Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ,

(национальный исследовательский университет)» (МАИ)

КУРСОВОЙ ПРОЕКТ

по дисциплине: «Вычислительные системы»

на тему «Разработка вычислительной системы   
«Doodle Jump»

**Выполнил:**  студент группы: М8О-111Б-17

Ломакина София Васильевна

(Фамилия, имя, отчество)

(подпись)

**Принял**: доцент кафедры 813

Мокряков Алексей Викторович

(Фамилия, имя, отчество)

(подпись)

**Оценка: Дата:**

МОСКВА 2017

Содержание

[Цель и задачи проекта 3](#_Toc502052053)

[Предлагаемые методы решения 3](#_Toc502052054)

[Реализация проекта 3](#_Toc502052055)

[Алгоритмы 3](#_Toc502052056)

[Первый ключевой алгоритм: 3](#_Toc502052057)

[Второй ключевой алгоритм: 4](#_Toc502052058)

[Третий ключевой алгоритм: 5](#_Toc502052059)

[Тестирование проекта 8](#_Toc502052060)

[Описание разработанной системы 8](#_Toc502052061)

[Руководство пользователя 9](#_Toc502052062)

[Список заявленных особенностей проекта 12](#_Toc502052063)

[Список литературы 12](#_Toc502052064)

[*Код программы* 13](#_Toc502052065)

## Цель и задачи проекта

Цель этого курсового проекта состояла в том, чтобы создать полноценную мобильную игру под названием “doodle jump”.

Задачи же заключались в том, чтобы адаптировать эту игру под компьютер. То есть: настроить управление персонажем с помощью кнопок клавиатуры, настройка игрового окна (экрана) и реализация работы с графикой.

## Предлагаемые методы решения

Поставленную задачу первым делом можно решить с помощью графики. Это один из главных элементов, так как с помощью графики можно организовать саму игру, которая, по сути, состоит из движущихся и управляемых изображений, и можно обрисовать меню для этой игры.

Вторым важным критерием является организация “кликабельности” функций в игре, благодаря которым процесс игры будет вообще возможен. То есть еще одним инструментом, с помощью которого будет совершаться игра, является клавиатура и компьютерная мышь.

## Реализация проекта

## **Алгоритмы**

### Первый ключевой алгоритм:

В этом алгоритме создается основное графическое окно, в котором и будет происходить игра. Далее обозначаются размеры этого окна по x и по y, а затем эти размеры фиксируются. После идет присваивание переменным типа picture адреса изображений, которые хранятся на диске.

window.Caption := 'Doodle\_Jump'; //создание окна

SetWindowSize(532, 850); //обозначение размера окна

window.IsFixedSize := true; //фиксация окна

LockDrawing; //ограничивает любые действия вне размера окна (по сути, тоже фиксация размеров окна)

OnKeyDown := KeyDown; //эти действия связаны с процедурами, о которых будет рассказано позже

OnKeyUp := KeyUp; //эти действия связаны с процедурами, о которых будет рассказано позже

OnMouseUp := MouseUp; //эти действия связаны с процедурами, о которых будет рассказано позже

//далее идет присваивание изображений

background := picture.Create('D:\\dudl\background.png');

platform := picture.Create('D:\\dudl\platform.png');

blueplatform := picture.Create('D:\\dudl\blueplatform.png');

wood1 := picture.Create('D:\\dudl\wood1.png');

one := picture.Create('D:\\dudl\one.png');

attack := picture.Create('D:\\dudl\attack.png');

### Второй ключевой алгоритм:

Если пропустить немалое количество процедур, создание разных переменных совершенно различных типов, то помимо создания окна игры, можно выделить два самых главных алгоритма. Первый из них это active. Этот алгоритм перемещает пользователя в игровое меню с кнопками, в котором можно изменить настройки (выбрать персонажа и сменить ник), посмотреть инструкцию к игре, выйти или начать саму игру. Меню прописано так, что перейти на предыдущее окно можно при помощи клавиши esc. Выйти из игры, начав ее, можно при помощи той же клавиши (но тогда набранный вами результат не сохранится). Также в этом алгоритме происходит создание/инициализация базы данных, в которой будет сохранятся максимально-набранный результат, имя игрока и номер выбранного вами персонажа. Еще в алгоритме происходит генерирование месторасположения различных платформ по x и по y, а также обозначение с помощью трех переменных середины игрового окна (для того, чтобы персонаж смог все время оставаться в середине, не передвигаясь вместе с экраном).

**while** active **do //начало цикла**

**begin**

assign(t, 'D:\\dudl\database'); //инициализация базы данных

reset(t);

**for** i := 1 **to** 5 **do** readln(t,Data[i]);

close(t);

//так как у пользователя должна быть возможность выбора персонажа, далее идет “сборная конструкция” адреса изображения персонажа. Каждое изображение оканчивается на свою цифру, и именно эта цифра и заисывается в базу данных, благодаря чему ее можно менять. Весь остальной адрес у всех изображений одинаковый

**var** doodpic1 := 'D:\\dudl\doodler';

**var** doodpic2 := Data[5];

**var** doodpic3 := '.png';

**var** doodpic := 'D:\\dudl\doodler1.png';

doodpic := doodpic1 + doodpic2 + doodpic3;

doodler := picture.Create(doodpic);

name := false; //эти действия связаны с процедурами, о которых будет рассказано позже

dood := false; //эти действия связаны с процедурами, о которых будет рассказано позже

OnMouseUp := MouseUp; //эти действия связаны с процедурами, о которых будет рассказано позже

ClearWindow;

x := 266;

y := 400;

h := 425;

//генерирование платформ

**for** i := 1 **to** 20 **do //обычные зеленые платформы**

**begin**

platforms[i].x := random(532);

platforms[i].y := random(850);

**end**;

**for** j := 1 **to** 7 **do //синие двигающиеся платформы**

**begin**

blueplatforms[j].xb := random(532);

blueplatforms[j].yb := random(850);

**end**;

**for** k := 1 **to** 5 **do //треснутая платформа**

**begin**

woods1[k].xc := random(532);

woods1[k].yc := random(850);

**end**;

**for** l := 1 **to** 3 **do //исчезающая платформа**

**begin**

ones[l].xd := random(532);

ones[l].yd := random(850);

**end**;

m := random(520); //координаты монстра по x

n := 0;

plat := 20; //кол-во зеленых платформ

platblue := 3; //кол-во синих платформ

woo := 5; //кол-во треснутых платформ

onn := 3; //кол-во исчезающих платформ

//далее идет “разукрашивание” меню в разные цвета и создание прямоугольников (кликабельных кнопок)

SetPenColor(clBlack);

SetFontColor(clBlack);

SetBrushColor(clTransparent);

setfontsize(10);

SetPenColor(clBlack);

SetBrushColor(clWhite);

setfontsize(20);

//это картинки в менюшке для атмосферы ☺

fon1:=picture.create('D:\\dudl\monstr.png');

fon1.Draw(5,340);

fon2:=picture.create('D:\\dudl\nlo.png');

fon2.Draw(10,20);

//картинки закончились, вот теперь идут кнопки

Rectangle(340, 30, 500, 80);

TextOut(360, 37, 'Начать');

Rectangle(340, 90, 500, 140);

TextOut(360, 97, 'Настройки');

Rectangle(340, 150, 500, 200);

TextOut(360, 157, 'Выход');

SetFontColor(clBlack);

setfontsize(30);

TextOut(360, 420, 'Рекорд:');

SetFontColor(clBlack);

TextOut(360, 465, Data[1]);

setfontsize(10);

SetBrushColor(clTransparent);

Redraw;

### Третий ключевой алгоритм:

Третий ключевой алгоритм это game. Это цикл самой игры, в которой и происходят действия с изображениями. Проблема в том, что этот цикл достаточно большой, и вставить его в документ полностью немного проблематично, поэтому я покажу повторяющиеся/зацикленные куски кода.

Начинается все с прорисовки всех сгенерированных ранее платформ и самого дудла. Некоторые платформы появляются лишь на определенной высоте, поэтому их циклы немного отличаются от других.

**while** game **do //начало цикла**

**begin**

background.Draw(0, 0); //вставка фона игры

//далее прорисовываются платформы

**for** i := 1 **to** plat **do //обычные зеленые платформы**

platform.Draw(platforms[i].x, platforms[i].y);

**for** j := 1 **to** platblue **do //синие двигающиеся платформы**

blueplatform.Draw(blueplatforms[j].xb, blueplatforms[j].yb);

**for** k := 1 **to** woo **do //треснутые платформы**

wood1.Draw(woods1[k].xc, woods1[k].yc);

**if** score >= 4000 **then //исчезающие платформы**

**//эти платформы появляются н высоте боле 4000**

**for** l := 1 **to** onn **do**

one.Draw(ones[l].xd, ones[l].yd);

doodler.Draw(x, y); //прорисовка дудлера

Далее идет организация действий с синими платформами. По сценарию, они должны безостановочно двигаться по горизонтали. Доходя до края экрана, они должны поворачивать в обратную сторону. Сделано это при помощи переменной pb. Это переменной изначально присвоено значение -1. При каждом прогоне цикла игры, к координате x синей платформы добавляется эта переменная, тем самым уменьшая значение на единицу и постепенно двигая платформу к нулю (то есть к левой стороне экрана). Как только координата x платформы становится почти равна нулю (<2), то значение этой переменной становится ее же значение, умноженное на -1. То есть переменная переходит из отрицательной в положительную (из -1 в 1). И далее к координате x платформы начинает добавляться уже измененная переменная, тем самым двигая платформу уже вправо. Те же действия повторяются, когда платформа почти доходит до правого края.

Все это прописано для всез 7 платформ отдельно, иначе действия воспринимались бы как для одной.

**if** (blueplatforms[1].xb < 2) **or** (blueplatforms[1].xb > 476) **then begin**

pb1 := pb1\*(-1);

**end**;

blueplatforms[1].xb := blueplatforms[1].xb + pb1;

blueplatform.Draw(blueplatforms[1].xb, blueplatforms[1].yb);

Чем больше высота, тем больше синих платформ.

Если рекорд побит, на экране во время игры будет висеть соответствующее уведомление

**if** score > sc **then** TextOut(10, 30, 'Побит рекорд!');

Поворот дудлера осуществляется при помощи процедуры и добавлению цифры 10 к номеру текущего дудлера. В папке с картинками можно найти дудлера как с номером 1, так и с номером 110.

**if** Left = 1 **then begin**

x := x - 4;

**var** doodpic10 := 'D:\\dudl\doodler';

**var** doodpic20 := Data[5] + 10;

**var** doodpic30 := '.png';

**var** doodpic0 := 'D:\\dudl\doodler1.png';

doodpic := doodpic10 + doodpic20 + doodpic30;

doodler := picture.Create(doodpic);

**end**;

При нажатии пробела дудлер стреляет бумажным шариком. При этом картинка дудлера меняется в течении всего времени, что нажат пробел.

**if** space = 1 **then begin**

doodler := picture.Create('D:\\dudl\doodv.png');

a := x+22; b := y;

attack.Draw(a,b);

**end**;

Дудлер может умереть от монстра. Это произойдет, если их координаты по x и по y совпадут. На этом игра закончится. Также дудлер может прыгнуть на монстра сверху. Если это произойдет, то дудлер отпрыгнет от него, как от обычной платформы, а монстр исчезнет (исчезнет он путем добавления к его координате y значения 1000, тем самым отправляя его вниз). Монстр также исчезнет, если его координаты совпадут с координатами бумажного шарика при атаке дудлера.

**if** (a > m) **and**

(a < m + 56) **and**

(b > n) **and**

(b < n + 56) **and** (vy > 0) **then begin**

n:=1000;

monster := picture.create('D:\\dudl\monster.png');

monster.Draw(m,n);

**end**;

**if** (x + 36 > m) **and**

(x + 5 < m + 56) **and**

(y + 53 > n) **and**

(y + 53 < n + 56) **and** (vy > 0) **then begin**

vy := vminus;

n:=1000;

monster := picture.create('D:\\dudl\monster.png');

monster.Draw(m,n);

**end**;

**if** (x + 36 > m) **and**

(x + 5 < m + 56) **and**

(y > n) **and**

(y < n + 56) **then begin**

game := false;

gameover := true;

**break**

**end**;

В цикле прописано отталкивание дудлера от платформ (для каждой платформы отдельно). Если координаты дудлера в определенных точках совпадают с координатами платформы, то координате ускорения дудлера по оси y (vy) добавляется значение vminus, которое является фиксированным и которому присвоено значение -4 перед самим циклом игры.

**for** i := 1 **to** plat **do**

**if** (x + 36 > platforms[i].x) **and**

(x + 5 < platforms[i].x + 56) **and**

(y + 53 > platforms[i].y) **and**

(y + 53 < platforms[i].y + 14) **and** (vy > 0) **then**

vy := vminus;

Также в цикле прописан оператор, благодаря которому платформы не остаются на месте, а едут вниз вместе с экраном. Суть в чем: если координата y дудлера становится меньше середины игрового экрана (то есть дудлер движется вверх), то координате дудлера присваивается значение середины экрана, а из значения платформы вычитается то расстояние, которое дудлер успел пройти при своем ускорении. Все это происходит плавно, так как цикл повторяется много раз.

Если координата платформы уходит ниже экрана (>850), то генерируется новая платформа в самом начале оси y, а вот координата x определяется рандомом.

**if** y < h **then**

**begin**

**for** i := 1 **to** plat **do**

**begin**

y := h;

platforms[i].y := platforms[i].y - round(vy);

**if** platforms[i].y > 850 **then**

**begin**

platforms[i].y := 0;

platforms[i].x := random(532);

**end**;

**end**;

Чем больше высота, тем меньше кол-во зеленых (обычных) платформ. Также при высоте более 10000 скорость дудлера значительно увеличивается.

**if** score >= 2000 **then** plat := 19;

**if** score >= 2500 **then** plat := 18;

**if** score >= 3000 **then** plat := 17;

**if** score >= 3500 **then** plat := 16;

**if** score >= 4000 **then** plat := 15;

**if** score >= 4500 **then** plat := 14;

**if** score >= 5000 **then** plat := 13;

**if** score >= 10000 **then**

**begin**

vplus := 0.2;

vminus := -10;

**end**;

Игра заканчивается если координата y дудлера становится больше границ экрана (>850), то есть он выходит за его пределы.

**if** y > 850 **then**

**begin**

game := false;

gameover := true;

**break**

**end**;

## **Тестирование проекта**

Тестирование происходило много раз. Первым делом была проверена кликабельность игровых кнопок и кнопок, представленных в меню. Также проверялась возможность выти на предыдущий экран при помощи клавиши esc.

Проверилась возможность управления дудлом при помощи клавиатурных стрелок и возможности атаки при нажатии пробела.

Проверена была и возможность менять ник в главном меню, а также возможность смены внешнего вида дудла, путем ввода нужной цыфры.

Тесты платформ проводились в процессе написания игры.

## **Описание разработанной системы**

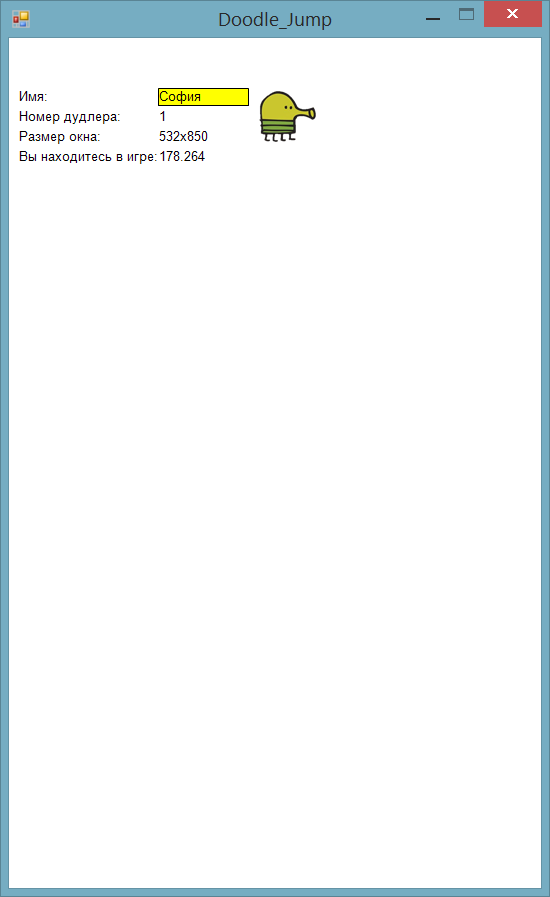
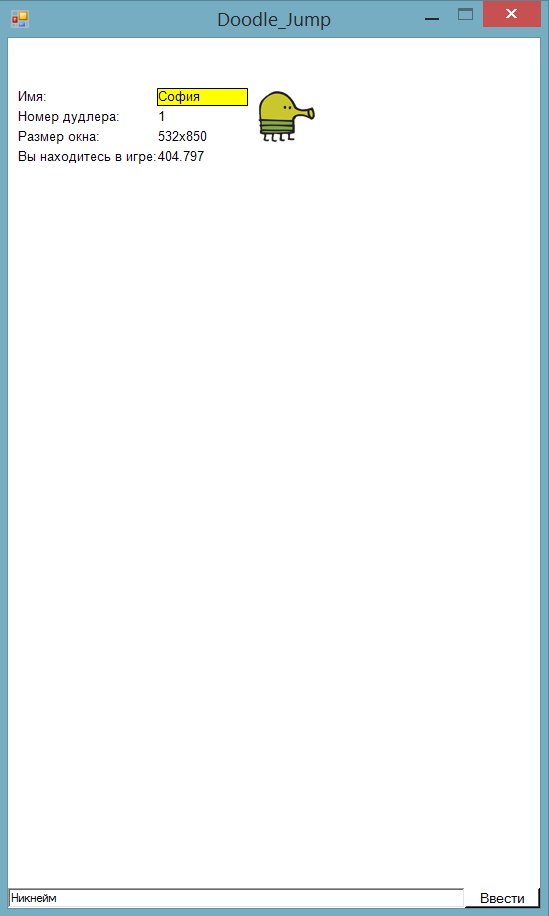
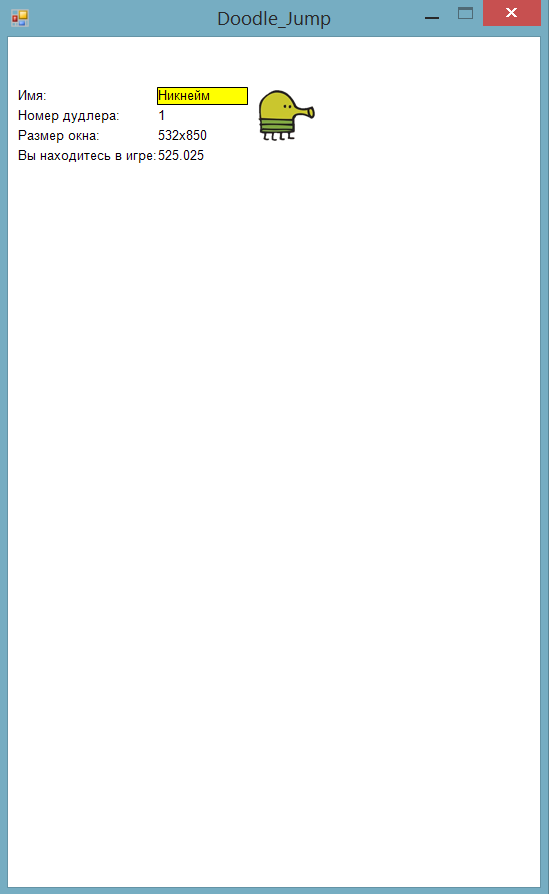
## **Руководство пользователя**

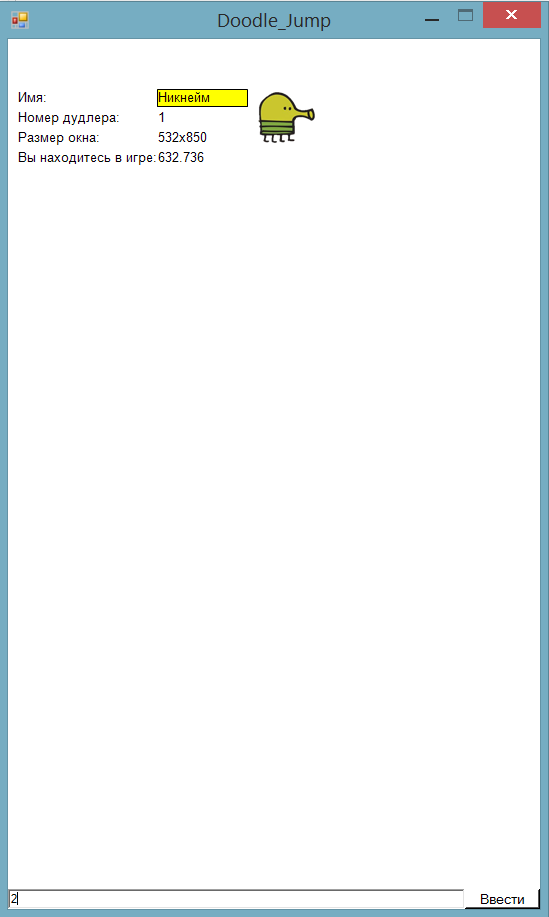
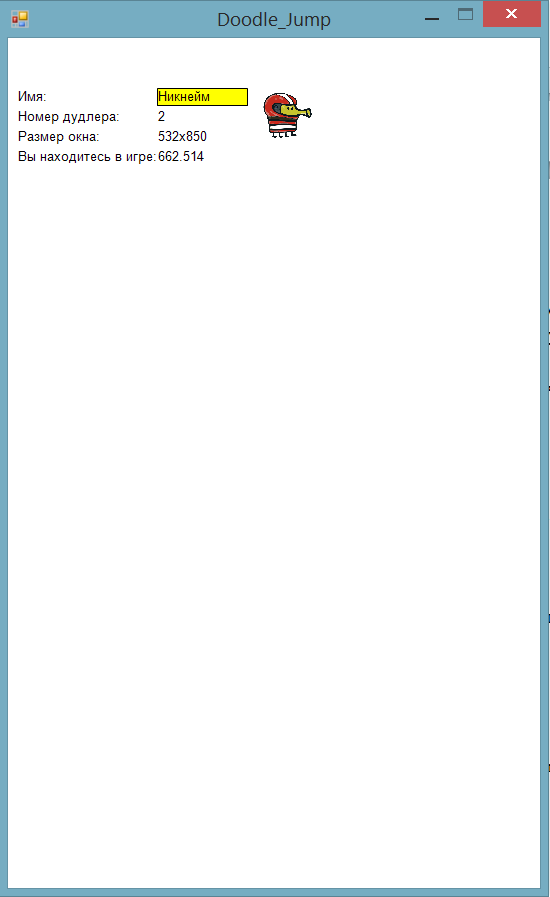
При запуске программы пользователь попадает в главное меню:



У пользователя есть три варианта: начать игру, зайти в настройки или выйти. При выходе игра просто закроется. Начав игру, можно будет вернуться клавишей esc.

Для начала демонстрация настроек. У пользователя есть возможность сменить ник или дудла, просто нажав на желтое окошко, либо нажав на изображение дудла.

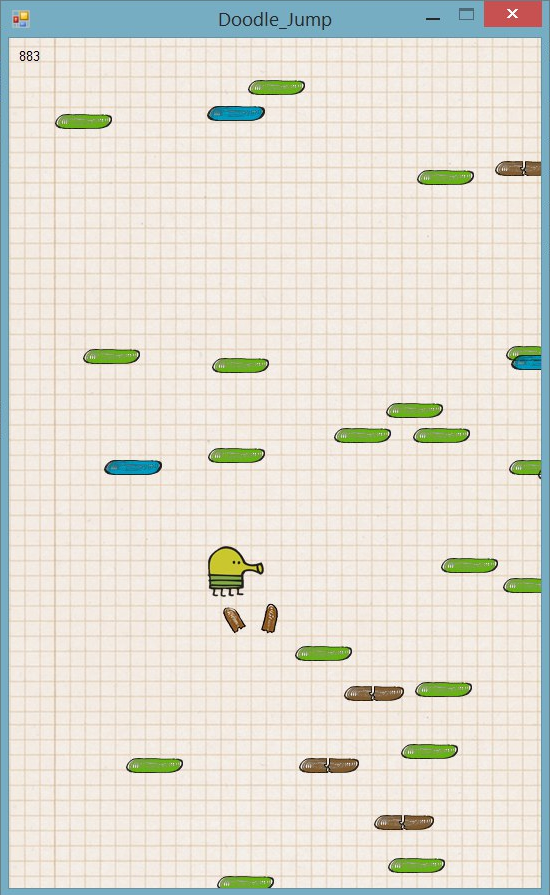
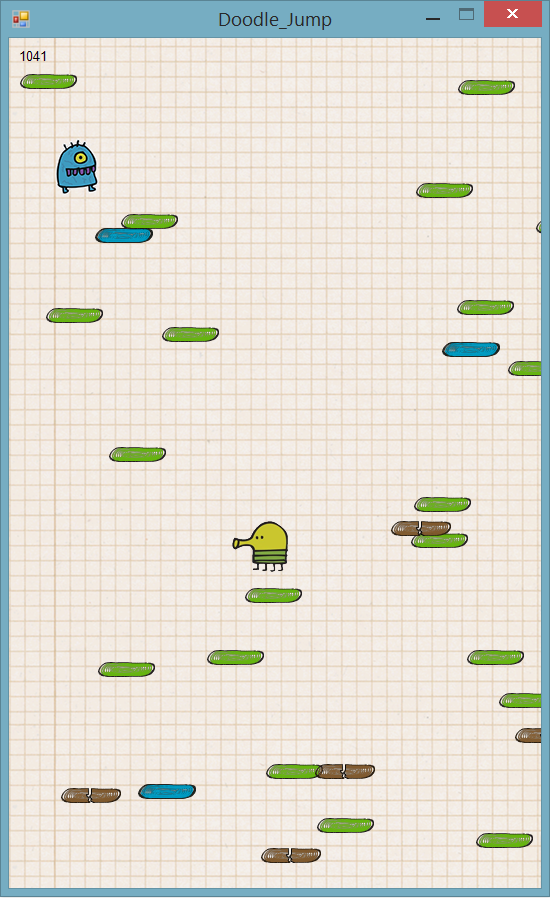
 

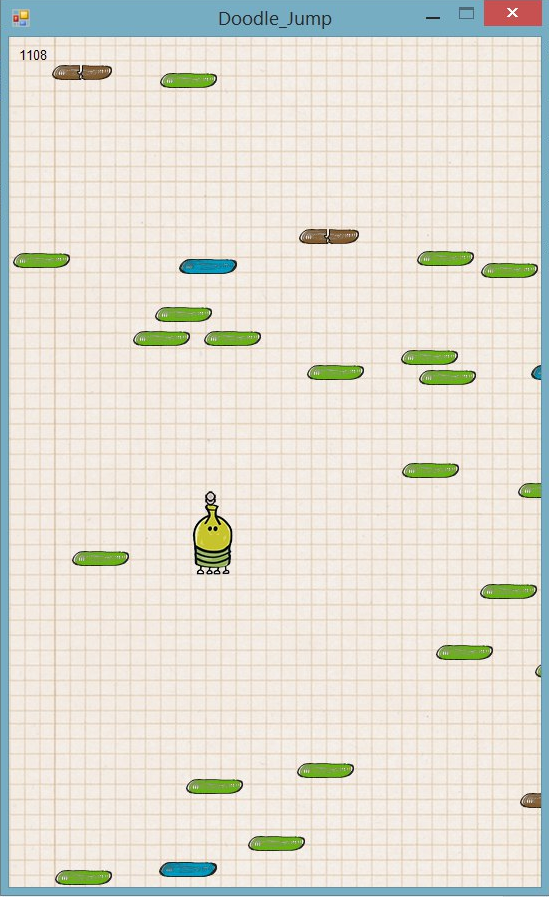
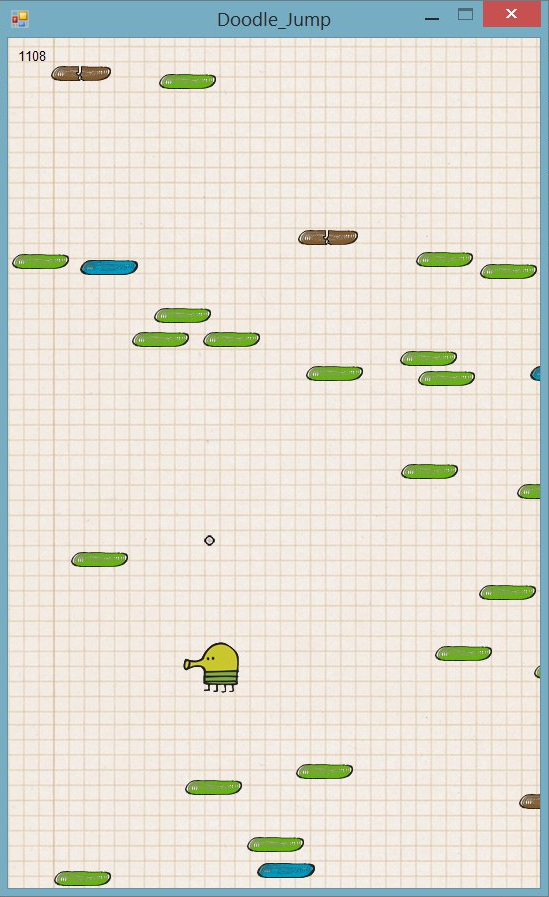
После нажатия внизу появится окошко ввода. Для смены имени можно ввести любой набор символов или букв. Для смены дудла нужно ввести цифру (цифра не должна превышать значение кол-ва скинов на дудла). После ввода можно либо нажать кнопку ввести, либо enter. Никнейм сразу появится в окошке, а для смены изображения дудла надо будет выйти с помощью esc и потом зайти обратно.

Игра начинается с обычных, треснутых и синих платформ. Управление стрелками на клавиатуре. Атака на пробел.

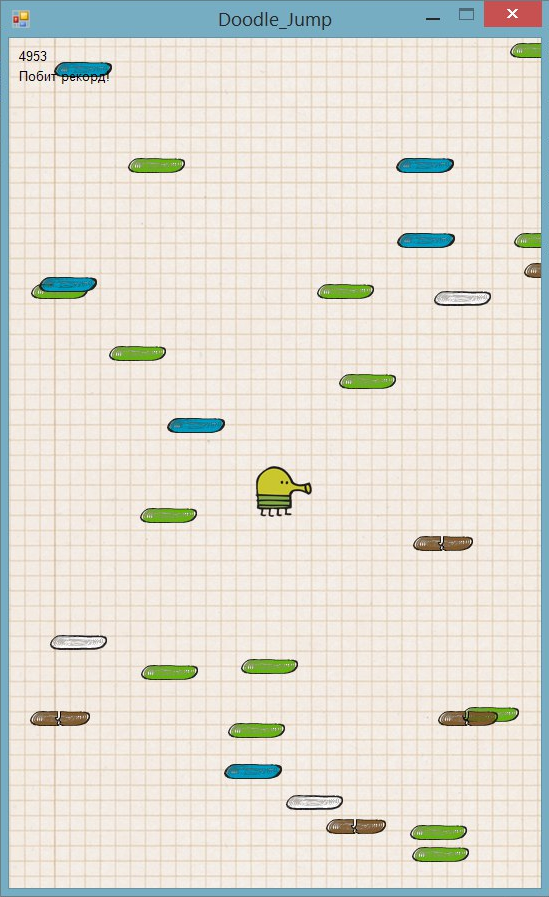
До монстров лучше не дотрагиваться, они убивают. Можно прыгнуть на него сверху, либо пульнуть в него шариком. Оба способа убивают монстра.

Треснутые платформы тоже могут убить. Прыгнув на нее, дудл не отталкивается, а проваливается, платформа ломается и летит вниз.

Чем выше, тем меньше зеленых платформ, больше синих и на определенной высоте появятся белые (исчезающие платформы). Также, если предыдущий рекорд будет побит, то появится соответствующая надпись.



## **Список заявленных особенностей проекта**

В отличие от созданных ранее другими людьми версий doodle jump на компютере, эта отличается привычным форматом игрового экрана (как на мобильной версии), разнообразием платформ, возможностью атаковать монстров, ускорением на определенной высоте, возможностью менять внешний вид дудла. Еще, при нажатии разных клавиш разных сторон, дудл смотрит в разные стороны. При атаке анимация дудла тоже меняется.

Единственная проблема, которая так и не была решена, это бонусы в виде джетпака и шапки.

## **Список литературы**

[1] <http://pascalabc.net/stati-po-pascalabc-net>

[2] <http://pas1.ru/graphical>

[3] Л.И. Долинер “Основы программирования в среде PascalABC.NET”

<http://elar.urfu.ru/bitstream/10995/28702/1/978-5-7996-1260-3_2014.pdf>

## Код программы

**uses** GraphABC;

**type** point = **record**

x, y: integer;

**end**;

**type** pointb = **record**

xb, yb: integer;

**end**;

**type** pointc = **record**

xc, yc: integer;

**end**;

**type** pointd = **record**

xd, yd: integer;

**end**;

**type** pointe = **record**

xe, ye: integer;

**end**;

**var**

doodler, background, platform, blueplatform, fon1, fon2, wood1, wood2, one, attack, monster, hole: picture;

Left, Right, space, x, y, xb, yb, xc, yc, xd, yd, a, b, i, j, k, l, m, n, h, mns, score, mnst, sc, err, plat, platblue, woo, onn, pb1, pb2, pb3, pb4, pb5, pb6, pb7, vminus: integer;

vx, vy, vplus: real;

platforms: **array** [1..20] **of** point;

blueplatforms: **array** [1..7] **of** pointb;

woods1: **array** [1..5] **of** pointc;

woods2: **array** [1..5] **of** pointe;

ones: **array** [1..3] **of** pointd;

game, active, settings, gameover, name, dood, thanks: boolean;

t : text;

Data: **array**[1..5] **of** string;

**procedure** KeyDown(Key: integer);

**begin**

**if** Key = vk\_Left **then** Left := 1;

**if** Key = vk\_Right **then** Right := 1;

**if** key = vk\_Escape **then** game := false;

**if** key = vk\_Escape **then** settings := false;

**if** key = vk\_Escape **then** gameover := false;

**if** Key = vk\_Space **then** space := 1;

**end**;

**procedure** KeyUp(Key: integer);

**begin**

**if** Key = vk\_Left **then** Left := 0;

**if** Key = vk\_Right **then** Right := 0;

**if** Key = vk\_Space **then** space := 0;

**end**;

**procedure** MouseUp(x, y, mb: integer);

**begin**

**if** (mb = 1) **and** (x > 340) **and** (x < 500) **and** (y > 30) **and** (y < 80) **then** game := true;

**if** (mb = 1) **and** (x > 340) **and** (x < 500) **and** (y > 90) **and** (y < 140) **then** settings := true;

**if** (mb = 1) **and** (x > 340) **and** (x < 500) **and** (y > 150) **and** (y < 200) **then** active := false;

**if** (mb = 1) **and** (x > 100) **and** (x < 300) **and** (y > 133) **and** (y < 163) **and** (gameover = true) **then begin** game := true; gameover := false; **end**;

**end**;

**procedure** MouseUpSettings(x, y, mb: integer);

**begin**

**if** (mb = 1) **and** (x > 149) **and** (x < 240) **and** (y > 50) **and** (y < 68) **then** name := true;

**if** (mb = 1) **and** (x > 250) **and** (x < 357) **and** (y > 50) **and** (y < 107) **then** dood := true;

**end**;

**begin**

window.Caption := 'Doodle\_Jump';

SetWindowSize(532, 850);

window.IsFixedSize := true;

LockDrawing;

OnKeyDown := KeyDown;

OnKeyUp := KeyUp;

OnMouseUp := MouseUp;

background := picture.Create('D:\\dudl\background.png');

platform := picture.Create('D:\\dudl\platform.png');

blueplatform := picture.Create('D:\\dudl\blueplatform.png');

wood1 := picture.Create('D:\\dudl\wood1.png');

one := picture.Create('D:\\dudl\one.png');

attack := picture.Create('D:\\dudl\attack.png');

active := true;

game := false;

gameover := false;

**while** active **do**

**begin**

assign(t, 'D:\\dudl\database');

reset(t);

**for** i := 1 **to** 5 **do** readln(t,Data[i]);

close(t);

**var** doodpic1 := 'D:\\dudl\doodler';

**var** doodpic2 := Data[5];

**var** doodpic3 := '.png';

**var** doodpic := 'D:\\dudl\doodler1.png';

doodpic := doodpic1 + doodpic2 + doodpic3;

doodler := picture.Create(doodpic);

name := false;

dood := false;

OnMouseUp := MouseUp;

ClearWindow;

x := 266;

y := 400;

h := 425;

**for** i := 1 **to** 20 **do**

**begin**

platforms[i].x := random(532);

platforms[i].y := random(850);

**end**;

**for** j := 1 **to** 7 **do**

**begin**

blueplatforms[j].xb := random(532);

blueplatforms[j].yb := random(850);

**end**;

**for** k := 1 **to** 5 **do**

**begin**

woods1[k].xc := random(532);

woods1[k].yc := random(850);

**end**;

**for** l := 1 **to** 3 **do**

**begin**

ones[l].xd := random(532);

ones[l].yd := random(850);

**end**;

m := random(520);

n := 0;

plat := 20; //кол-во платформ

platblue := 3;

woo := 5;

onn := 3;

mns := 1;

SetPenColor(clBlack);

SetFontColor(clBlack);

SetBrushColor(clTransparent);

setfontsize(10);

SetPenColor(clBlack);

SetBrushColor(clWhite);

setfontsize(20);

fon1:=picture.create('D:\\dudl\monstr.png');

fon1.Draw(5,340);

fon2:=picture.create('D:\\dudl\nlo.png');

fon2.Draw(10,20);

Rectangle(340, 30, 500, 80);

TextOut(360, 37, 'Начать');

Rectangle(340, 90, 500, 140);

TextOut(360, 97, 'Настройки');

Rectangle(340, 150, 500, 200);

TextOut(360, 157, 'Выход');

SetFontColor(clBlack);

setfontsize(30);

TextOut(360, 420, 'Рекорд:');

SetFontColor(clBlack);

TextOut(360, 465, Data[1]);

setfontsize(10);

SetBrushColor(clTransparent);

Redraw;

//--------------------------- ЦИКЛ НАСТРОЕК

**while** settings **do**

**begin**

ClearWindow;

OnMouseUp := MouseUpSettings;

SetPenColor(clBlack);

SetFontColor(clBlack);

SetBrushColor(clTransparent);

SetBrushColor(clTransparent);

SetFontColor(clYellow);

SetBrushColor(clYellow);

Rectangle(149, 50, 240, 68);

SetBrushColor(clTransparent);

SetFontColor(clBlack);

TextOut(10, 50, 'Имя: ');

SetFontColor(clBlack);

TextOut(150, 50, Data[2]);

SetFontColor(clBlack);

TextOut(10, 70, 'Номер дудлера: ');

SetFontColor(clBlack);

TextOut(150, 70, Data[5]);

picture.create(doodpic);

doodler.draw(250, 50);

SetFontColor(clBlack);

TextOut(10, 90, 'Размер окна: ');

SetFontColor(clBlack);

TextOut(150, 90, '532x850');

SetFontColor(clBlack);

TextOut(10, 110, 'Вы находитесь в игре: ');

SetFontColor(clBlack);

TextOut(150, 110, (Milliseconds / 1000));

Redraw;

**if** name **then**

**begin**

readln(Data[2]);

name := false;

**end**;

**if** dood **then**

**begin**

readln(Data[5]);

dood := false;

**end**;

**end**;

score := 0;

vplus := 0.1;

vminus := -7;

pb1 := -1; pb2 := -1; pb3 := -1; pb4 := -1; pb5 := -1; pb6 := -1; pb7 := -1;

//--------------------------- ПРОЦЕСС ИГРЫ

**while** game **do**

**begin**

background.Draw(0, 0);

**for** i := 1 **to** plat **do**

platform.Draw(platforms[i].x, platforms[i].y);

**for** j := 1 **to** platblue **do**

blueplatform.Draw(blueplatforms[j].xb, blueplatforms[j].yb);

**for** k := 1 **to** woo **do**

wood1.Draw(woods1[k].xc, woods1[k].yc);

**if** score >= 4000 **then**

**for** l := 1 **to** onn **do**

one.Draw(ones[l].xd, ones[l].yd);

doodler.Draw(x, y);

**if** (blueplatforms[1].xb < 2) **or** (blueplatforms[1].xb > 476) **then**

**begin**

pb1 := pb1\*(-1);

**end**;

blueplatforms[1].xb := blueplatforms[1].xb + pb1;

blueplatform.Draw(blueplatforms[1].xb, blueplatforms[1].yb);

**if** (blueplatforms[2].xb < 2) **or** (blueplatforms[2].xb > 476) **then**

**begin**

pb2 := pb2\*(-1);

**end**;

blueplatforms[2].xb := blueplatforms[2].xb + pb2;

blueplatform.Draw(blueplatforms[2].xb, blueplatforms[2].yb);

**if** (blueplatforms[3].xb < 2) **or** (blueplatforms[3].xb > 476) **then**

**begin**

pb3 := pb3\*(-1);

**end**;

blueplatforms[3].xb := blueplatforms[3].xb + pb3;

blueplatform.Draw(blueplatforms[3].xb, blueplatforms[3].yb);

**if** score >= 2000 **then**

**begin**

platblue := 4;

**if** (blueplatforms[4].xb < 2) **or** (blueplatforms[4].xb > 476) **then**

**begin**

pb4 := pb4\*(-1);

**end**;

blueplatforms[4].xb := blueplatforms[1].xb + pb4;

blueplatform.Draw(blueplatforms[4].xb, blueplatforms[4].yb);

**end**;

**if** score >= 3200 **then**

**begin**

platblue := 5;

**if** (blueplatforms[5].xb < 2) **or** (blueplatforms[5].xb > 476) **then**

**begin**

pb5 := pb5\*(-1);

**end**;

blueplatforms[5].xb := blueplatforms[5].xb + pb5;

blueplatform.Draw(blueplatforms[5].xb, blueplatforms[5].yb);

**end**;

**if** score >= 4500 **then**

**begin**

platblue := 6;

**if** (blueplatforms[6].xb < 2) **or** (blueplatforms[6].xb > 476) **then**

**begin**

pb6 := pb6\*(-1);

**end**;

blueplatforms[6].xb := blueplatforms[6].xb + pb6;

blueplatform.Draw(blueplatforms[6].xb, blueplatforms[6].yb);

**end**;

**if** score >= 5100 **then**

**begin**

platblue := 7;

**if** (blueplatforms[7].xb < 2) **or** (blueplatforms[7].xb > 476) **then**

**begin**

pb7 := pb7\*(-1);

**end**;

blueplatforms[7].xb := blueplatforms[7].xb + pb7;

blueplatform.Draw(blueplatforms[7].xb, blueplatforms[7].yb);

**end**;

TextOut(10, 10, score.ToString);

val(Data[1], sc, err);

SetBrushColor(clTransparent);

SetPenColor(clBlack);

**if** score > sc **then** TextOut(10, 30, 'Побит рекорд!');

**if** Left = 1 **then**

**begin**

x := x - 4;

**var** doodpic10 := 'D:\\dudl\doodler';

**var** doodpic20 := Data[5] + 10;

**var** doodpic30 := '.png';

**var** doodpic0 := 'D:\\dudl\doodler1.png';

doodpic := doodpic10 + doodpic20 + doodpic30;

doodler := picture.Create(doodpic);

**end**;

**if** Right = 1 **then**

**begin**

x := x + 4;

**var** doodpic10 := 'D:\\dudl\doodler';

**var** doodpic20 := Data[5];

**var** doodpic30 := '.png';

**var** doodpic0 := 'D:\\dudl\doodler1.png';

doodpic := doodpic10 + doodpic20 + doodpic30;

doodler := picture.Create(doodpic);

**end**;

**if** space = 1 **then**

**begin**

doodler := picture.Create('D:\\dudl\doodv.png');

a := x+22; b := y;

attack.Draw(a,b);

**end**;

**if** space = 0 **then**

doodler := picture.Create(doodpic);

b := b-4;

attack.Draw(a,b);

**if** (a > m) **and**

(a < m + 56) **and**

(b > n) **and**

(b < n + 56) **and** (vy > 0) **then**

**begin**

n:=1000;

monster := picture.create('D:\\dudl\monster.png');

monster.Draw(m,n);

**end**;

**if** (x + 36 > m) **and**

(x + 5 < m + 56) **and**

(y + 53 > n) **and**

(y + 53 < n + 56) **and** (vy > 0) **then**

**begin**

vy := vminus;

n:=1000;

monster := picture.create('D:\\dudl\monster.png');

monster.Draw(m,n);

**end**;

**if** (x + 36 > m) **and**

(x + 5 < m + 56) **and**

(y > n) **and**

(y < n + 56) **then**

**begin**

game := false;

gameover := true;

**break**

**end**;

vy := vy + vplus;

y := y + round(vy);

**if** score > 1000 **then**

**begin**

monster := picture.create('D:\\dudl\monster.png');

monster.Draw(m,n);

**if** y < h **then**

n := n - round(vy);

**if** round(vy) > 0 **then**

n := n;

**end**;

**for** i := 1 **to** plat **do**

**if** (x + 36 > platforms[i].x) **and**

(x + 5 < platforms[i].x + 56) **and**

(y + 53 > platforms[i].y) **and**

(y + 53 < platforms[i].y + 14) **and** (vy > 0) **then**

vy := vminus;

**for** j := 1 **to** platblue **do**

**if** (x + 36 > blueplatforms[j].xb) **and**

(x + 5 < blueplatforms[j].xb + 56) **and**

(y + 53 > blueplatforms[j].yb) **and**

(y + 53 < blueplatforms[j].yb + 14) **and** (vy > 0) **then**

vy := vminus;

**for** k := 1 **to** woo **do**

**if** (x + 36 > woods1[k].xc) **and**

(x + 5 < woods1[k].xc + 56) **and**

(y + 53 > woods1[k].yc) **and**

(y + 53 < woods1[k].yc + 14) **and** (vy > 0) **then**

**begin**

wood2 := picture.create('D:\\dudl\wood2.png');

woods2[1].xe := woods1[k].xc;

woods2[1].ye := woods1[k].yc;

wood2.Draw(woods2[1].xe,woods2[1].ye);

woods1[k].yc := woods1[k].yc + 1000;

wood1.Draw(woods1[k].xc,woods1[k].yc);

**end**;

woods2[1].ye := woods2[1].ye + 5;

wood2:=picture.create('D:\\dudl\wood2.png');

wood2.Draw(woods2[1].xe,woods2[1].ye);

**for** l := 1 **to** onn **do**

**if** (x + 36 > ones[l].xd) **and**

(x + 5 < ones[l].xd + 56) **and**

(y + 53 > ones[l].yd) **and**

(y + 53 < ones[l].yd + 14) **and** (vy > 0) **then**

**begin**

vy := vminus;

**while** ones[l].yd <> 850 **do**

ones[l].yd := ones[l].yd + 1;

one.Draw(ones[l].xd, ones[l].yd);

**end**;

**if** y < h **then**

**begin**

**for** i := 1 **to** plat **do**

**begin**

y := h;

platforms[i].y := platforms[i].y - round(vy);

**if** platforms[i].y > 850 **then**

**begin**

platforms[i].y := 0;

platforms[i].x := random(532);

**end**;

**end**;

**for** j := 1 **to** platblue **do**

**begin**

yb := h;

blueplatforms[j].yb := blueplatforms[j].yb - round(vy);

**if** blueplatforms[j].yb > 850 **then**

**begin**

blueplatforms[j].yb := 0;

blueplatforms[j].xb := random(532);

**end**;

**end**;

**for** k := 1 **to** woo **do**

**begin**

yc := h;

woods1[k].yc := woods1[k].yc - round(vy);

**if** woods1[k].yc > 850 **then**

**begin**

woods1[k].yc := 0;

woods1[k].xc := random(532);

**end**;

**end**;

**for** l := 1 **to** onn **do**

**begin**

yd := h;

ones[l].yd := ones[l].yd - round(vy);

**if** ones[l].yd > 850 **then**

**begin**

ones[l].yd := 0;

ones[l].xd := random(532);

**end**;

**end**;

score := score + 1;

**end**;

**if** y > 850 **then**

**begin**

game := false;

gameover := true;

**break**

**end**;

**if** score >= 2000 **then** plat := 19;

**if** score >= 2500 **then** plat := 18;

**if** score >= 3000 **then** plat := 17;

**if** score >= 3500 **then** plat := 16;

**if** score >= 4000 **then** plat := 15;

**if** score >= 4500 **then** plat := 14;

**if** score >= 5000 **then** plat := 13;

**if** score >= 10000 **then**

**begin**

vplus := 0.2;

vminus := -10;

**end**;

Redraw;

**end**;

**while** gameover **do**

**begin**

ClearWindow;

OnMouseUp := MouseUp;

background.Draw(0, 0);

SetBrushColor(clTransparent);

TextOut(10, 25, 'Игра закончена');

SetBrushColor(clYellowGreen);

Rectangle(100, 133, 300, 163);

TextOut(180, 140, 'Снова?');

SetBrushColor(clYellow);

Rectangle(100, 190, 300, 220);

TextOut(150, 197, 'Твой счет: ');

TextOut(240, 197, score.ToString);

vy := 0;

val(Data[1], sc, err);

SetBrushColor(clTransparent);

SetPenColor(clBlack);

**if** score > sc **then**

TextOut(100, 230, 'Новый рекорд!');

Redraw;

**end**;

**if** score > sc **then**

Data[1] := score.ToString;

assign(t,'D:\\dudl\database');

rewrite(t);

**for** i := 1 **to** 5 **do**

writeln(t, Data[i]);close(t);

**end**;

Halt;

**end**.