

## ЛАБОРАТОРНАЯ РАБОТА № 1(4часа)

**Тема: «Среда программирования Visual Basic. Написание первых программ».**

**Цель работы:** Изучить интегрированную среду и основные правила разработки проекта.

### **О языке Visual Basic**

Перед начинающими программистами всегда встаёт один и тот же вопрос, а именно, какой язык программирования выбрать? Можно сказать, что лучше начинать с лёгкого и в то же время мощного языка - Visual Basic (VB). Изучив приёмы программирования на VB, вы сможете без особых усилий изучить другие языки, такие как Pascal, C++ и др.

Слово "БЕЙСИК" (BASIC) - "базовый, основной" - образовано из начальных букв английского выражения "Универсальный язык символического кодирования для начинающих". Это "для начинающих" долго вызывало пренебрежение программистов, причём подобное пренебрежение не исчезло до сих пор, несмотря на наличие профессиональных изданий VB.

Предполагается изучение наиболее общих основ программирования, характерных всем языкам и при том одинаково в каждом из них проявляющихся. Алгоритмы и принципы построения программ, которые вы изучите в этом курсе, используются в тех или иных модификациях в каждом языке программирования, как основа. Необходимо понимать и уметь использовать их, чтобы иметь возможность писать эффективные и грамотные программы.

### **Событийная модель программирования**

В "старых" версиях бейсика, таких как QBasic, использовалась плоская структура написания программы. Каждая программа начиналась и заканчивалась в определённых местах. Вся программа выполнялась последовательно, и иногда, возможно, вызывались пользовательские процедуры и функции. Т.е. если программу "запускали", то она сразу начинала выполняться с первой команды, и, дойдя до конца, завершалась.

В Visual Basic это происходит совсем иначе. В Visual Basic, как и во многих других языках, предназначенных для написания приложений под Windows, используется событийно-управляемая модель программирования. Дело в том, что ОС Windows имеет GUI (Graphical User Interface), т.е. графический интерфейс пользователя, в котором используются стандартные элементы управления, такие, как окна (они же формы), кнопки, списки, поля, для ввода текста и т.п. В любом языке высокого уровня программа строится на основе этих элементов.

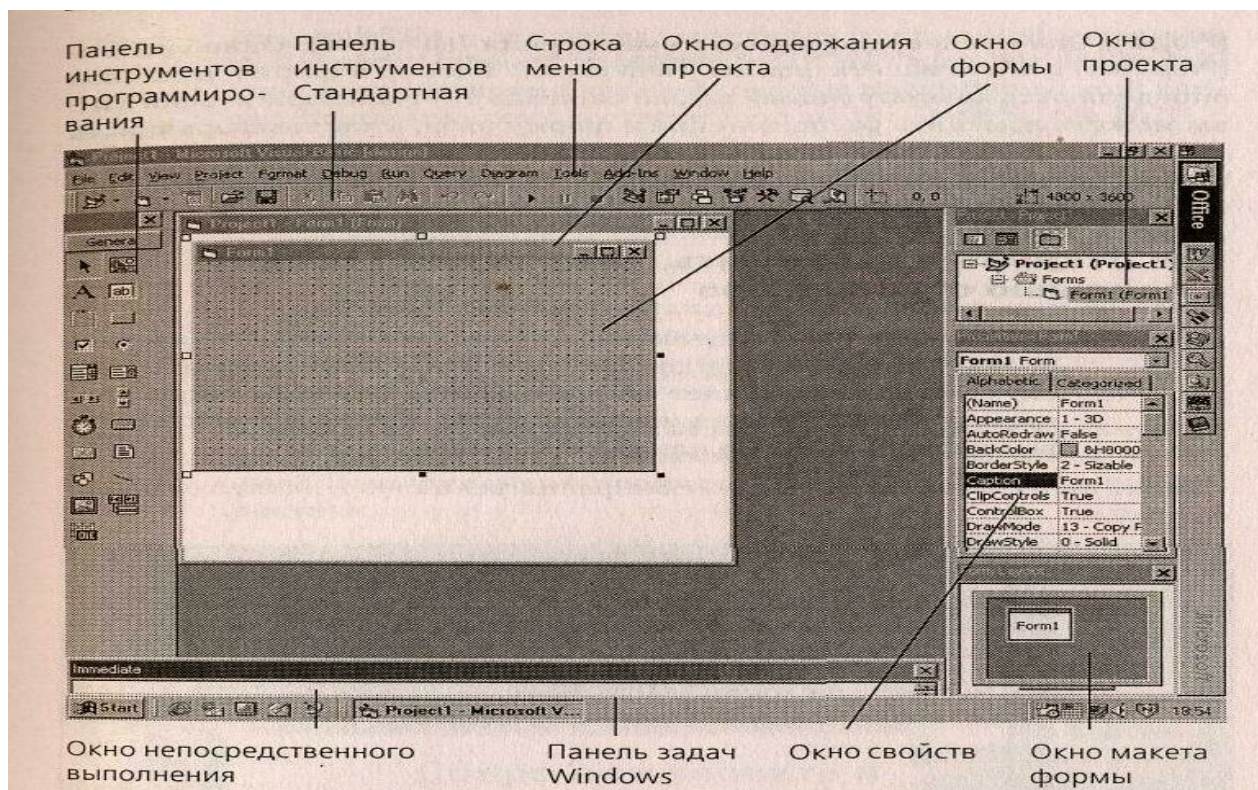
Программа, работающая под управлением операционной системы Windows, выполняется не последовательно, а в зависимости от поведения пользователя или протекающих в системе процессов. Каждое происходящее в системе действие, для программы является событием. Так, например, нажатие на кнопку мышью – это событие. Перемещение указателя мыши – тоже событие. Старт программы (загрузка формы и отрисовка ее на экране) – тоже событие. И даже «тик» секунды таймера может быть событием. Программисту необходимо понимать, что современные программы пишутся по принципу «реагирования на события». После запуска программа загружена и «ждет» от пользователя (или от системы) действий. И в дальнейшем реагирует на них. Так вы можете написать такую программу, чтобы она производила сложение при нажатии на кнопку или выводила некоторую надпись. Можно заставить программу вести себя тем или иным образом при различных событиях, происходящих в системе.

**Запуск Visual Basic**

1. В Microsoft Windows щелкните на кнопке Start (Пуск), выберите пункт Programs (Программы) и укажите на папку Microsoft Visual Basic 6.0. Появятся значки, соответствующие содержимому папки.
2. Щелкните на значке программы Visual Basic 6.0. Появится диалоговое окно нового проекта (New Project). В нем указан тип программного проекта, установленный по умолчанию.
3. Щелкните на команде Open (Открыть), чтобы выбрать проект, установленный по умолчанию –

стандартное 32-битное приложение Visual Basic. Новый проект откроется в среде программирования Visual Basic вместе с некоторыми окнами и инструментами, как показано на рис.1.1.

Инструменты, имеющиеся в среде программирования Visual Basic, помогут вам при конструировании Basic-программ. На рис. 1.1 изображена интегрированная среда разработки IDE VB. Среда называется **интегрированной**, поскольку с экрана можно вызвать любой необходимый инструмент программирования. Среду также часто называют **программой конструирования**. Для



краткости изложени я будем называть ее просто **IDE**.

Рис . 1.1 Интегрированная среда разработк и (IDE) Visual Basic

Чт обы

получить подсказку о функции конкретной кнопки, поместите на эту кнопку указатель мыши и не перемещайте его некоторое время.

Строка меню обеспечивает доступ к большинству команд, управляющих средой программирования. Меню и команды работают в соответствии со стандартными соглашениями, общими для всех приложений Windows. Запустить команду меню можно с помощью мыши или клавиатуры.

Под строкой меню расположена панель инструментов - набор кнопок, являющихся ярлыками для команд, с помощью которых осуществляется работа в среде Visual Basic.

В нижней части экрана расположена панель задач Windows. Ее можно использовать для переключения между компонентами Visual Basic или для активации других приложений Windows. Если вы работали с такими программами, как Microsoft Word или Microsoft Excel, то понятие панели задач вам уже знакомо. Для активации кнопки на панели задач щелкните на кнопке, поместив на нее указатель мыши. На панели задач может также находиться значок программы Microsoft Internet Explorer или другого Internet - браузера.

**IDE** состоит из следующих компонентов:

- главного меню;
- окна проекта (проводник проекта);
- окна макета формы;
- конструктора форм;
- панели инструментов;
- окна свойств;
- панели элементов;
- окна просмотра объектов

Главным меню называется строка текста, расположенная в верхней части окна **IDE**, которая внешне похожа на меню других приложений **Windows**. Рассмотрим основные позиции меню,

используемые для работы с **VB**.

**Файл (File)** — содержит команды открытия и сохранения проектов, создания исполняемых файлов и список последних открывавшихся проектов.

**Правка (Edit)** - стандартные команды по работе с буфером обмена: вырезание, копирование, вставка, а также множество команд форматирования и редактирования кода программы. Команда поиска может использоваться в процедуре, модуле или целом проекте.

**Вид (View)** - команды отображения или скрытия компонент и инструментов.

**Проект (Project)** - команды добавления и удаления форм, программных модулей, страниц свойств и компонентов.

**Формат (Format)** -команды выравнивания элементов управления по координатной сетке формы.

**Отладка [Debug]** - команды отладки программы приложения, т. е. можно запустить и остановить приложение, расставить точки прерывания и выбрать рассматриваемые объекты.

**Запуск (Run)** - команды запуска, прерывания и останова текущего приложения.

**Запрос (Query)** - Это меню доступно при создании баз данных

**Инструменты (Tools)** - средства создания компонент и элементов управления **Active X**, команды запуска **Menu Editor** и открытия окна **Option** для настройки рабочей среды **IDE** .

**Добавления (Add-Ins)** - настройки IDE, которые при необходимости можно добавлять или удалять. Используется при создании баз данных.

**Окна (Windows)** - стандартное меню, содержащее команды упорядочения, разделения и расположения окон на экране.

**Помощь (Help)** ~ команда вызова справочной системы.

В среде Visual Basic имеются также Окно инструментов (Toolbox), Окно содержания проекта (Project Container), Окно формы (Form), Окно проекта (Project), Окно непосредственного выполнения (Immediate), Окно свойств (Properties) и Окно макета формы (Form Layout). Размер и форма этих окон определяются конфигурацией вашей системы. В Visual Basic версий 5 и 6 вы можете изменять расположение и форму окон, а также сворачивать их, чтобы сделать доступными и видимыми на экране все необходимые элементы системы программирования. О том, как настраивать среду программирования, вы узнаете в ходе этого урока.

**Панель инструментов** Панели инструментов позволяют получить быстрый доступ к наиболее часто используемым командам меню. По умолчанию **Панель инструментов** находится под главным меню. Если панель инструментов не видна, выполните команду «**Вид**», выберите вкладку «**Панели**» и уставьте флажок  $\checkmark$  в позиции **Standard**. Кроме того, в **IDE VB** есть возможность использовать дополнительные панели инструментов, применяемые при редактировании, конструировании и отладке форм. Команда **Вид** => **Панели** => **Customize** открывает доступ к этим панелям. Установка \ снятие флажка  $\checkmark$  в соответствующей позиции делает панели доступными для работы или скрывает их.



Рис. 1.2. Панель инструментов VB

### Структура проекта VB и как сохранить проект

В Visual Basic любой проект состоит из следующих файлов:

- Файл каждой формы (расширение frm). Это обычный текстовый (ASCII) файл, в котором

- записан весь код, помещённый в форму, а также свойства всех помещённых на форму элементов управления и самой формы тоже.
- Файл проекта, содержащий информацию о проекте (расширение vbp) .
- Информация о рабочей области проекта (workspace) (расширение vbw) .
- Файл каждой формы, содержащий бинарную информацию, если она задействована в программе (расширение frx). Например, это может быть картинка PictureBox.

Это необходимый минимум. (Хотя бывают и исключения, например, когда в проекте не используются формы, тогда вместо frm файла, будет bas файл.)

После запуска нового проекта Visual Basic автоматически создает одну форму, которая в начале пуста – на ней нет кнопок и других элементов управления. Вы можете переключаться между формой и ее кодом.

Вначале работы проект сразу необходимо сохранить, чтобы в дальнейшем не отвлекаться на это. Сохранять каждый проект нужно в отдельной папке, потому что каждый проект – это несколько различных файлов и сохранив два проекта в одной папке, вы будете путаться какой файл, к какому проекту относится.

Итак, создайте отдельную папку для вашего проекта и выберите в меню «Файл» опцию «Сохранить проект как...». Проект, включает в себя: все файлы используемых вами форм, другие вспомогательные файлы, а так же файл проекта, в котором все это описано. Вам необходимо дать имена всем формам и самому проекту при сохранении. В начале Basic спрашивает, какое имя дать форме. По умолчанию форма у вас одна (если вы не создавали других) и ей можно дать любое имя, отражающее ее назначение. Это может быть, например, «frmMain» - frm потому что это форма, а Main, потому что она главная. Вы можете выбрать любое имя на английском. Сохраненная форма будет находиться в файле «frmMain.frm» в папке, которую вы указали для проекта. Теперь нужно сохранить сам проект. Дайте ему название, отражающее его суть. Например, «MyFirstProject» - если это ваш первый проект. Тогда файл проекта получит имя «MyFirstProject.vbp».

***Важно помнить, что все имена файлам нужно давать английскими буквами и без пробелов!***

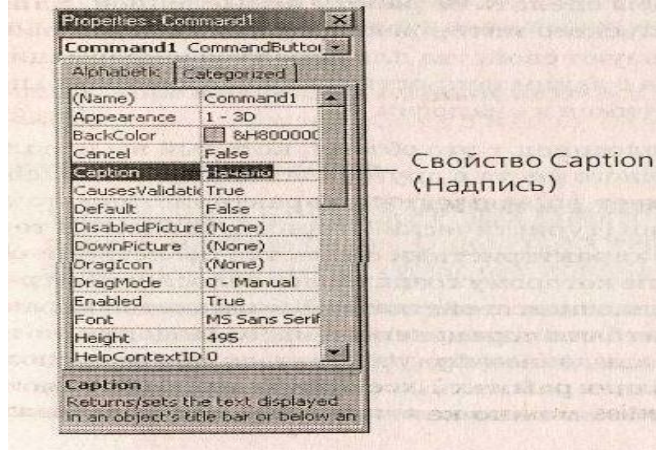
### **Изменение размеров и расположения инструментов программирования**

В среде программирования Visual Basic имеется семь инструментов. Вы можете размещать их в любом месте рабочего стола и изменять их размеры так, чтобы вам было удобно работать. Visual Basic 6 позволяет перемещать любые инструменты программирования, прикреплять их, изменять их размеры.

*В Visual Basic 6 появилась новая возможность работы с окнами ~ прикрепление (docking).*

Чтобы переместить окно или панель инструментов, поместите указатель мыши на строку заголовка, нажмите клавишу мыши и, не отпуская ее, «перетащите» объект в новое положение. Если вы поместите окно так, что его граница совпадет с границей другого окна, то оно присоединится (прикрепится) к этому окну. Прикрепленные окна удобны тем, что они всегда находятся поверх других окон и поэтому всегда видны на экране. Если вам нужно увеличить рабочее поле прикрепленного окна, перетащите одну из его границ. Если вы устали от прикреплений и хотите, чтобы эти окна перекрывались другими окнами, щелкните на команде Options (Настройка) в меню Tools (Инструменты), щелкните на вкладке Docking (Прикрепления) и удалите контрольные маркеры от тех инструментов, которые вы хотите сделать неприкрепленными. В следующих упражнениях вы будете перемещать, прикреплять и изменять размеры разных инструментов в среде программирования Visual Basic.

**Форма интерфейса пользователя**      *Форма* в Visual Basic - это окно, которое вы настраиваете для создания пользовательского интерфейса вашей программы. Форма может содержать меню, кнопки, окна списков, полосы прокрутки и другие элементы, существующие в Windows-



программах. *Форма* — это окно в интерфейсе пользователя.

При запуске среды программирования Visual Basic по умолчанию появляется форма, которая называется Form1, со стандартной сеткой (группа регулярно расположенных точек). Вы можете использовать эту сетку для создания пользовательского интерфейса вашей программы. Вы можете изменить размеры формы с помощью мыши; форма может

занимать часть экрана или весь экран. Если вы хотите добавить новые формы, щелкните на команде Add Form (Добавить форму) в меню Project (Проект). Если часть формы скрыта инструментами программирования, то вы можете либо закрыть часть инструментов, либо изменить их размеры, либо перетащить их так, чтобы они занимали меньше места. Перемещение формы по экрану не влияет на то, где появится эта форма при реальном запуске программы. Характеристики запуска управляются из окна Form Layout (Макет формы). Для установки расположения формы при запуске программы поместите уменьшенное изображение этой формы в нужное место в окне Form Layout.

### Панель инструментов программирования



- Инструменты и средства управления на панели инструментов служат для того, чтобы добавлять новые элементы пользовательского интерфейса. Чтобы открыть панель инструментов, щелкните на кнопке Панель инструментов. Панель инструментов обычно располагается вдоль левой стороны экрана. После того, как средства управления внесены в форму, они становятся объектами, или программируемыми элементами пользовательского интерфейса. Средства управления, находящиеся на панели инструментов, можно использовать для добавления в форму рисунков, этикеток, кнопок, списков, полос прокрутки, меню и геометрических фигур. После того, как пользователь запустит программу, эти элементы появятся на экране и будут работать так же, как любой объект в стандартном Windows-приложении.

Панель инструментов содержит также средства управления для создания объектов, выполняющих специальные «заэкранные» операции, такие как, управление информацией в базах данных, контроль временных интервалов и т.д. *Если вы поместите указатель мыши на тот или иной инструмент, то через некоторое время рядом с этим инструментом появится его название-подсказка.*

**Окно Properties (Свойства)** Окно Properties (Свойства) позволяет изменять характеристики (установки) элементов пользовательского интерфейса в форме. Например, вы можете изменить шрифт, кегль или выравнивание в сообщении, которое выводится вашей программой. (С помощью Visual Basic вы можете выводить текст любым шрифтом, инсталлированным в вашей системе - так же, как в программах Excel или Word.) Изменять установки свойств в окне Properties можно как при создании пользовательского интерфейса, так и тогда, когда программа уже запущена - через программный код. Окно свойств содержит список всех объектов, использующихся в данном пользовательском интерфейсе (форме). В окне свойств также могут перечисляться изменяемые установки свойств, для каждого объекта. Свойства можно просматривать в алфавитном порядке или по категориям.

В следующем упражнении вы измените свойство Caption (Надпись) у командной кнопки.

**Упражнение 1: Изменение свойств** 1. Поместите на форму командную кнопку (CommandButton). Щелкните на созданном объекте в форме. Когда объект (командная кнопка) обведен прямоугольником,

значит, он выбран. Прежде чем работать с объектом в форме Visual Basic, вы должны сначала выбрать объект.

2. Если окно свойств не было открыто, щелкните на кнопке Properties Windows (Окно свойств) на панели инструментов. Окно свойств будет подсвечено (выбрано), т.е. оно появится на экране.

3. Дважды щелкните по выделенному полю Caption (Надпись) окна Properties (Свойства). Ваш экран будет выглядеть так, как показано на рисунке 1.3. В окне свойств - перечислены все установки свойств, для командной кнопки в форме (всего для командных кнопок доступны 33 свойства). Имена свойств указаны в левой колонке окна, а текущие установки для каждого свойства - в правой. На вкладке Alphabetic свойства перечисляются в алфавитном порядке.

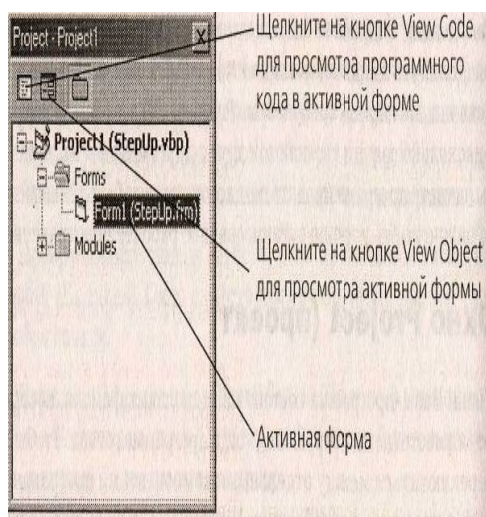
4. Дважды щелкните на свойстве Caption (Надпись) в левой колонке. Текущая надпись **Command1** в правой колонке будет выделена, и справа от нее замигает курсор.

5. Нажмите клавишу Del, наберите **Выход**, затем нажмите клавишу Enter. Обратите внимание, что надпись на форме изменилась. При следующем запуске программы на командной кнопке появится надпись **Выход**.

**Упражнение 2. Текстовое окно** Аналогично упражнению № 1, разместите на форме текстовое окно, поменяйте текст в окне, выберите другой шрифт, его размер и начертание, цвет.

### Размышления о свойствах

В Visual Basic каждый элемент пользовательского интерфейса в программе (включая его форму) имеет ряд определяемых свойств. Свойства могут устанавливаться в процессе разработки в окне Properties, кроме того, свойства могут задаваться и изменяться в программном коде после запуска программы, чтобы сделать ее работу осмысленной. Рассмотрим способы прикрепления. Самый быстрый - дважды щелкнуть на заголовке окна и подтащить окно к границе другого окна. Конечно, этот прием требует некоторого навыка, особенно когда окон много и несколько границ смыкаются друг с другом; но мы надеемся, что когда вы начнете программировать по-настоящему, то быстро поймете, насколько важно иметь хорошо организованное рабочее пространство.



**Окно Project (проект).** Visual Basic-программа состоит из нескольких файлов, которые собираются вместе (компилируются), когда программа готова. Чтобы помочь вам переключаться между отдельными компонентами программы при работе над проектом, разработчики Visual Basic включили в среду программирования окно проекта (Project). В окне Project перечисляются все файлы, используемые при программировании, и осуществляется доступ к ним при помощи двух кнопок: View Code (Просмотр кода) и View Object (Просмотр объекта). Команды в меню File (Файл) в окне Project (Проект) позволяют вам добавлять, удалять или сохранять ваши файлы. Изменения отображаются в окне Project сразу после их внесения в проект.

Рис. 1.4 Окно проекта

**Файл проекта** содержит список всех поддерживаемых файлов в программном проекте и имеет расширение .vbp (Visual Basic Project). В Visual Basic 6, в окне проекта, можно одновременно загрузить несколько файлов проектов. Для переключения между ними достаточно щелкнуть мышью на имени проекта. Под именем проекта в окне Project показаны компоненты каждого проекта и связь между ними. «Дерево» проектов похоже на структуру папок в окне Проводника в Windows. Вы можете разворачивать и сворачивать эти «ветви», щелкая на значках (+) и (-) рядом с папками.

- Просмотр окна проекта** 1. Щелкните на кнопке Project Explorer (Проводник проекта) на панели инструментов. Окно Project теперь выделено. Если это окно было закрыто, сейчас оно появилось.
2. Дважды щелкните на строке заголовка окна Project (Проект), чтобы отобразить его на экране как плавающее (не прикрепленное). Ваш экран будет выглядеть так, как показано на рис.1.4.
3. Щелкните на значках (+) около папок Forms (Формы) и Modules (Модули), если эти папки еще не развернуты. Файл проекта в этом программном проекте называется StepUp. В проекте StepUp перечислены файлы StepUp.frm и StepUp.bas. StepUp.bas содержит код, разделяемый всеми частями программы. StepUp.frm содержит форму пользовательского интерфейса и программный код, связанный с объектами в форме. Когда программа компилируется в выполняемый файл, эти файлы будут скомбинированы в единственный .exe-файл.
4. Дважды щелкните на заголовке окна Project, чтобы прикрепить его.


#### Панель элементов управления.




Панель элементов управления — основной рабочий инструмент при визуальной разработке форм приложения. Панель элементов управления вызывается из меню **View** (Вид) командой **Toolbox** (Панель элементов управления). Для вызова этой панели можно воспользоваться также кнопкой **Toolbox** на стандартной панели инструментов. В составе панели элементов управления содержатся основные элементы управления форм: метки, текстовые поля, кнопки, списки и другие элементы для быстрого визуального проектирования макета формы. На панели представлены кнопки, назначение которых описано в табл.1.1.

Таблица 1.1.

*Кнопки панели элементов управления*

Кнопка	Название	Назначение
	Pointer (Указатель)	Используется для позиционирования маркера (указателя) мыши
	Timer (Таймер)	Размещает в форме таймер
	PictureBox (Графическое окно)	Размещает в форме графическое окно, предназначенное для объединения элементов в группы, для вывода в него графических изображений, а также текста, графических элементов и анимации
	Label (Метка)	Размещает в форме объекты, предназначенные для создания текстовой информации, надписей и примечаний
	TextBox (Текстовое поле)	Размещает в форме текстовое поле, предназначенное для ввода текстовой информации, чисел и дат
	Frame (Рамка)	Создает в форме рамку с заголовком для группировки объектов в логическую группу
	CommandButton (Кнопка управления)	Размещает в форме кнопки управления для инициации действий, выполнения команд, запуска программ
	CheckBox (Флажок)	Размещает в форме флажок, предназначенный для формирования условий выполнения программ или каких-либо настроек, работающий по принципу "да — нет"

	OptionButton (Переключатель)	Создает в форме переключатели для выбора режима работы или настроек выполнения программы
	ComboBox (Поле со списком)	Создает в форме объект, содержащий одновременно поле ввода и раскрывающийся список
	ListBox (Список)	Создает в форме список для выбора одного или нескольких значений из предлагаемого списка значений
	HScrollBar (Горизонтальная полоса прокрутки)	Размещает в форме горизонтальную полосу прокрутки, используемую в качестве ползунка для выбора значения из заданного диапазона
	VScrollBar (Вертикальная полоса прокрутки)	Размещает в форме вертикальную полосу прокрутки, используемую в качестве ползунка для выбора значения из заданного диапазона
	DriveListBox (Список устройств)	Создает в форме список устройств
	DirListBox (Список папок)	Создает в форме древовидный список папок
	FileListBox (Список файлов)	Создает в форме список файлов
	Shape (Очертание)	Создает в форме геометрические фигуры, такие как прямоугольник, квадрат, круг, эллипс, прямоугольник и квадрат со скругленными углами
	Line (Линия)	Создает линии
	Image (Изображение)	Создает в форме поля, предназначенные для отображения графических изображений
	Data (Данные)	Создает элемент управления данными в базе данных для перемещения по записям и отображения результата навигации

Для размещения элементов управления в форме с помощью панели элементов выполните следующие действия:

1. Выделите требуемый элемент управления с помощью мыши. 2. Перейдите в окно конструктора форм. Указатель мыши при этом превратится в крестик, при помощи которого можно установить местоположение размещаемого объекта.левой кнопкой мыши зафиксируйте позицию нового объекта и, удерживая кнопку, задайте размеры объекта.

**Упражнение 2:** Вывести на форму текущую дату и время.

На форме создать следующие объекты: - командную кнопку, на которой написано «Показать»; - командную кнопку, на которой написано «Выход»; - два текстовых окна.

Набрать следующие коды:

```
Private Sub Command1_Click()
Text1.Text = Date:   Text2.Text = Time
End Sub
Private Sub Command2_Click()
end
End Sub
```

- Запустите программу, установите свойства текстовых меток, сохраните ее.

**Выход из Visual Basic**



Если вы решили закончить работу в Visual Basic сегодня, то сохраните все открытые проекты и закройте программную среду. В меню File (Файл) щелкните на Exit (Выход). Если вы увидите диалоговое окно Save (Сохранить), укажите Yes (Да).

### Контрольные вопросы и задания:

- Как вы понимаете термин «событийная модель программирования»?
- Как запустить **Visual Basic**?
- Что означает термин IDE?
- Каковы основные компоненты интегральной среды IDE?
- Каковы возможности главного меню окна IDE?
- Каковы назначение и состав панели инструментов?
- Состав и назначение элементов управления?
- Где будут размещаться кнопки, текстовые окна, заголовки и все другие объекты управления создаваемого вами в **Visual Basic** приложения?
- Каково назначение, состав и возможности окна свойств?
- Каково назначение окна проекта?
- Как перейти в окно объектного кода?
- Как сохранять ваш проект и изменения в нем?



### Составление программ на Visual Basic

Теперь, когда у вас есть некоторый опыт работы в среде программирования Visual Basic, настало время создать вашу первую Visual Basic-программу. Вы сконструируете приложение под названием **Счастливая семерка** - программу, имитирующую игровой автомат «однорукий бандит». Программа **Счастливая семерка** обладает простым пользовательским интерфейсом и может быть создана и скомпилирована за несколько минут. После того как программа закончит работу, ваш экран будет выглядеть так, как показано на рисунке

### Последовательность программирования

Пользовательский интерфейс **Счастливая семерка** содержит две командных кнопки, три окна, в которых выводятся числа, графическую картинку со столбиками монет и метку **Счастливая семерка**. Эти элементы были созданы как семь объектов в форме **Счастливая семерка** с последующим изменением свойств каждого объекта. После разработки интерфейса был добавлен программный код для командных кнопок **Вращать** и **Конец**, который обрабатывал нажатия на кнопки и генерировал случайные числа. При создании **Счастливой семерки** вы используете определенную последовательность действий, состоящую из трех этапов. Итак, *первый этап* - создание пользовательского интерфейса, *второй этап* - установка свойств, *третий* - составление программного кода.

Процесс программирования игры «**Счастливая семерка**» представлен в следующей таблице.

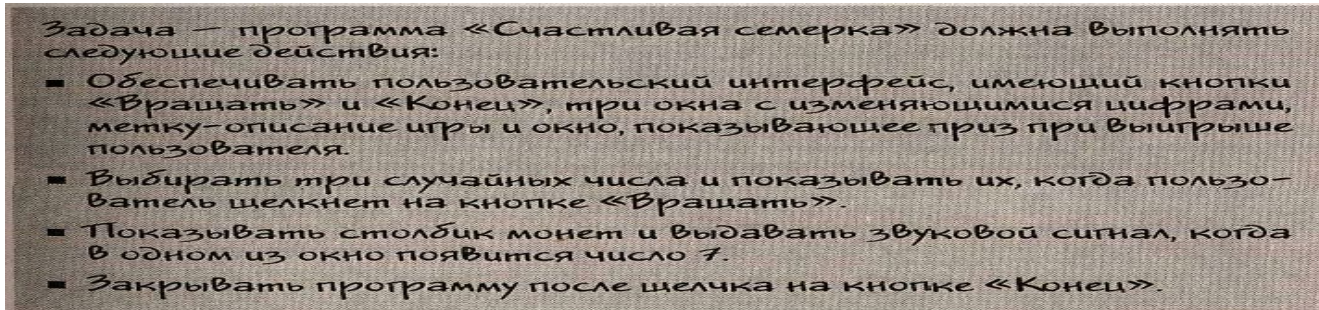
Этап программирования	Число пунктов
1. Создание пользовательского интерфейса	7 объектов.
2. Установка свойств	10 свойств.
3. Составление программного кода	2 объекта.

**Разработка пользовательского интерфейса.** Интерфейс — это внешняя оболочка приложения вместе с программами управления доступом и другими скрытыми от пользователя механизмами

управления, дающая возможность работать с документами, данными и другой информацией, хранящейся в компьютере или за его пределами. Главная цель любого приложения — обеспечить максимальное удобство и эффективность работы с информацией: документами, базами данных, графикой или изображениями. Поэтому интерфейс является, пожалуй, самой важной частью любого приложения.

Работу над программой **Счастливая семерка** начнем с создания нового проекта. Затем мы создадим пользовательский интерфейс с помощью средств управления из панели инструментов.

Можно действовать и по-другому: сначала составить алгоритм, или перечень шагов будущей программы.

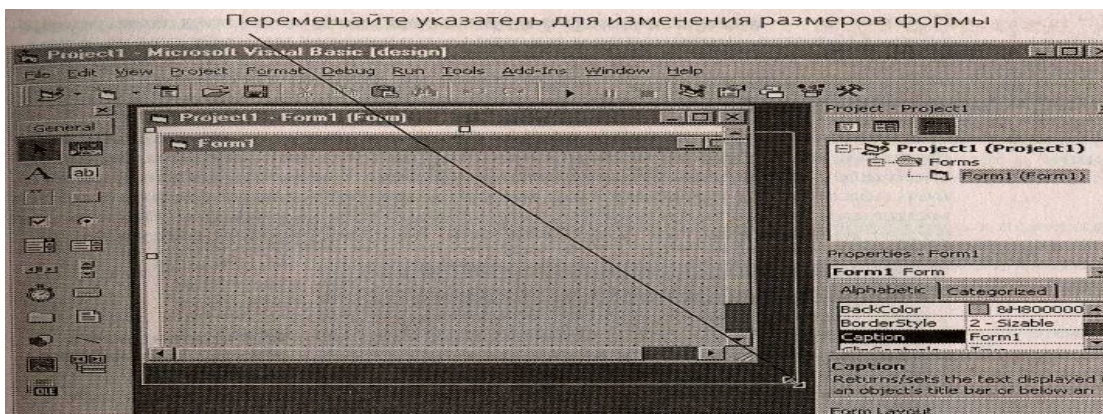


### Создайте пользовательский интерфейс

1. В меню File (Файл) щелкните на команде New Project (Новый проект).

2. Щелкните на кнопке ОК, чтобы перейти к созданию нового 32-битного приложения Visual Basic.

Visual Basic выделяет место для создания нового программного проекта и показывает чистую форму в центре экрана, которая будет использоваться для создания пользовательского интерфейса вашей программы. Далее вы увеличите размеры формы и создадите две кнопки.



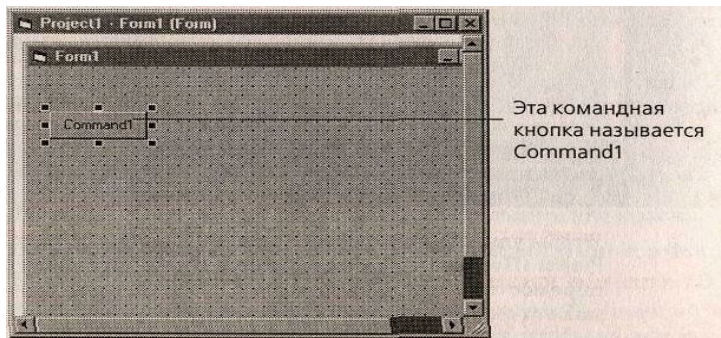
3. Поместите указатель мыши в правый нижний угол окна Формы (Form) (но не окна Project (Проект)). Указатель примет вид двунаправленной

стрелки. Перемещайте указатель, увеличивая размеры формы так, чтобы на нем могли разместиться все объекты вашей программы. После увеличения размеров формы в окне проекта появляются полосы прокрутки, как показано на следующем рисунке. Чтобы увидеть всю форму целиком, увеличивайте размер окна Project (Проект), пока полосы прокрутки не исчезнут, а также переместите или закройте окно свойств, окно проекта и окно макета формы.

4. Чтобы создать на форме объект «Командная кнопка», щелкните на кнопке Command Button (Командная кнопка) на панели инструментов. Переместите указатель мыши на форму. Теперь, когда выбрана опция управления командной кнопкой, указатель мыши принимает крестообразную форму. Такая форма облегчает работу по вычерчиванию прямоугольных объектов. Удерживая левую кнопку мыши нажатой, перетащите указатель. Объект «командная кнопка» при этом приобретает определенные очертания и привязывается к сетке, сформированной равномерно расположенными на форме точками.

5. Создадим первую командную кнопку. Переместите указатель мыши ближе к левому верхнему углу формы, затем, удерживая нажатой левую кнопку мыши, переместите указатель вниз и вправо. Когда командная кнопка станет

похожа на показанную на рисунке, остановите перемещение указателя и отпустите левую



кнопку мыши. Ваш экран будет выглядеть так, как показано на рисунке.

На форме появилась командная кнопка с набором команд управления. По умолчанию первая командная кнопка в программе называется Command1. К этому названию вы можете добавить комментарий - вы увидите его снова позднее, когда будете писать программный код.

Вы можете перемещать командные кнопки по форме с помощью мыши и изменять их размеры с помощью элементов управления всякий раз, когда Visual Basic находится в режиме разработки (когда среда программирования Visual Basic активна). Однако после запуска программы на выполнение пользователь сможет изменить положение элементов интерфейса, только изменив специальные свойства (если программа это позволяет).

### Перемещение и изменение размеров командной кнопки

1. С помощью мыши перетащите командную кнопку вправо. Командная кнопка привяжется к сетке, когда вы отпустите кнопку мыши. Сетка формы призвана помочь вам редактировать и выравнивать различные элементы пользовательского интерфейса. Вы можете изменить шаг сетки, щелкнув в меню Tools (Сервис) на команде Options (Настройка), а затем — на вкладке General (Общие). 2. Поместите указатель мыши в правый нижний угол командной кнопки. Если указатель находится на углу или стороне выбранного объекта и некоторое время неподвижен, то он принимает вид указателя размеров. Теперь, перетащив край объекта, вы измените его размеры. 3. Увеличьте размеры объекта, перетаскивая указатель вниз и вправо и удерживая нажатой левую кнопку мыши. Когда вы отпустите кнопку мыши, командная кнопка с новыми размерами привяжется к сетке. 4. С помощью указателя размеров верните кнопку к первоначальному виду и переместите ее в прежнее положение на форме.

**Создание второй командной кнопки** 1. Щелкните на кнопке Command Button (Командная кнопка) на панели инструментов. 2. Вычертите новую командную кнопку. Расположите ее под первой кнопкой и сделайте ее такого же размера. 3. При необходимости измените размеры или расположение кнопки. Если вы ошиблись, вы можете удалить кнопку и создать ее заново.

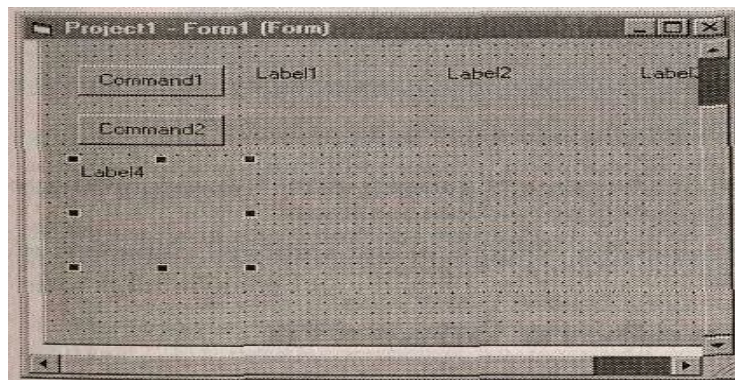
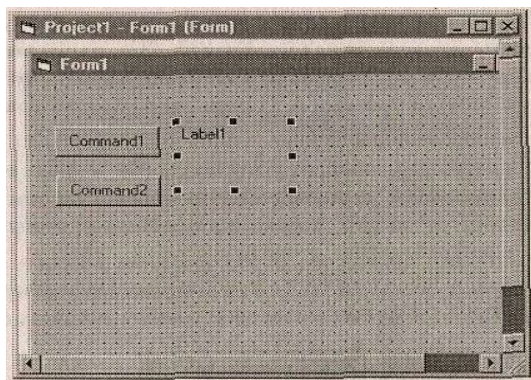
*Вы можете удалить объект, выделив его на форме и нажав клавишу Del*

### Как добавить числовые метки

Добавим метки для показа чисел. *Метка* — это специальный элемент пользовательского интерфейса, предназначенный для показа текста, чисел и символов при работе программы. Когда пользователь щелкает на кнопке **Вращать**, в окнах меток появляются три случайных числа. Если одно из чисел — семерка, пользователь выигрывает.

1. Щелкните на кнопке Label (Метка) на панели инструментов, затем поместите указатель мыши на форму. Теперь, когда выбрана опция управления метками, указатель мыши принимает крестообразную форму. 2. Создайте маленькое прямоугольное окно, как показано на следующем рисунке. Объект «метка», который вы создали, называется Label1 - это первая метка в программе. Создайте на форме еще две метки с названиями Label2 и Label3. 3. Щелкните на кнопке управления метками и вычертите окно второй метки справа от первой. Создайте метку такого же размера, как и первая. На метке появится надпись Label2, затем вычертите окно третьей метки справа от второй. Создайте метку такого же размера, как и предыдущие. На метке появится надпись Label3. 4. Добавим метку описания игры в вашу форму. Это будет четвертая и последняя метка в программе. Щелкните на кнопке

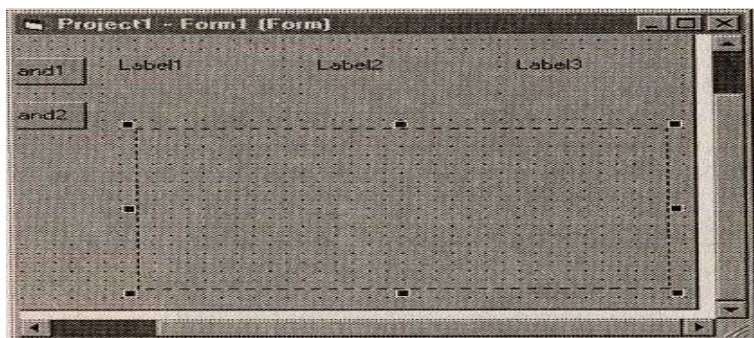
Label (Метка) на панели инструментов. 5. Создайте большой прямоугольник под двумя командными кнопками. Ваш экран будет выглядеть примерно так, как показано на следующем рисунке. Если ваши метки видны на экране не полностью, вы можете изменить их размеры.



### Как создать окно рисунка

Теперь создадим окно рисунка, в котором при выигрыше пользователя будет появляться столбик монет. Окно рисунка предназначено для вывода на экран иллюстраций, значков и других изображений. В Visual Basic имеется собственная галерея картинок, которые тоже могут быть показаны в этом окне.

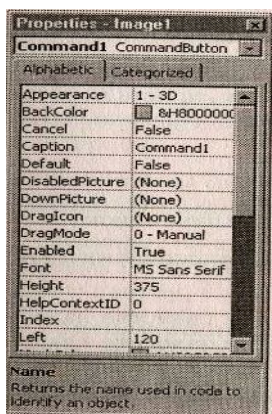
1. Щелкните на кнопке Image (Рисунок) на панели инструментов. 2. Создайте прямоугольное окно под тремя числовыми метками. Ваш экран будет выглядеть так, как показано на следующем рисунке.



Созданный объект будет назван Image1. Вы используете это имя при составлении программного кода. Теперь интерфейс создан, и вы можете перейти к установке его свойств.

### Установка свойств объекта

Вы узнали, что свойства объекта можно изменять. Для этого нужно выбрать объект на форме и изменить установки в окне свойств. В программе **Счастливая семерка** мы начнем установку свойств с изменения названий командных кнопок.



**Установка свойств командных кнопок** 1. Щелкните на первой командной кнопке (Command1) на форме. Вокруг командной кнопки появится рамка, показывающая, что кнопка выделена. 2. Дважды щелкните на заголовке окна Properties (Свойства). Появится окно свойств. Ваш экран будет выглядеть так, как показано на следующем рисунке. В окне свойств перечислены установки для первой командной кнопки. Сюда включены установки цвета фона, названия, размера шрифта (кегля) и ширины командной кнопки. 3. Дважды щелкните на свойстве Caption (Название) в левой колонке окна свойств. Будет выделена текущая установка названия (Command1). 4. Наберите новое название **Вращать** и нажмите клавишу Enter. Название кнопки изменится на **Вращать** в окне свойств

и на форме. Теперь изменим название второй кнопки на **Конец**. Сделать это можно и другим способом. 5. Откройте ниспадающее окно Object List (Список объектов) в верхней части окна свойств. В списке перечислены объекты вашей программы. 6. Щелкните на кнопке Command2 в окне списка - Установки свойств, для второй командной кнопки появятся окно свойств. 7. Дважды щелкните на текущем свойстве названия («Command2»), наберите **Конец** и затем нажмите клавишу Enter\_Название второй командной кнопки изменится на **Конец**.

**Установка свойств числовых меток.** Теперь установим в программе свойства для меток.

Первые три метки будут выводить на экран случайные числа, и свойства этих меток будут одинаковы (мы установим их как группу). Затем мы перейдем к установке свойств метки-описания.

1. Щелкните на первой числовой метке, затем, удерживая нажатой клавишу Shift, щелкните на второй и третьей числовых метках. (Если метки закрыты окном свойств, передвиньте его в другое место.)

Для выделения на форме нескольких объектов удерживайте нажатой клавишу Shift и поочередно щелкайте на выделяемых объектах. Вокруг каждой метки, по которой вы щелкнули, появятся прямоугольники выделения. Когда все три метки будут выделены, отпустите клавишу Shift.

**Примечание.** При выделении нескольких объектов в окне свойств отображаются только те свойства объектов, которые могут быть изменены как групповые. Сейчас вы измените такие свойства, как выравнивание, стиль рамки и шрифт, так чтобы числа, которые появляются в метках, были выровнены по центру, обрамлены и имели одинаковый шрифт.

2. Щелкните на свойстве Alignment (Выравнивание), а затем - на стрелке окна ниспадающего списка, который появится справа. 3. Щелкните на цифре 2 - это опция Center (Выравнивание по центру). Теперь во всех трех кнопках установлено выравнивание по центру. 4. Изменим стиль рамки. Щелкните на свойстве Border Style (Стиль рамки), а затем - на стрелке окна ниспадающего списка, который появится справа. В окне списка появится перечень доступных установок свойств (0 - None, I - Fixed Single).

5. Щелкните на свойстве I - Fixed Single в окне списка, чтобы добавить тонкую рамку вокруг каждой метки.

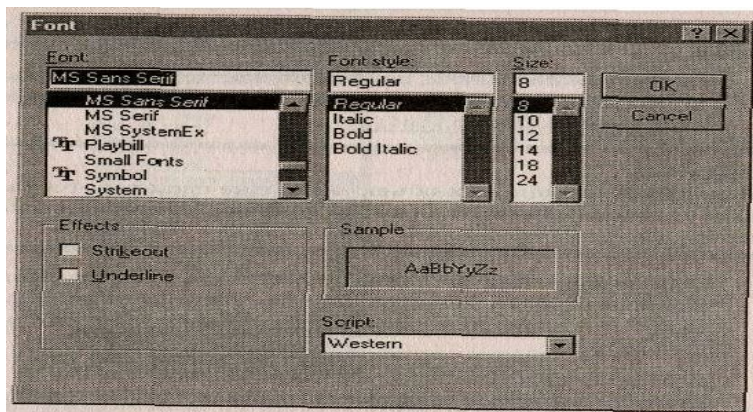
6. Изменим шрифт, которым будут выведены цифры в окнах меток. Дважды щелкните на свойстве Font (Шрифт) в окне свойств. Появится диалоговое окно Font (Шрифт). Ваш экран будет выглядеть так, как показано на следующем рисунке. 7. Измените шрифт (Font) на TimesNewRoman, начертание - на полужирное (Bold), кегль (Size) - на 24, и затем щелкните на кнопке ОК. Теперь названия меток

отображаются выбранным вами шрифтом, кеглем и начертанием. Удалим названия меток, чтобы при запуске программы окна были пустыми. Установки шрифта при этом сохранятся, так как они хранятся как отдельные свойства. Для выполнения этой операции вам придется выбирать каждую кнопку отдельно. 8. Щелкните на форме, чтобы снять выделение с трех меток, затем щелкните на правой метке. 9. Дважды щелкните на свойстве Caption, затем нажмите клавишу | Del [. Название объекта Label1 исчезнет. Позже мы расскажем, как с помощью программного кода поместить в эту метку случайное число (игровой автомат). 10. Сотрите названия второй и третьей меток на форме.

Вы подготовили три первых метки. Теперь изменим название, шрифт и цвет последней метки.

### Установка свойств метки-описания

1. Щелкните на четвертом объекте «метка» на форме. 2. Измените свойство Caption (Надпись) на **Счастливая семерка**. 3. Дважды щелкните на



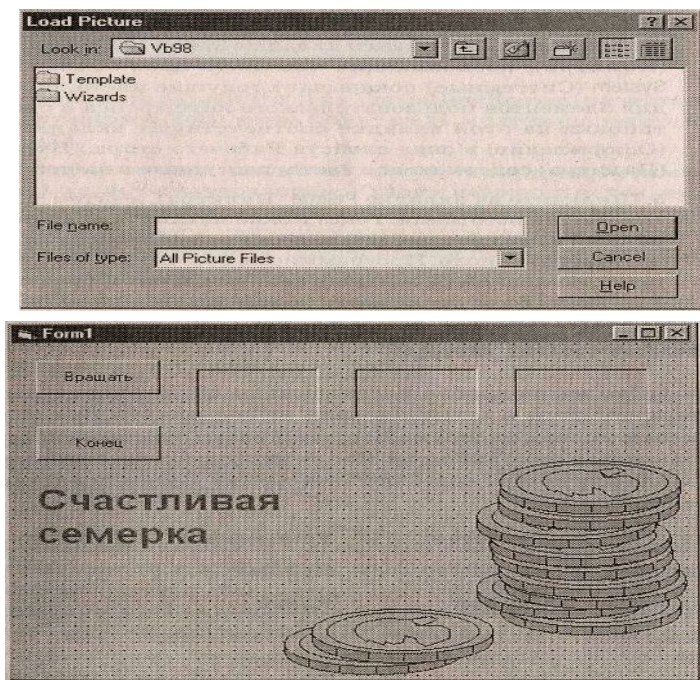
свойстве Font (Шрифт), в диалоговом окне, измените шрифт на Arial, начертание - на полужирное, кегль - на 20 и щелкните на ОК. Шрифт в окне метки обновился. Обратите внимание, что теперь название метки располагается на двух строках (длина кнопки недостаточна для того, чтобы название поместилось на строке целиком). Это очень важно: содержимое объекта должно всегда уместиться внутри объекта.

Содержимое может либо переноситься на следующую строку, либо сжиматься. 4. Изменим цвет текста. Для этого дважды щелкните на свойстве Fore Color в окне свойств. В поле списка появились вкладки System (Системные) и Palette (Палитра), позволяющие изменить цвет объекта. Вкладка System

(Системные) показывает текущие цвета, применяемые для элементов пользовательского интерфейса в системе. (Установки на этой вкладке соответствуют вкладке Appearance (Оформление) в окне свойств Рабочего стола.) Вкладка Palette (Палитра) содержит все цвета, доступные в вашей системе. 5. Щелкните на вкладке Palette (Палитра), а затем - на квадратике вишневого цвета. Текст в окне метки стал вишневым. Выбранный вами цвет представляется в окне свойств как шестнадцатеричное число. Программисты обычно имеют дело с таким представлением различных величин, и интересно посмотреть, как Visual Basic записывает подобную информацию в программе.

Теперь вы готовы изменить свойства последнего объекта.

**Свойства окна рисунка.** Объект «окно рисунка» будет содержать изображения столбиков монет. Оно появляется, когда пользователь выигрывает (т.е. в числовых метках появилась хотя бы одна семерка). Вам понадобятся три свойства: Stretch (Вытягивание) для установления точного размера картинке; Picture (Рисунок) для указания имени графического файла, который загружается в метку, и свойство Visible (Видимость), которое определяет состояние рисунка в начале работы программы.



### Установка свойств окна рисунка

1. Щелкните на объекте «окно рисунка» на форме.
2. Щелкните на свойстве Stretch (Вытягивание) в окне свойств, затем щелкните на стрелке, чтобы вызвать список, и в списке щелкните на опции True (Верно). При установке опции True (Верно) Visual Basic точно «впишет» картинку в окно рисунка. (Обычно это свойство устанавливается перед установкой свойства Picture (Рисунок).
3. Дважды щелкните на свойстве Picture (Рисунок) в окне свойств. Появится диалоговое окно Load Picture (Загрузить рисунок). Ваш экран будет выглядеть так, как показано на следующем рисунке.
4. В диалоговом окне Load Picture (Загрузить рисунок) найдите папку с рисунками. Вы увидите содержимое папки.
5. Дважды щелкните на значке папки . В диалоговом окне Load Picture (Загрузить рисунок) появится рисунок Coins.wmf (.wmf - это формат, называемый Windows-метафайл). Windows-метафайлы содержат графические объекты, которые могут масштабироваться без искажений и хорошо выглядят при любом увеличении или уменьшении.
6. Выделите файл Coins.wmf и щелкните на кнопке Open. Windows-метафайл Coins (Монеты) будет загружен в окно рисунка на форме.
7. Изменим свойство Visible (Видимость) на False (Ложно). Теперь при запуске программы монеты будут невидимы. (Позднее мы составим программный код, который будет делать их видимыми при определенных условиях.) Щелкните на свойстве Visible (Видимость) в окне свойств, затем щелкните на стрелке, чтобы вызвать список.
8. В списке щелкните на опции False (Ложно). Свойство Видимость установлено в положение «Ложно». Это свойство будет применено к программе после ее запуска, но не в процессе разработки. Законченная форма будет выглядеть так, как показано на рисунке.
9. Дважды щелкните на заголовке окна свойств, чтобы вернуть его в прикрепленное положение.

**Представление свойств в табличном виде** Свойства, которые мы уже установили для программы Счастливая семерка, можно представить следующей таблицей.

Объект	Свойство	Установка
Command1	Caption	Вращать

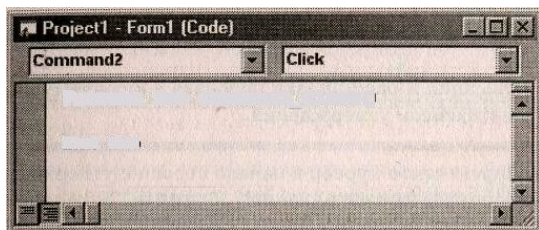
Command2	Caption	Конец
Label1, Label2, Label3	Border Style	1 - Fixed Single
	Alignment	2 - Center
	Font	TimesNewRoman, Bold, 24 (Empty)
Label4	Caption	Счастливая семерка
	Font	Arial, Bold, 20
Image1	ForeColor	Dark Purple (&H008000808)
	Picture	\coins.wmf
	Stretch	True
	Visible	False

### Составление программного кода

Теперь вы можете перейти к составлению кода для программы Счастливая семерка. Большинство объектов, которые вы создали, уже «знают», как работать после запуска программы, и готовы к приему данных от пользователя и их обработке. Внутренняя функциональность создаваемых объектов - одна из мощнейших особенностей Visual Basic! Если объекты однажды созданы и помещены на форму, то они готовы к работе без всякого дополнительного программирования. В нашей программе недостает только «начинки» — кода, который будет вычислять случайные числа, показывать их в окнах и сообщать о выигрыше. Вычислительная логика может быть встроена в приложение только с помощью программного кода, который четко расписывает, что именно программа должна делать на каждом этапе работы. Программа управляется кнопками **Вращать** и **Конец**, поэтому и код будет связан с информацией, поступающей от этих кнопок.

**Окно Code (Код)** - это специальное окно в среде программирования, которое используется для ввода и редактирования программных утверждений Visual Basic. **Работа в окне кода.**

1. Дважды щелкните на командной кнопке Конец на форме. Появится окно Code (Код), как показано на следующем рисунке.



на следующем рисунке.

Если окно меньше, чем показано выше, измените его размеры с помощью мыши. Точные размеры не очень важны, поскольку для просмотра данных вы можете пользоваться полосами прокрутки. В окне должна быть надпись

**Private Sub Command2\_Click0**

### End Sub

Блок кода, связанный с частным объектом интерфейса, мы будем называть *процедурой события* Visual Basic. Тело процедуры заключено между операторами, указывающими на начало и конец подпрограммы. Тело процедуры всегда располагается между этими строками и выполняется всякий раз, когда пользователь активизирует элемент интерфейса, ассоциированный с процедурой. В данном случае событием является щелчок мышью, но оно может быть событием и другого типа.

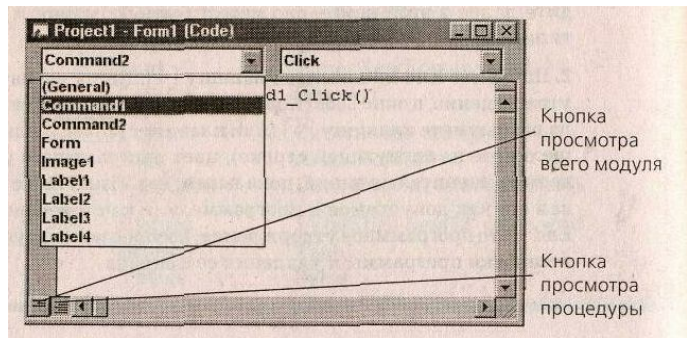
2. Наберите *End* и нажмите клавишу (стрелка вниз). Когда вы набираете утверждение, в окне Code (Код) появляются черные буквы. Когда вы нажмете клавишу [стрелка вниз] (или клавишу Enter, или просто щелкнете на следующей строке), цвет программного утверждения изменится на синий, показывая, что Visual Basic распознал его как допустимое в программе, или как ключевое слово End - это программное утверждение, которое используется для остановки программы и удаления ее с экрана.

Система программирования Visual Basic содержит несколько сотен уникальных ключевых слов, дополненных ключевыми операторами и символами. Правила их написания и расстановки пробелов должны строго соблюдаться, иначе Visual Basic не распознает их.

3. Переместите курсор в начало строки с утверждением End и четыре раза нажмите клавишу Spacebar. Утверждение End переместится на четыре пробела вправо. Так в Visual Basic принято отмечать ключевые слова начала и конца подпрограммы, чтобы сделать тексты программ более понятными и легко читаемыми. Группа соглашений относительно того, как организован программный код, обычно называется стилем программирования.

### Программный код для кнопки **Вращать**

1. Откройте список объектов в окне Code (Код). Объекты интерфейса **Счастливая семерка** появятся в списке, как показано на следующем рисунке.



2. Щелкните на строке Command1 в списке.

Появится процедура, связанная с кнопкой Command1. По умолчанию Visual Basic показывает все процедуры событий для формы в одном окне, так что вы можете легко переключаться между ними. Если вы щелкнете на кнопке просмотра процедуры в левом нижнем углу окна кода, то будете просматривать в одном окне только одну

процедуру. Чтобы снова увидеть все процедуры в одном окне, щелкните на кнопке просмотра всего модуля.

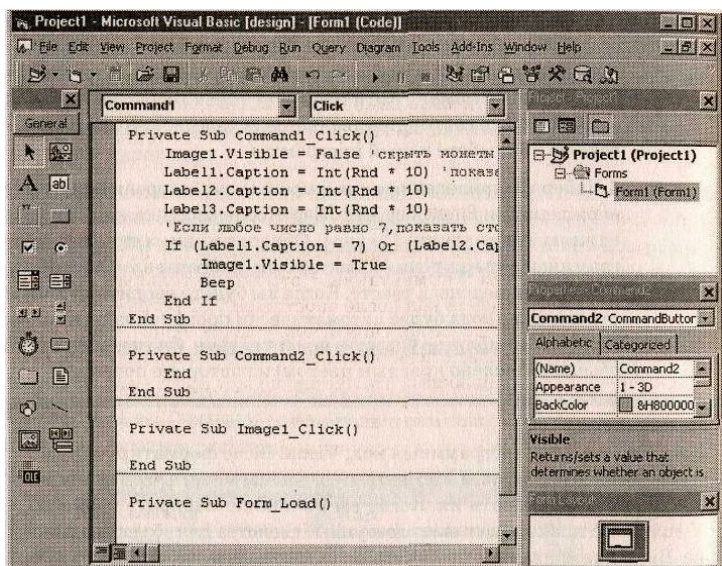
Несмотря на то, что вы поменяли название этой кнопки на **Вращать**, в программе ее имя осталось прежним (Command1). Имя программы может не совпадать с ее названием. У каждого объекта может быть несколько ассоциированных с ним процедур, по одной для каждого связанного с этим объектом события. В данный момент нас интересует только событие «щелчок мышью», так как это единственное действие, доступное пользователю при работе нашей программы.

Наберите приведенные ниже программные строки между утверждениями **Private Sub** и **End Sub**. После ввода каждой строки нажимайте клавишу [Enter]. Обратите внимание, что программные утверждения должны быть набраны в точности так, как они приведены в тексте. Когда вы будете вводить длинные строки, окно кода будет автоматически прокручиваться влево. Если вы ошиблись, удалите некорректное утверждение (оно обычно выделено красным цветом) и повторите попытку.

```
Private Sub Command1_Click()
    Image1.Visible = False
    Label1.Caption = Int(Rnd * 10)
    Label2.Caption = Int(Rnd * 10)
    Label3.Caption = Int(Rnd * 10)
    If (Label1.Caption = 5) Or (Label2.Caption = 5) Or (Label3.Caption = 5) Then
        Image1.Visible = True
        Beep
    End If
End Sub
Private Sub Command2_Click()
    End
End Sub
```

Когда вы закончите ввод, ваш экран будет выглядеть так, как показано на следующей иллюстрации.





Процедура Command1\_Click выполняется после щелчка пользователя на кнопке **Вращать** на форме. Процедура использует несколько достаточно сложных утверждений, которые формально еще не были введены. Однако, присмотревшись внимательнее, вы, возможно, увидите кое-что знакомое. Найдите ошибки. Переходите к разделу «Сохранение программы».

#### *Анализ программных утверждений.*

Процедура Command1\_Click выполняет три задачи: скрывает столбик монет, создает три случайных числа для вывода в окнах меток, показывает столбик монет при появлении числа.

Рассмотрим каждый из этих шагов. *Первая задача в процедуре выполняется строкой:*

```
Image1.Visible = False
```

Эта строка состоит из двух частей: программного утверждения и комментария. Программное утверждение (Image .Visible = False) устанавливает свойство Visible первого объекта «окно образа» (Image) ложным (False). Вы можете вспомнить, что уже устанавливали это свойство с помощью окна свойств. Сейчас вам придется снова делать это в программном коде, так как первым выполняется вращение, и вы должны удалить рисунок, который мог остаться от предыдущей игры. Так как свойство будет меняться во время работы программы, а не во время ее разработки, оно должно устанавливаться с помощью программного кода. Это удобная особенность Visual Basic, мы ещё вернемся к ней.

*Вторая часть первой строки* (показанная на экране зеленым цветом) называется комментарием. *Комментарии* — это пояснительные заметки, включенные в программный код после апострофа ('). Программисты используют комментарий для сообщения о роли утверждения в программе. Эти примечания Visual Basic не обрабатывает; они используются только как документация, сообщающая, что именно делает программа.

*Следующие три строки* управляют вычислением случайных чисел. Функция Rnd в каждой строке создает случайное число между 0 и 1 (число с Десятичной точкой), а функция Int умножает число на 10 и округляет его до ближайшего целого. Этими вычислениями создается случайное число от 0 до 9. Затем три случайных числа присваиваются названиям первых трех меток на форме, и после такого присвоения они появляются на экране шрифтом Times New Roman, кеглем 24.

*Последняя группа утверждений* в программе проверяет, нет ли семерки среди данных случайных чисел. Если хоть одна семерка обнаруживается, столбик монет делается видимым, и звуковой сигнал сообщает пользователю о выигрыше. Каждый раз после щелчка на кнопке Вращать вызывается процедура Command1\_Click и выполняются программные Утверждения этой процедуры.

**Сохранение программы.** Вы завершили работу над программой *Счастливая семерка*. Теперь нужно сохранить ее на диске. Visual Basic сохраняет код вашей формы и объектов в одном файле, а «список упаковки» компонентов проекта — в другом. Вы можете использовать эти файлы компонентов в других программных проектах с помощью команды Add File (Добавить файл) в меню проекта. Для сохранения программы в Visual Basic щелкните на команде Save Project As (Сохранить проект как) в меню File (Файл) или на кнопке Save Project (Сохранить проект) на панели инструментов.

#### **Сохраните программу *Счастливая семерка***

1. В меню File (Файл) щелкните на команде Save Project As (Сохранить проект как). Появится диалоговое

окно Save File As (Сохранение файла), подсказывающее имя и место для сохранения вашей формы.

2. В этом диалоговом окне выберите свою папку, если она не выбрана. Файлы проекта вы будете сохранять в своей папке упражнений. 3. Наберите My Lucky( или другое название) в поле ввода File Name (Имя файла) и нажмите клавишу Enter .Форма **Счастливая семерка** будет сохранена под именем My Lucky.frm. Затем появится диалоговое окно Save Project As (Сохранение проекта). 4. Наберите My Lucky и нажмите клавишу | Enter | . Проект **Счастливая семерка** будет записан под именем My Lucky.vbp

Чтобы загрузить этот проект, щелкните на команде Open Project (Открыть проект) в меню File (Файл) и затем щелкните на имени файла My Lucky в диалоговом окне открытия проекта. Вы можете также загрузить проект, с которым недавно работали, щелкнув на имени проекта в нижней части меню File (Файл).

**Запуск программы** 1. Щелкните на кнопке Start (Пуск) на панели инструментов. Программа My Lucky запустится в среде программирования Visual Basic. На экране появится разработанный вами программный интерфейс. 2. Щелкните на кнопке **Вращать**. Программа выберет три случайных числа и покажет их в метках на форме, как представлено на рисунке. Семерка появилась в первой метке, поэтому на экране возник столбик монет, и вы услышали звуковой сигнал. Его характеристики определяются установками в панели управления.3. Щелкайте на кнопке **Вращать** и следите за числами, появляющимися в числовых окнах. 4. Когда вам надоест экспериментировать с программой, щелкните на кнопке **Конец**

**Построение выполняемого файла.** В конце этого занятия вы завершите разработку приложения для Windows -создадите *выполняемый файл*.

Приложения для Windows, созданные с помощью Visual Basic, имеют расширение имени файла .exe и могут быть запущены на любой системе, если в ней установлены Windows 95, Windows 98, Windows NT 3.51 (или более поздней версии), а также необходимые файлы поддержки. (Visual Basic устанавливает эти файлы поддержки - включающие динамически связанные библиотеки и настраиваемые элементы управления - автоматически). Для получения более подробной информации о распространении ваших приложений смотрите «Руководство программиста Visual Basic».

**Создайте выполняемый файл.** 1. В меню File щелкните на команде Make My Lucky.exe. (Visual Basic добавляет имя вашей программы к команде автоматически). Используйте кнопку Options для расширенных установок компилятора. Диалоговое окно содержит: текстовые окна и окна списков, которые вы можете использовать для спецификации имени и местонахождения вашего выполняемого файла на диске. Оно также содержит кнопку Options, на которой вы щелкаете для открытия окна Project Properties (Свойства проекта). Это окно можно использовать для управления значком (пиктограммой) программы и информацией о версии, связанной с файлом. По умолчанию Visual Basic предлагает папку VB 98 для размещения файла. 2. Щелкните на ОК для подтверждения имени файла и его местонахождения по умолчанию. Visual Basic создаст выполняемый файл на диске, в указанном месте. Для запуска этой программы позже под Windows используйте команду Run (Выполнить) в главном меню или дважды щелкните на имени файла в Проводнике (Windows Explorer). Вы можете также создать значок ярлыка для программы *Счастливая семерка* на рабочем столе Windows, щелкнув правой кнопкой на рабочем столе Windows, указав в контекстном меню на строку New (Создать) и затем щелкнув на строке Shortcut (Ярлык). Когда Windows спросит вас о местонахождении вашего файла приложения, щелкните на кнопке Browse (Обзор) и выберите выполняемый файл My Lucky в папке \Vb6Sbs\ VB 98, (или в своей папке). Щелкните на кнопках Open (Открыть), Next (Дальше) и Finish (Закончить), и тогда, Windows поместит на рабочий стол пиктограмму, по которой вы можете дважды щелкнуть для запуска вашей программы. 3. В меню File (Файл) щелкните на команде Exit (Выход). Файл My Lucky и программа Visual Basic будут закрыты.

**Шаг вперед: дополнение к программе** Вы можете перезапустить Visual Basic в любое время и

работать в программном проекте, который вы сохранили на диске. Перезапустите Visual Basic и добавьте специальное утверждение, называемое **Randomize** (Фактор случайности), в программу **Счастливая семерка**.

1. Щелкните на кнопке Start (Пуск) на панели задач Windows, укажите на Programs (Программы), далее укажите на Visual Basic 6.0.

2. Щелкните на закладке Recent (Последний) в диалоговом окне Project (Проект). Появится перечень последних проектов, с которыми вы работали. Поскольку вы только что закончили работу с проектом **Счастливая семерка**, первым проектом в списке должен быть **My Lucky**.

3. Дважды щелкните на **My Lucky** для загрузки программы **Счастливая семерка** с диска. Программа **Счастливая семерка** загрузится с диска, и форма **My Lucky** появится в окне. (Если вы не видите ее, щелкните на **My Lucky.form** (форме **My Lucky**) в окне Project и затем щелкните на кнопке View Object (Просмотр объекта).

4. Дважды щелкните на форме (только не на одном из объектов) для показа процедуры **Form\_Load**. Процедура **Form\_Load** появится в окне кода, добавим утверждение **Randomize** в процедуру **Form\_Load** (Загрузка формы), специальную процедуру, ассоциированную с формой и выполняющуюся каждый раз при запуске программы.

5. Нажмите пробел и наберите **Randomize**.

Утверждение **Randomize** добавлено к программе и будет выполняться каждый раз при ее запуске. **Randomize** использует системные часы для создания истинно случайной стартовой точки, или «зерна» («семени») для утверждения **Rnd**, используемого в процедуре **Command1 - Click**. Вы, может быть, не обратили внимание, но без **Randomize** программа **Счастливая семерка** производит одинаковые строки случайных чисел каждый раз при перезапуске программы. С правильно поставленным утверждением **Randomize** программа будет прокручивать числа случайным образом при каждом запуске. Числа уже не будут следовать по какой-либо схеме.

6. Запустите новую версию **Счастливой семерки** и затем сохраните проект на диске. Если вы планируете использовать новую версию в дальнейшем, вы можете также захотеть создать новый .exe-файл. Visual Basic не обновляет выполняемый файл автоматически, когда вы меняете исходный код.

#### **Контрольные вопросы и задания:**

1. Как в общих чертах осуществляется разработка интерфейса пользователя создаваемого проекта?
2. Как и для чего изменяют свойства объектов управления, размещаемых на форме?
3. Как установить свойства: Группировка, выравнивание, выделение рамки, шрифт, цвет, вытягивание, видимость?
4. Как и где записать команды, благодаря которым ваш проект будет работать?
5. Как сохранить и запустить проект?
6. Что делать, если происходит сбой в работе проекта?
7. Что такое процедура события и где она записывается?
8. Что такое комментарии и как они записываются?
9. Как задать видимость и невидимость рисунка, звук и что означают функции **RND**, **Int**, как присвоить значения меткам?
10. Для чего вы добавили **Randomize** в вашу программу?
11. Как создать выполняемый файл?