
			xn1 = j;
			yk1 = i - 1;
			xk1 = j;
		}

		swap(mas[i][j], mas[i - 1][j]);
		poisk3(i, j, i - 1, j, mas, sc);
		swap(mas[i][j], mas[i - 1][j]);
	}
	else if (i > 1 && mas[i - 2][j] == 0)
	{
		vector <pair<int, int>> n;
		n.push_back(make_pair(i, j));
		n.push_back(make_pair(i - 2, j));

		double sc = score[i - 2][j] - score[i][j];
		if (sc >= maxs)
		{
			maxs = sc;
			yn1 = i;
			xn1 = j;
			yk1 = i - 2;
			xk1 = j;
		}
```

```
            swap(mas[i][j], mas[i - 2][j]);
            poisk3(i, j, i - 2, j, mas, sc);
            poisk1ch(i, j, i - 2, j, n, mas, sc);
            swap(mas[i][j], mas[i - 2][j]);
    }

    if (i < 7 && mas[i + 1][j] == 0)
    {
            double sc = score[i + 1][j] - score[i][j];
            if (sc >= maxs)
            {
                    maxs = sc;
                    yn1 = i;
                    xn1 = j;
                    yk1 = i + 1;
                    xk1 = j;
            }

            swap(mas[i][j], mas[i + 1][j]);
            poisk3(i, j, i + 1, j, mas, sc);
            swap(mas[i][j], mas[i + 1][j]);
    }
    else if (i < 6 && mas[i + 2][j] == 0)
    {
            vector <pair<int, int>> n;
            n.push_back(make_pair(i, j));
            n.push_back(make_pair(i + 2, j));

            double sc = score[i + 2][j] - score[i][j];
            if (sc >= maxs)
            {
                    maxs = sc;
                    yn1 = i;
                    xn1 = j;
```

```
                yk1 = i + 2;
                xk1 = j;
            }

            swap(mas[i][j], mas[i + 2][j]);
            poisk3(i, j, i + 2, j, mas, sc);
            poisk1ch(i, j, i + 2, j, n, mas, sc);
            swap(mas[i][j], mas[i + 2][j]);
        }

        if (j > 0 && mas[i][j - 1] == 0)
        {
            double sc = score[i][j - 1] - score[i][j];
            if (sc >= maxs)
            {
                maxs = sc;
                yn1 = i;
                xn1 = j;
                yk1 = i;
                xk1 = j - 1;
            }

            swap(mas[i][j], mas[i][j - 1]);
            poisk3(i, j, i, j - 1, mas, sc);
            swap(mas[i][j], mas[i][j - 1]);
        }
        else if (j > 1 && mas[i][j - 2] == 0)
        {
            vector <pair<int, int>> n;
            n.push_back(make_pair(i, j));
            n.push_back(make_pair(i, j - 2));

            double sc = score[i][j - 2] - score[i][j];
            if (sc >= maxs)
```

```
		{
			maxs = sc;
			yn1 = i;
			xn1 = j;
			yk1 = i;
			xk1 = j - 2;
		}

		swap(mas[i][j], mas[i][j - 2]);
		poisk3(i, j, i, j - 2, mas, sc);
		poisk1ch(i, j, i, j - 2, n, mas, sc);
		swap(mas[i][j], mas[i][j - 2]);
	}

	if (j < 7 && mas[i][j + 1] == 0)
	{
		double sc = score[i][j + 1] - score[i][j];
		if (sc >= maxs)
		{
			maxs = sc;
			yn1 = i;
			xn1 = j;
			yk1 = i;
			xk1 = j + 1;
		}

		swap(mas[i][j], mas[i][j + 1]);
		poisk3(i, j, i, j + 1, mas, sc);
		swap(mas[i][j], mas[i][j + 1]);
	}
	else if (j < 6 && mas[i][j + 2] == 0)
	{
		vector <pair<int, int>> n;
		n.push_back(make_pair(i, j));
```

```
                                        n.push_back(make_pair(i, j + 2));

                                        double sc = score[i][j + 2] - score[i][j];
                                        if (sc >= maxs)
                                        {
                                                maxs = sc;
                                                yn1 = i;
                                                xn1 = j;
                                                yk1 = i;
                                                xk1 = j + 2;
                                        }
                                        swap(mas[i][j], mas[i][j + 2]);
                                        poisk3(i, j, i, j + 2, mas, sc);
                                        poisk1ch(i, j, i, j + 2, n, mas, sc);
                                        swap(mas[i][j], mas[i][j + 2]);
                                }
                        }
                }
        }
}
```