9. События в программе.

В **Паскале** программа при запуске, начинает выполнять все операторы один за другим пока не закончит. А **Windows** – программы?
Представьте, что Вы запустили текстовый редактор **Word**, и он начинает выполнять все операции, которые умеет. Кошмарный ужас!
Windows - программы работают иначе. В основе программирования под **Windows** заложен принцип процедурного программирования. Программа в **Delphi** (как и в **Windows**) состоит из процедур. После того, как программа запущена, она выполняет некий код инициализации, т.е. помещается в память компьютера. И всё! Она ничего не делает, пока не наступит нужное событие. И в зависимости от того, какое событие наступило, выполняется та или иная процедура. Что-то вроде распорядка дня.

> **Лирическое отступление.**
> Развитие и поведение психически нормального человека – пример процедурного программирования.
> Нам расписывают, и мы осваиваем разные процедуры: умываться, чистить зубы, мыть посуду, прыгать через скакалку, считать вслух до ста, находить корни квадратного уравнения, пользоваться законом сохранения энергии, петь противным голосом, подмигивать левым глазом и т.д. Но ведь не делаем мы это беспрерывно! Ждём события, запускающего данную процедуру.
> На ситуации запуска процедуры, не соответствующей событию, основано большое количество анекдотов.
>
> Официант посетителю:
> – Как же Вы пойдёте? Вы же на ногах еле стоите?
> – А я на машине.

Однако продолжим.
Программа выполняется на основе сообщений о событиях, которые обрабатываются программным кодом приложения. Такой код необходимо написать для каждого события, на которое должна реагировать программа. Процедура, предназначенная для реагирования на какое-либо событие, называется **процедурой обработки события. Delphi** генерирует процедуры обработки для каждого события и дает им имена в соответствии с именами компонентов.

Если сделать двойной щелчок на кнопке, появится:

```
Procedure Tform1.Button1Click(Sender: TObject);
Begin

end;
```

Процедура расшифровывается так: на форме **Form1** существует объект (кнопка **Button1),** щелчок по которой **(Click),** запустит данную процедуру.
Чтобы этот текст появился в окне редактора кода, мы выполняли двойной щелчок на кнопке **Button1.** Это быстрый, но не лучший способ. Есть другой, универсальный способ создавать процедуры для обработки событий.

Создание процедуры обработки событий.

Наиболее часто используемые события:

Событие	**Расшифровка**
OnClick	Вызывается, когда пользователь нажимает и отпускает кнопку мыши, т.е. выполняет щелчок.
OnClose	Вызывается непосредственно перед закрытием формы.
OnCreate	Для формы - вызывается при создании формы, т.е. в самом начале выполнения программы.
OnDblClick	Вызывается, когда пользователь выполняет двойной щелчок.
On Key Press	Вызывается, когда пользователь нажимает любую клавишу с "читаемым" символом из набора ASCII. Клавиши, не имеющие символьного значения, например, Shift или F1, не вызывают это событие.
OnMouseDown	Вызывается, когда пользователь нажимает кнопку мыши.
OnMouseMove	Вызывается при перемещении пользователем указателя мыши над данным элементом управления.
OnMouseUp	Вызывается, когда пользователь отпускает нажатую кнопку мыши.
OnResize	Вызывается, как только размеры компонента изменяются.

Задание 11а.

Сделайте так, чтобы в заголовке формы появлялось название объекта, на который наведён курсор.

1. Запустите **Delphi**.
2. Добавьте на форму две кнопки**.**
3. Щелкните один раз на форме **Form1** (т.е. выделите ее).
4. В окне инспектора объектов перейдите на закладку **Events (события).**

   Напомним, что изменить это свойство формы во время выполнения программы можно так: **Form1.Caption :='наберите текст, который должен быть в заголовке';**
5. Щёлкните событие OnMouseMove.
6. Двойной щелчок на пустом месте.

   Откроется окно редактора кода, и вы видите процедуру, созданную **Delphi**:

   procedure TForm1.FormMouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

   begin

   end;

7. между **begin** и **end** напишите текст: **Form1.Caption := ' Это форма';**
8. Теперь, в том же духе, для кнопок.
9. Проверьте работоспособность проекта.
10. Сохраните его.

Совет. Сделали кусочек – проверьте, как работает.

Самостоятельная работа.

Экспериментируйте, создавая проекты с использованием событий, описанных выше.