Лабораторная работа № ##
Таймер (Timer)

Часть 1. Бильярдный шар

По форме должен перемещаться круг, отскакивая от стенок формы. Требуется попасть мышкой в круг, если попадание удачно – круг останавливается.

Для осуществления нужно объявить переменный X1, Y1 – центр перемещающегося круга. Само перемещение описывается в процедуре **Timer1Timer**.

Алгоритм следующий:

1. В процедуре по созданию формы (событие **Create**) задаем начальные координаты круга на форме и величины смещения центра круга по оси X и Y(например dx:=+5; и dy:=+5;). В этой же процедуре делаем первую прорисовку круга (с помощью метода **Canvas.Ellipse**)
2. Создаем процедуру таймера **Timer1Timer.** Эта процедура выполнится первый раз через 1000 мс (по умолч) после создания формы.
3. В этой процедуре изменяем цвет карандаша на цвет формы и прорисовываем круг этим цветом в центре в т.X1,Y1. (круг исчез)
4. Далее изменяем координаты центра X1 и Y1 на величины dx и dy (X1:=X1+dx; и т.д.).
5. Меняем цвет карандаша на черный и прорисовываем круг в центре в т.X1,Y1
6. (далее процедура **Timer1Timer** выполняется снова через 1000 мс и шаги 3-5 повторяются, круг движется)

Чтобы круг отражался от стенок нужно учесть, когда круг достигнет стенок (краев формы) и поменять смещение dx или dy в зависимости от того в какую стенку ударился круг.

Например:

Если смещение dx = 5 и dy = 5, т.е. круг движется по диагноали (вниз и вправо), круг достигает правой стенки формы, т.е. если величина X1 + R >= Form1.Width , то теперь он должен двигаться влево, значит меняем знак смещения dx, теперь оно не +5, а **-5**. (меняем знак так **dx:= -dx**). Аналогично для остальных трех стенок. Итого получается 4 условных оператора, все они должны быть внутри процедуры таймера, после прорисовки круга.

Итак круг движется, остается только описать процедуру **FormMouseUp** (событие **onMouseUp**) в которой есть параметры X и Y, которые принимают значение координаты клика мыши. С помощью них и нужно определить попали ли мы в круг или нет. Если попали, то таймер нужно остановить.

Часть 2. Траэктория тела, брошенного под углом к горизонту

На форме изображены координатные оси, мы должны задать цвет траектории, а также угол и нач. скорость бросания тела. Затем щелкаем мышью, где-то на форме и оттуда рисуется траектория заданным цветом в зависимости от начальных параметров. Путь полета и максимальная высота полета высчитывается и выводится в соответствующие поля. (последним этапом разработки проекта будет прорисовка траектории не сразу, а медленно, по таймеру). В заголовок формы выведите координаты текущего положения мыши на форме, в соответствии с приведенной координатной системой (см. рис.)

$$x1 = x0 + V_0 \cdot t \cdot \cos\alpha$$

$$y1 = y0 + V_0 \cdot t \cdot \sin\alpha - \frac{g \cdot t^2}{2} \qquad h_{\max} = \frac{V_0^2 \cdot \sin^2\alpha}{2 \cdot g}$$

Вообще, высоту и дальность полета можно посчитать запомнив начальную координату точки броска и сравнив с точкой в момент пересечения с осью X.

Вначале оформите форму как на рисунке, расположите необходимые объекты (таймер пока добавлять не обязательно). Нарисовать оси нужно в процедуре создания формы **FormPaint** (событие **onPaint**).

Шаг 1

Для начала нужно сделать чтобы график чертился сразу по нажатию на кнопку мыши (в процедуре **FormMouseUp**).

Начало процедуры выглядит примерно так:

Canvas.Pen.Color:=ColorBox1.Selected; *{цвет графика – выбранный цвет в ColorBox'е}*

```
Canvas.Pen.Width:=3;
Canvas.MoveTo(X,Y);     {переводим графический курсор в точку клика мыши}
t:=0;          {время, с течением времени изменяется положение тела}
v:=StrToFloat(Edit2.Text);        {СКОРОСТЬ}
g:=10;                            {УСКОРЕНИЕ СВ. ПАДЕНИЯ}
dt:=0.01;                         {ШАГ для параметра t}
a:=StrToFloat(Edit1.Text)*(3.14/180);  {угол броска, перевод в радианы}
x1:=X;                            {}
y1:=Y;                            {начальная точка из которой бросаем}
```

Шаг 2

Далее в цикле, чертим график (цикл заканчивается когда «тело упало», т.е. когда координата x1 и y1 достигнет оси OX)

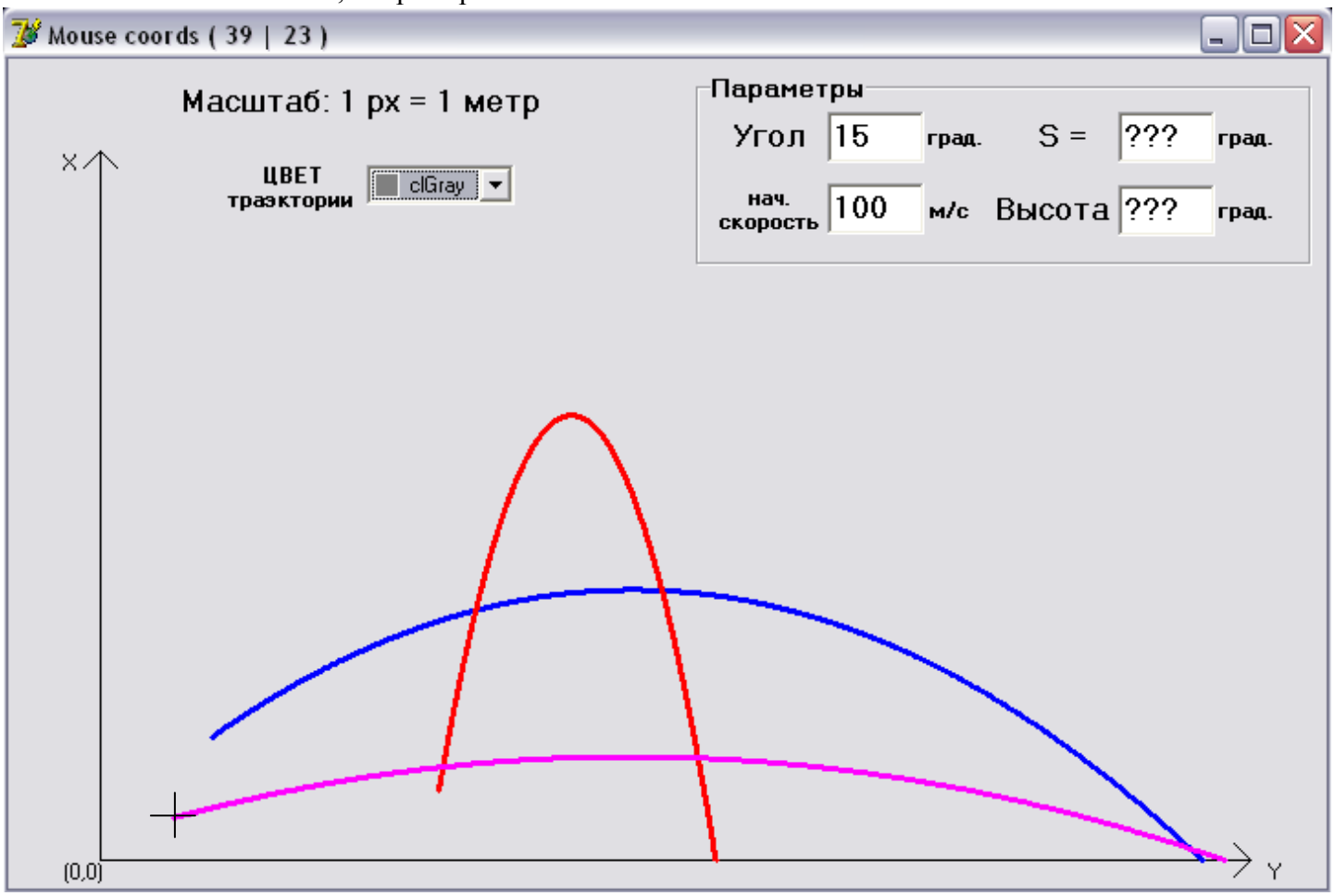
```
x1:=X+v*cos(a)*t;
y1:=Y-v*sin(a)*t+(g*t*t/2);
LineTo(round(x1),round(y1));          {чертим линию до вновьвычисленной координаты}
MoveTo(round(x1),round(y1));          {перемещаем графический курсор в эту точку, чтобы
потом ОТ НЕЁ ЧЕРТИТЬ СЛЕДУЮЩУЮ ЛИНИЮ графика }
    t:=t+dt;          {увеличиваем параметр t}
```

Шаг 3

Вычисляем высоту и дальность полета и выводим на форму. Только если все работает, приступаем к последнему шагу – прорисовка с течением времени (по таймеру).

Шаг 4

Прорисовку графика из цикла переносим в процедуру таймера, которая сама по себе является циклом. Нужно будет задать условие, по которому таймер остановится. Интервал таймера требуется поставить очень маленький, например 50 мс.



На рисунке 3 графика, после трёх последовательных нажатий мыши.

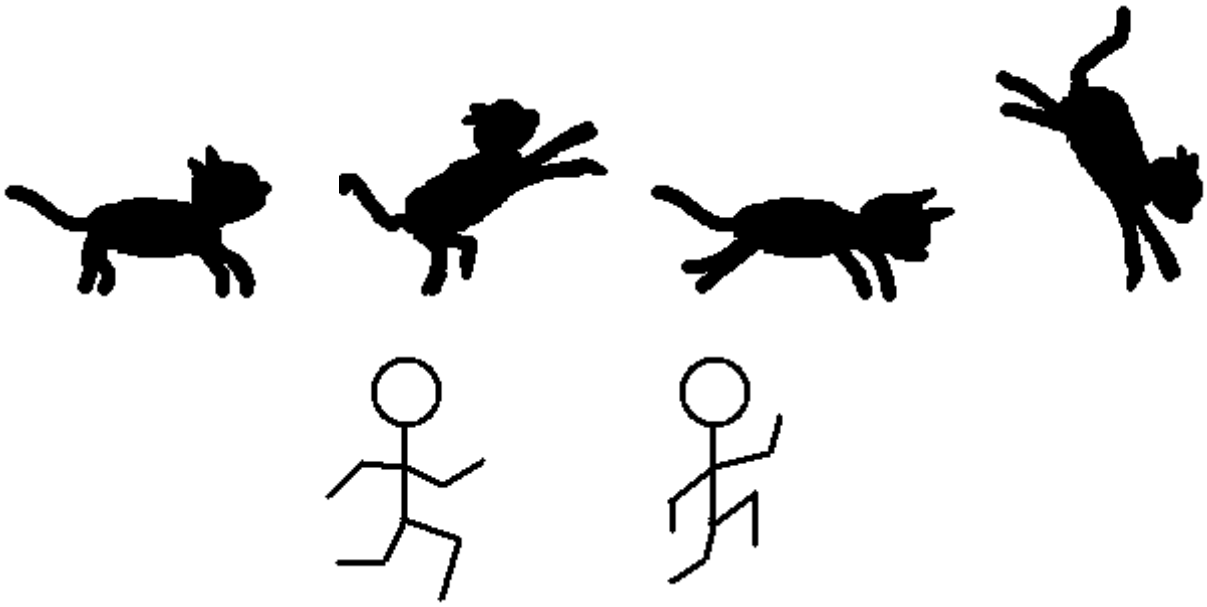
Часть 3. Простейшая анимация

Задание 1

Требуется создать простейшую анимацию, например, перемещение по форме кошки, птицы и т.п.

Для этого проекта нам понадобится объект **Image**, он находится на второй закладке (**Additional**). Свойство **Picture** этого объекта, служит для помещения в него изображения.

Итак, размещаем на форме 3 объекта **Image** (одинаковых размеров)**.** Два из них Image1 и Image2 сделать невидимыми. Рисуем в графическом редакторе 2 кадра нашей анимации. Если это кошка, то например 1 кадр – прыжок, 2 кадр – приземление. Либо, рисуем человечков. (Сохраняем с расширением ***.bmp**)



В начале в Image1 загружаем первый кадр, в Image2 – 2 кадр (свойство **Proportional** позволит автоматически подгонять размеры картинки под размеры объекта на форме). Объект Image3 с течением времени должен перемещаться по форме и в него последовательно загружаются картинки из объекта Image1 потом из Image2, затем снова из Image1 и т.д. В результате получается «мультфильм»

Загружать картинки из одного объекта в другой можно так:

Image2.Picture := Image1.Picture;

Задание 2

Сделайте такую же анимацию, но кадров должно быть больше. Для этого придется размещать на форме больше невидимых объектов **Image** для дополнительных кадров, либо поступить проще, воспользовавшись методом ***Image1.Picture.LoadFromFile(Filename: string);***