**Индивидуальное задание**

**1. Пользовательская документация**

При запуске игры высвечивается окно, рисунок 1, которое включает в себя:

1. меню игры;
2. поле 21x21 квадратов (каждый квадрат окрашивается в один из 3 цветов);
3. табло, показывающие результаты игры в данный момент (обновляется после сделанного хода).

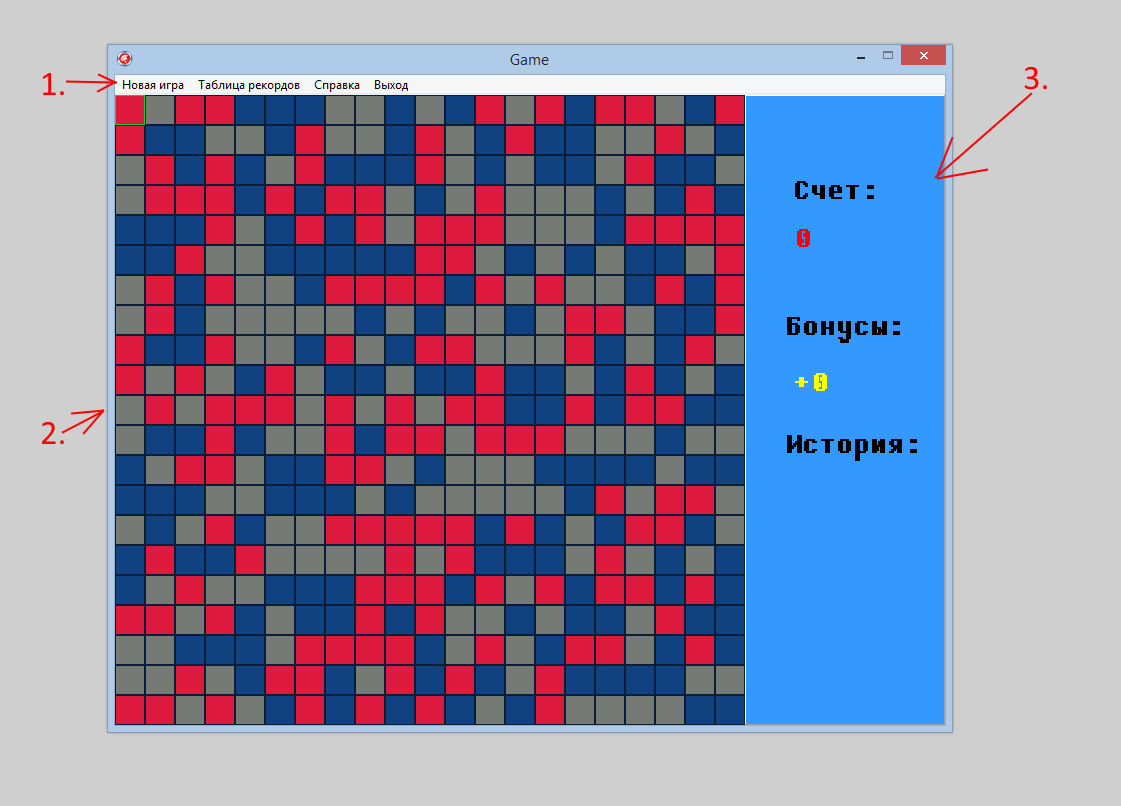


Рисунок 1 – Главное окно программы

Поле помимо квадратов имеет еще и курсор, который можно перемещать стрелками вверх, вниз, вправо и влево.

Как видно из примера на рисунке 1, на игровом поле в некоторых местах образуются связки одинаковых цветов, например в левом верхнем углу 5 синих квадратов. При нажатии на один квадрат с помощью клавиши Enter, из связки вся цепочка удаляется, а освободившееся пространство заполняется белым цветом, рисунок 2.



Рисунок 2 – Пример нажатия на один из квадратов связки

При этом, если пользователь удалит цепочку, над которой находились квадраты другого цвета, то осуществляется сдвиг этих квадратов. Например, на рисунке 3 при удалении цепочки, которую образуют квадраты [16,2]; [17,1]; [17,2]; [18,1], получится следующая картина, изображенная на рисунке 4.

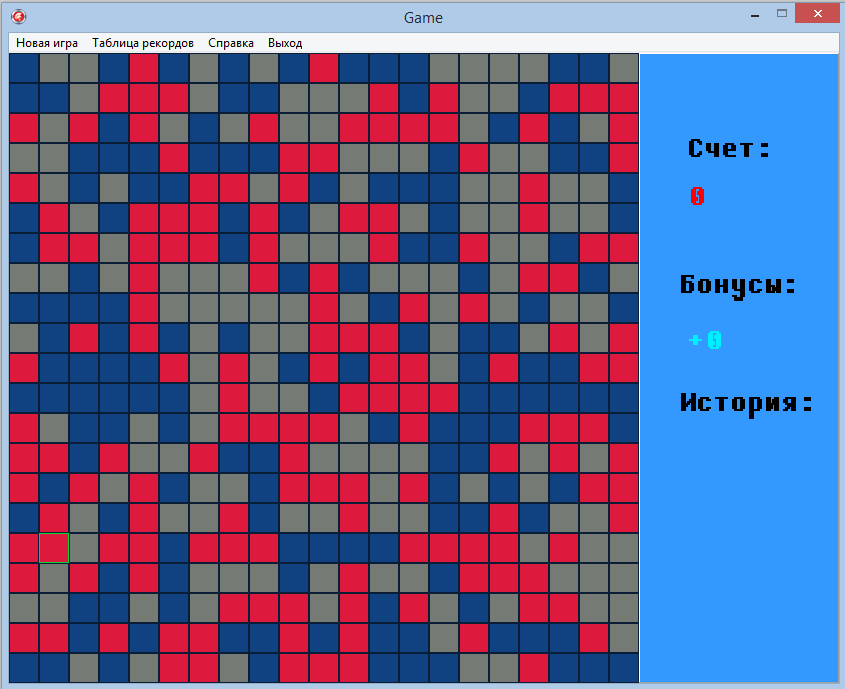


Рисунок 3 – Поле до удаления цепочки

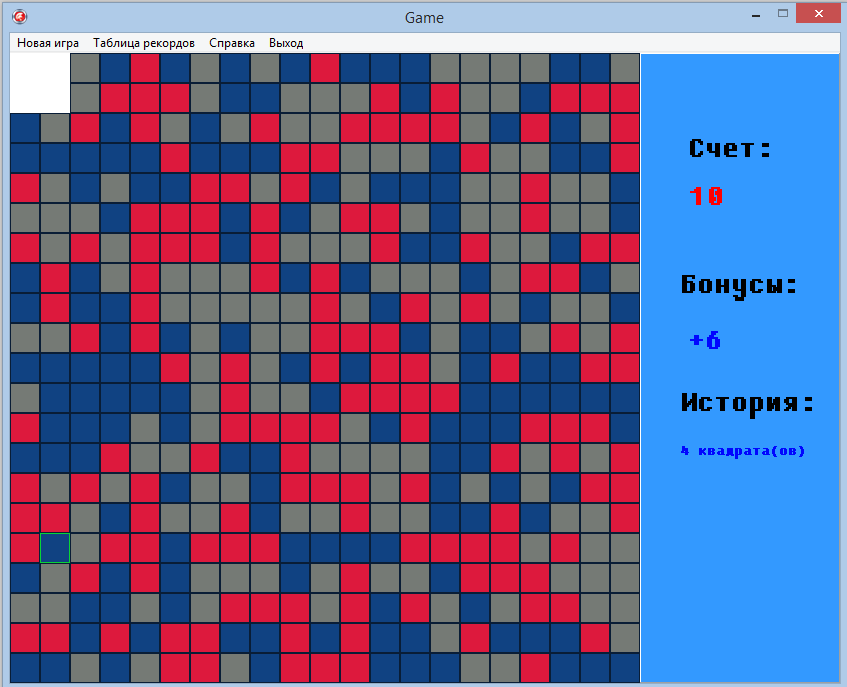


Рисунок 4 – Поле после удаления цепочки

Также при образовании пустого столбца, все столбцы, находящиеся слева от пустого, сдвигаются вправо.

Задача игры заключается в том, чтобы очистить поле и получить как можно больше очков. Очки могут быть начислены в следующих при следующих событиях:

* при удалении клеток начисляется столько очков, сколько удалено квадратов;
* за каждый квадрат в цепочке начисляется дополнительный бонус, увеличивающийся на единицу для каждого последующего квадрата в цепочке (первый квадрат = 1 очко, второй квадрат = 2 очка, третий квадрат = 3 очка и тд.);
* в случае, если игрок очисти поле полностью, то он получает дополнительные 5000 очков.

После завершения игры высвечивается окно, рисунок 5, на котором изображено:

* счет;
* самая длинная удаленная цепочка за игру;
* игроки, показавшие лучшие результаты в той или иной категории.

Так же, если пользователь показал результат лучше, чем игрок, уже находящийся в списке, то высвечивается дополнительное окно для занесения игрока в «Таблицу рекордов».

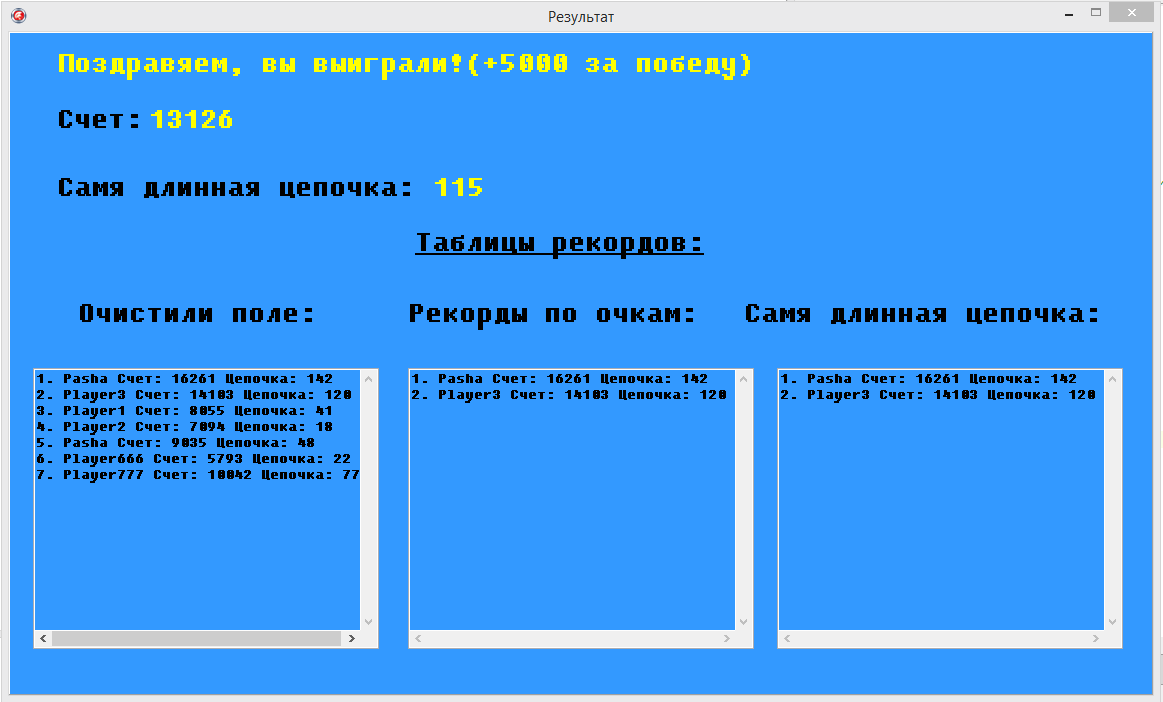


Рисунок 5 – Таблица результата

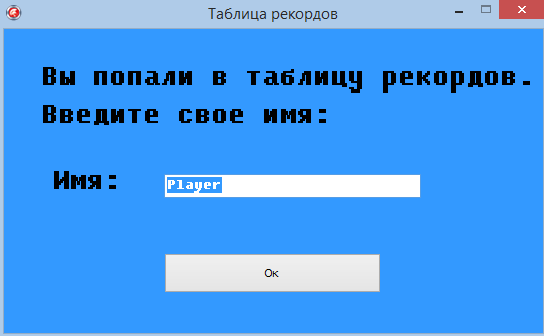


Рисунок 6 – Запись в таблицу рекордов

**2. Техническая документация.**

Теперь рассмотрим, как реализуется игра.

Сначала нужно сгенерировать поле. Для этого мы создаем массив размером поля и заполняем ячейки случайными цифрами от 1 до 3. Одновременно рисуем добавленный квадрат, рисунок 7.

for I := 0 to 20 do begin //Генерация игрового поля.

for j := 0 to 20 do begin

k:=random(3)+1; //Выбор случайного числа.

case k of

1: begin color:=SetOldColor(k); pole[i,j]:=1 end;

2: begin color:=SetOldColor(k); pole[i,j]:=2 end;

3: begin color:=SetOldColor(k); pole[i,j]:=3 end;

end;

DrawRectangle (i,j,color,Image1);

end; //Конец цикла j

end; //Конец цикла i

Рисунок 7 – Генерация поля

Рассмотрим функции и процедуры, участвующие в построении игрового поля.

Функция SetOldColor, рисунок 8, принимает значение элемента матрицы и возвращает цвет, который скрывает данный элемент. Так же были использованы оттенки, так как стандартные цвета слишком яркие.

function SetOldColor(k: integer):Tcolor;

begin

case k of

0: begin SetOldColor:=clWhite; end; //Белый

1: begin SetOldColor:=$3D19DD; end; //Оттенок синего

2: begin SetOldColor:=$824210; end; //Оттенок красного

3: begin SetOldColor:=$757A75; end; //Оттенок серого

end;

end;

Рисунок 8 – Функция возврата цвета по числу

Процедура DrawRectangle, рисунок 9, создана для рисования квадратов. Принимает она строку и столбец в матрице, а также цвет, в который нужно окрасить квадрат. Если цвет квадрата белый, то обводку делаем белой, в других случаях обводку выбираем черной.

procedure DrawRectangle (i,j: integer; color:TColor; Image1: Timage);

var color2: Tcolor;

begin

if color=clWhite then begin color2:= clWhite; end //Определение, нужна ли обводка.

else begin color2:=$361D09; end;

Image1.Canvas.Pen.Color:=color2; //Цвет обводки.

Image1.Canvas.Brush.Color:= color; //Цвет квадрата.

Image1.canvas.Rectangle(30\*j,30\*i,30\*(j+1),30\*(i+1));

end;

Рисунок 9 – Рисование квадрата

Последнее что нужно сделать перед игрой – позаботиться об указателе, который будет указывать, в какой точке поля находится пользователь, рисунок 10.

oldcolor:=SetOldColor(pole[Y,X]); //Цвет поля, на котором стоит указатель.

Place(color,image1,Point,X,Y); //Поле, на котором стоит указатель.

label2.Caption:=inttostr(Score); //Очки.

label4.Caption:=('+0'); //Бонусы.

history1.caption:=''; //История ходов.

history2.caption:='';

history3.caption:='';

history4.caption:='';

history5.caption:='';

Рисунок 10 – Последние подготовительные этапы

Указатель представляет собой квадрат, цвет которого совпадает с цветом квадрата, на который он указывает, но обводка у него другая. Когда передвигаемся по полю, то сначала отрисовываем квадрат, на котором стоял указатель, затем меняем координаты и рисуем новый квадрат, с цветом другой обводки.

begin

case Key of

VK\_UP: //Клавиша: стрелка вверх.

begin

if Y<>0 then begin //Следим за тем, чтобы пользователь не ушел за границы поля.

DrawRectangle (Y,X,oldcolor,Image1); //рисуем старый квадрат.

Y:=Y-1; //Меняем координаты.

oldcolor:=SetOldColor(pole[Y,X]); //Цвет поля, на котором стоит указатель.

end;

end;

<….>

Place(color,image1,Point,X,Y); //Рисуем квадрат, на котором стоит указатель.

end;

Рисунок 11 – Отрывок передвижения указателя

Теперь рассмотрим, как происходит процесс удаления цепочки с ее последствиями. На рисунке 12 отображена реакция при нажатии на клавишу Enter.

begin

case Key of

VK\_RETURN: //Клавиша: Enter.

begin

if pole[Y,X]<>0 then begin //Если указатель не стоит на пустой клетке.

DeleteRectangle (X,Y,NumberClearX,MaxLongWayoldcolor,pole,masRecord,Score,Win); //Удаление.

end;

end;

Рисунок 12 – Обработка клавиши Enter

Рассмотрим переменные, поступающие в процедуру DeleteRectangle:

* X,Y - координаты указателя;
* NumberClearX - число пустых столбов (при запуске игры NumberClearX:=0);
* MaxLongWay - максимальная длинна цепочки (для статистики);
* Oldcolor - цвет квадрата, на котором стоит указатель;
* Pole - массив, хранящий в себе цвета квадратов;
* masRecord - массив, в котором храним сведения о добавлении в таблицу рекордов. Изначально три числа (три категории в «таблице рекордов») в массиве заполняются “-1”, что означает, что человек не выиграл и не попал в таблицу рекордов, если будет другое число, то это означает, какое место в таблице рекордов он занял);
* Score – очки.
* Win – победил или нет (при запуске игры Win:=false;).

На рисунке 13 изображен подготовительный этап в процедуре.

Score1:=Score; //Очки до удаления.

NumberDeleteRectangle:=0; //Количество удаленных квадратов.

bonus:=0; //Бонусы.

b:=false; //Нужно ли сдвигать столбы(Столбы сдвигаются когда появляется пустой столб)

ColorPlace:=pole[Y,X]; //Получаем “цвет поля” на котором стоим.

Рисунок 13 – Подготовительный этап в процедуре DeleteRectangle

Часть кода процедуры определяет, есть ли цепочка кубиков с одинаковым цветом с помощью функции CheckPlace, которая возвращает “true”, если нашла рядом стоящий кубик с таким же цветом, а та же изменяет в матрице значение цепочки квадратов на “0”, рисунок 14.

if CheckPlace(X,Y,ColorPlace,pole,Score,bonus,NumberDeleteRectangle)=true then

//Если рядом находится хоть один кубик с таким же цветом.

begin

pole[Y,X]:=0;

for I := 0 to 20 do begin

for j := 0 to 20 do begin

if pole[i,j]=0 then begin //Ищем нули

MoveMassive(i,j,color,oldcolor,Image1,pole); //Сдвигаем строки, стоящние выше удаляемых кубиков.

end; //Конец if

end; //Конец цикла j

end; //Конец цикла i

Рисунок 14 – Основная часть процедуры DeleteRectangle

Если значение функции CheckPlace - “истина”, то пробегаемся по матрице поля и ищем нули, которые поставила функция. Нашли – сдвинули строки в столбце вниз, рисунок 15.

procedure MoveMassive(a,b:integer; color:TColor; var oldcolor:TColor; Image1: Timage; var pole: mas);

var i,k,j: integer;

begin

for i := a downto 1 do begin //Со строки, в которой нашли ноль до первой строки.

pole[i,b]:=pole[i-1,b];

color:=SetOldColor(pole[i,b]);

DrawRectangle (i,b,color,Image1); //Рисуем новый кубик на месте старого

end; //Конец цикла

pole[0,b]:=0;

DrawRectangle (0,b,clWhite,Image1); //Рисуем белый кубик в 1 строке.

end;

Рисунок 15 – Сдвиг соседних квадратов для заполнения удаленной цепочки

Рассмотрим отрывок кода функции CheckPlace, рисунок 16, которая проверяет, есть ли кубик такого же цвета сверху, снизу, справа и слева. Т.е. мы идем влево, вправо, вверх и вниз от выбранного квадрата, если нашли такой же цвет, то зануляем место найденного квадрата и относительно его запускаем снова функцию CheckPlace. В итоге, у нас занулится цепочка стоящих квадратов в матрице.

Следующим шагом, после удаления цепочки и сдвига кубиков, идет проверка на пустые столбцы, которые могут образоваться между двумя «Горами кубиков» после удаления цепочки. Если столб находится, то всем элементам, стоящим слева от столба применяется действие pole[j,i]:=pole[j,i-1]; При этом NumberClearX (Количество свободных столбцов) увеличится на единицу. То есть, в следующий раз, когда будем искать пустые столбцы, мы пройдемся по (20-NumberClearX) элементам, потому что точно знаем, что от 1 столбца до NumberClearX пустые столбцы и левее их ничего нет, рисунок 17.

Последнее действие процедуры DeleteRectangle – прибавление очков за ход к общим очкам и проверка окончания игры, рисунок 19.

Функция Gameover. Конец игры наступает тогда, когда подматрица 6x6, находящаяся в правом нижнем углу матрицы поля, не содержит хотя бы два рядом стоящих одинаковы кубика. Если кубиков вообще нет, значит поле полностью очищено(Win:=true).

Процедура ShowResult. Результаты прошлых игр хранятся в текстовых файлах в папке с игрой. Перед тем, как их показывать, нужно проверить со старыми результатами новый результат. В случае, если результат больше, мы вставляем строку в файл и измененный файл загружаем в окно результата.

function CheckPlace(X,Y:integer; ColorPlace:integer; var pole: mas; var Score,bonus,NumberDeleteRectangle: integer): boolean;

var ColorPlaceAround: integer;

begin

if ((X<>0) and (Y<>0))or((X<>21) and (Y<>21)) then

//Проверка на близко стоящие квадраты того же цвета.

begin

ColorPlaceAround:= pole[Y,X+1]; //Цвет кубика, стоящего справа.

if (ColorPlaceAround=ColorPlace) then //Если цвет совпадает.

begin

Score:=Score+1+bonus; //Увеличиваем очки

NumberDeleteRectangle:=NumberDeleteRectangle+1; //Длина цепочки увеличивается на 1. bonus:=bonus+1; //Увеличиваем бонус

pole[Y,X+1]:=0; //Удаляем цвет из матрицы поля.

CheckPlace(X+1,Y,ColorPlace,pole,Score,bonus,NumberDeleteRectangle); //Запускаем снова функцию CheckPlace и проверяем, есть ли кубики такого же цвета с найденным кубиком.

CheckPlace:=true; // Возвращаем значение true, которое говорит о том, что мы нашли хотя бы один кубик.

end;

<….>

end;

Рисунок 16 – Фрагмент функции удаления цепочки

for i := 20 downto NumberClearX do //Проверка, нужно ли сдвигать столбцы.

begin

if pole[20,i]=0 then //Проверяем нижний ряд

begin

b:=true; //Если в последней строке есть пустое место (0), то...

break;

end;

end;

if b=true then begin //..Сдвигаем столбы.

i:=20;

while i<>NumberClearX do begin

if pole[20,i]=0 then begin

MoveMassiveRight(i,NumberClearX,Image1,pole); //Сдвиг вправо.

for j := 0 to 20 do begin //Перерисовка изменившегося поля.

for k := 0 to i do begin

color:=SetOldColor(pole[j,k]);

DrawRectangle (j,k,color,Image1);

end; //Конец цикла k

end; //Конец цикла j

NumberClearX:=NumberClearX+1; //Увеличиваем количество пустых столбцов.

i:=i+1;

end; //Конец условия

i:=i-1;

end; //Конец цикла i

Рисунок 17 – Проверка и сдвиг целых столбцов при появлении пустоты между столбцами

procedure MoveMassiveRight(a,NumberClearX :integer; Image1: Timage; var pole: mas);

var i,j: integer;

begin

if a<>0 then begin

for i := a downto 1 do begin

for j := 0 to 20 do begin

pole[j,i]:=pole[j,i-1];

end; //Конец цикла j

end; //Конец цикла i

end;

for j := 0 to 20 do begin //Закрашиваем белым цветом столб, который образовался после заполнения пустого столба между «Горами кубиков».

pole[j,NumberClearX]:=0;

color:=clWhite;

DrawRectangle (j,0,NumberClearX,Image1);

end;

end;

Рисунок 18 – Сдвиг целого столбца

if NumberDeleteRectangle>MaxLongWay then //Максимальная длина цепочки.

begin

MaxLongWay:=NumberDeleteRectangle

end;

oldcolor:= SetOldColor(pole[Y,X]); //Новый цвет, на который указывает указатель.

Result(NumberDeleteRectangle,Score,Score1,); // Выведение результата хода на табло.

if Gameover(pole,Win)=false then //Конец игры.

begin

ShowResult(Score,MaxLongWay,Win,masRecord);

end;

Рисунок 19 – Окончание процедуры DeleteRectangle