**6.** Разработайте приложение в С++ Builder в соответствии с заданием. На форме расположите компонент типа TImage для отображения анимации изображения, компонент типа TTrackBar для задания скорости движения, компонент типа TColorBox для задания цвета.

В режиме проектирования задайте минимальное значение для компонент типа TTrackBar 0, а максимальное 10. Задайте цвет по умолчанию красный, растяните компонент типа TImage почти по размеру формы.

При загрузке формы нарисуйте схематично с использованием графических примитивов на канве компонента типа TImage зонтик цветом по умолчанию(+2.5). Осуществите анимацию движения зонтика снизу вверх (+2.5). При выходе за границы он должен появляться снизу (+1.5). Предоставьте возможность задавать цвет зонтика с использованием компонента типа TColorBox (+1). Реализуйте возможность задавать скорость при помощи ползунка, чем больше значение, тем выше скорость (+1.5). Выведите на канву компонента типа TImage свою фамилию в левом верхнем углу (+0.5).

int speed;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::TrackBar1Change(TObject \*Sender)

{

// Обновляем значение скорости при изменении положения ползунка

speed = TrackBar1->Position;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

ColorBox1->Selected = clRed; // задаем начальный цвет

// Рисуем зонтик на TImage

TCanvas \*canvas = Image1->Canvas;

canvas->Brush->Color = ColorBox1->Selected;

// Нарисовать зонтик со схематическими примитивами

int centerX = Image1->Width / 2;

int centerY = Image1->Height / 2; // изменено для размещения зонтика в центре картинки

int umbrellaHeight = Image1->Height / 3;

// Рисуем основание зонта (овал)

int baseWidth = umbrellaHeight \* 2;

int baseHeight = umbrellaHeight / 2;

canvas->Pen->Color = clBlack;

canvas->Brush->Color = clBlack;

canvas->Ellipse(centerX - baseWidth / 2, centerY - baseHeight / 2, centerX + baseWidth / 2, centerY + baseHeight / 2); // изменено для размещения зонта в центре картинки

// Рисуем ручку зонта

int handleWidth = 6;

int handleHeight = Image1->Height / 3;

int handleX = centerX - handleWidth / 2;

int handleY = centerY + baseHeight / 2; // изменено для размещения зонта в центре картинки

canvas->Pen->Color = clBlack;

canvas->Brush->Color = clBlack;

canvas->RoundRect(handleX, handleY, handleX + handleWidth, handleY + handleHeight, 5, 5);

// Выводим фамилию в левом верхнем углу

canvas->Font->Color = clWhite;

canvas->TextOut(10, 10, "Зиневич");

// Запускаем анимацию

Timer1->Enabled = true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Timer1Timer(TObject \*Sender)

{

// Анимация движения зонтика

// Получаем текущие координаты зонтика

int x = Image1->Left;

int y = Image1->Top;

// Перемещаем зонт вверх на скорость speed

y -= speed;

// Если зонт вышел за верхнюю границу формы

if (y + Image1->Height < 0)

y = ClientHeight; // появляем зонт снизу

// Обновляем положение зонтика

Image1->Left = x;

Image1->Top = y;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ColorBox1Change(TObject \*Sender)

{

// Update the color of all elements

// Get the selected color from the ColorBox

TColor selectedColor = ColorBox1->Selected;

// Update the color of the umbrella base

Image1->Canvas->Brush->Color = selectedColor;

Image1->Canvas->Pen->Color = selectedColor;

int centerX = Image1->Width / 2;

int centerY = Image1->Height / 2;

int umbrellaHeight = Image1->Height / 3;

int baseWidth = umbrellaHeight \* 2;

int baseHeight = umbrellaHeight / 2;

Image1->Canvas->Ellipse(centerX - baseWidth / 2, centerY - baseHeight / 2, centerX + baseWidth / 2, centerY + baseHeight / 2);

// Update the color of the umbrella handle

int handleWidth = 6;

int handleHeight = Image1->Height / 3;

int handleX = centerX - handleWidth / 2;

int handleY = centerY + baseHeight / 2;

Image1->Canvas->Brush->Color = selectedColor;

Image1->Canvas->Pen->Color = selectedColor;

Image1->Canvas->RoundRect(handleX, handleY, handleX + handleWidth, handleY + handleHeight, 5, 5);

// Update the font color

Image1->Canvas->Font->Color = selectedColor;

// Refresh the Image to reflect the changes

Image1->Refresh();

}