Решение задач в Visual Basic

примечание

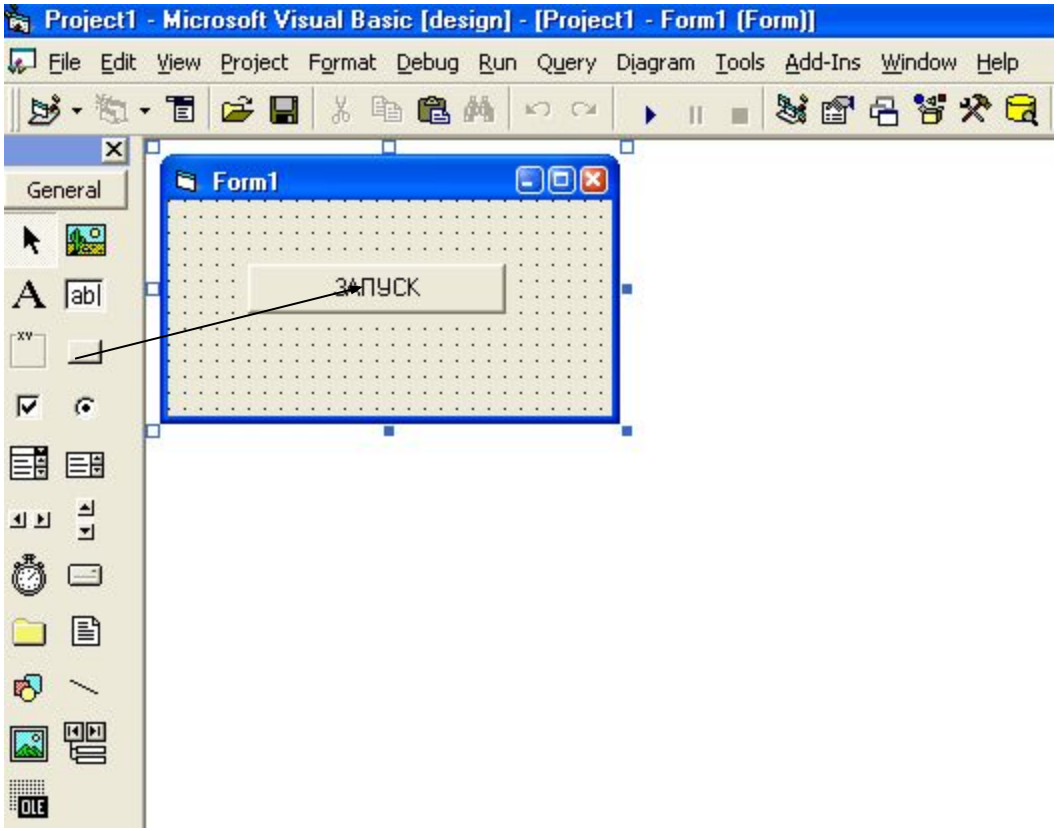
О программе

- Основные понятия о программе
- Основные понятия алгоритмизации
- Основы ООВП
- Интегрированная среда разработки языка Visual Basic
- Интерфейс

Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

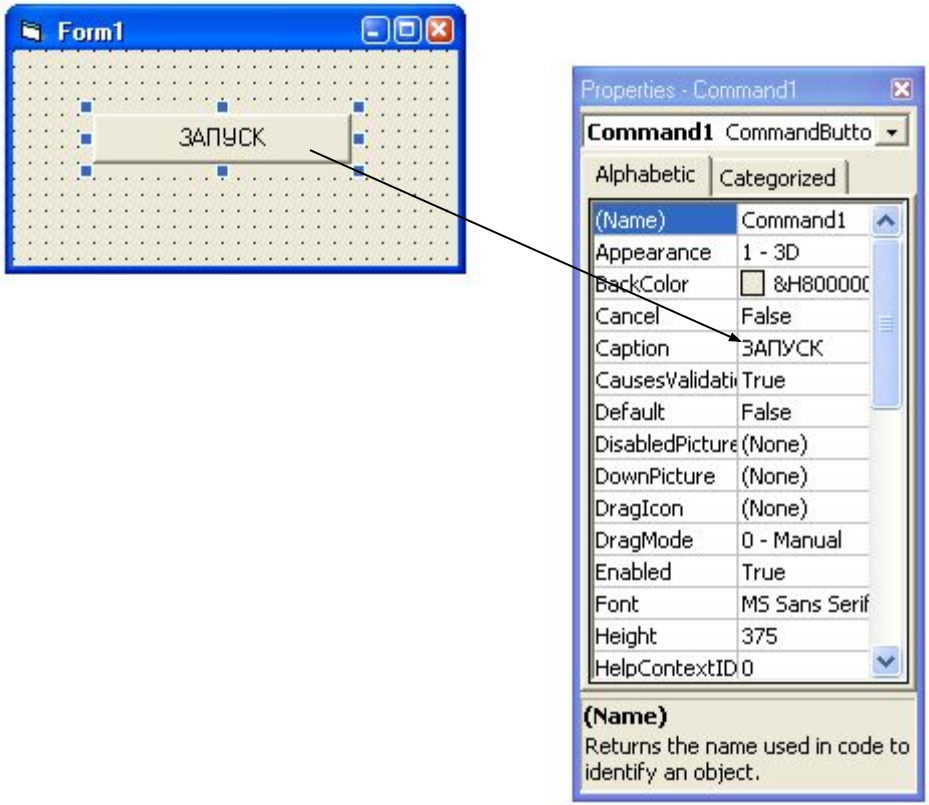
- Порядок работы.
- 1. создать на форме кнопку запуск



Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

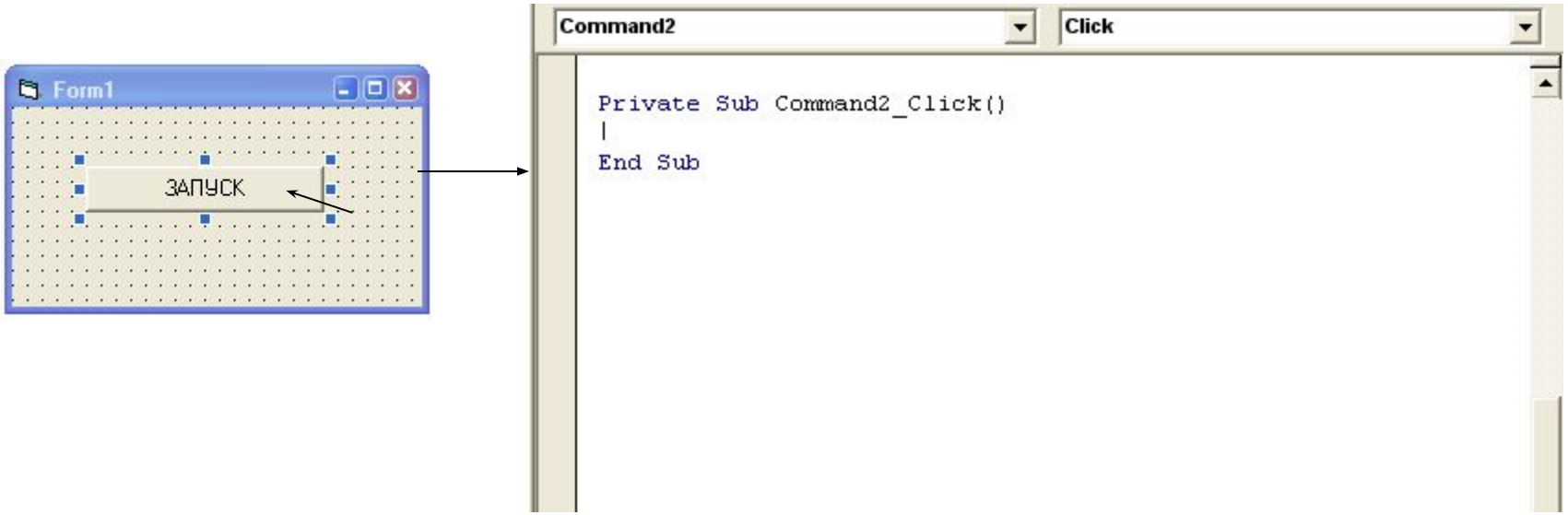
- 2. В свойствах кнопки, в разделе “Captions” вписываем слово “ЗАПУСК”



Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

- 3. Двойным щелчком левой кнопкой мыши кликнем по кнопке “ЗАПУСК” в режиме конструктора формы. После этого откроется конструктор программного кода.



Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

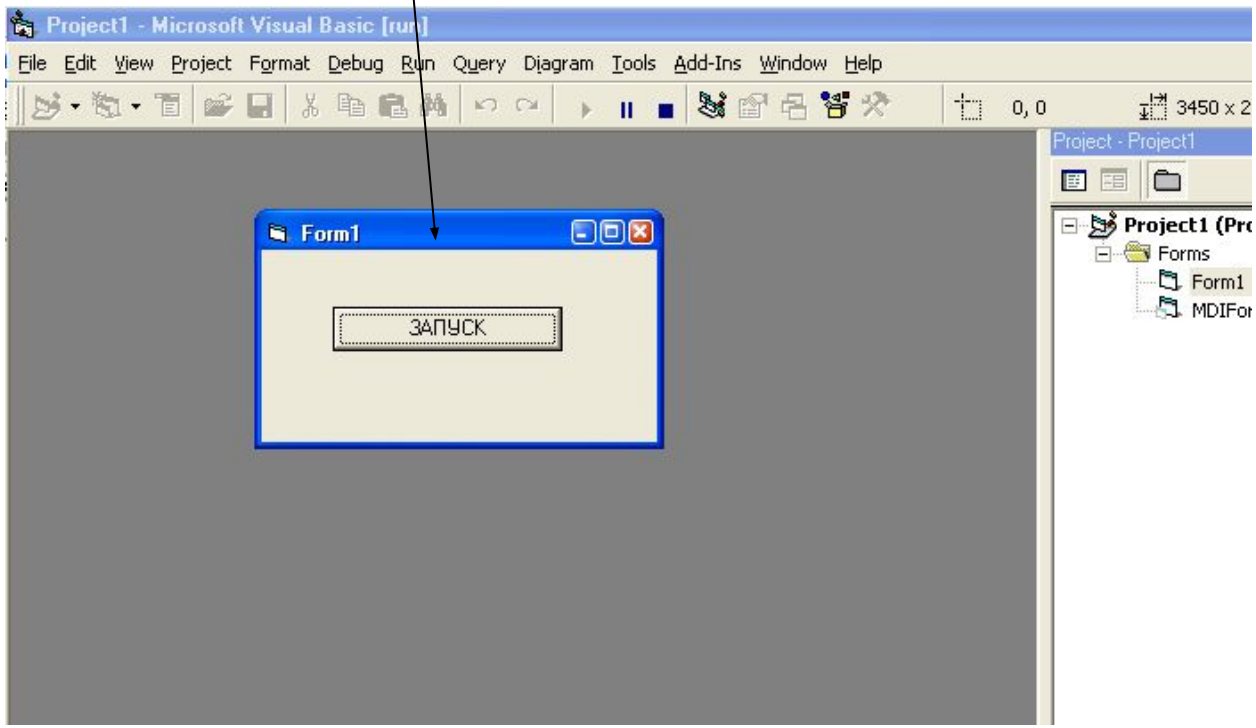
- 4. Между главными строчками начала и конца программного кода, вписываем код программы теста:
- strA = InputBox("Введите Ваше имя:", "Регистрация")
- bytB = MsgBox("Уважаемый(ая) " + strA + ", Вы готовы к проверке знаний?", 36, " Конец регистрации")
- If bytB = 7 Then End
- strC = InputBox("Чему равен 1 байт?:", "Первый вопрос")
- If strC = "8 бит" Then MsgBox "Правильно!", 0, "Первый вопрос" _
- Else MsgBox "Неправильно!", 0, "Первый вопрос": bytN = bytN + 1
- strC = InputBox("Переведите десятичное число 5 в двоичную систему счисления", "Второй вопрос")
- If strC = "101" Then MsgBox "Правильно!", 0, "Второй вопрос" _
- Else MsgBox "Неправильно!", 0, "Второй вопрос ": bytN = bytN + 1
- MsgBox "Уважаемый(ая) " + strA + ", вы сделали " + Str(bytN) + " ошибок/ошибки! ", 0, "Конец опроса «
- 5. В верхнем правом углу окна формы закройте конструктор программного кода крестиком .

Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

- 6. Запускаем получившуюся программу кнопкой “Start” .

- После появится стартовое окно нашей программы.



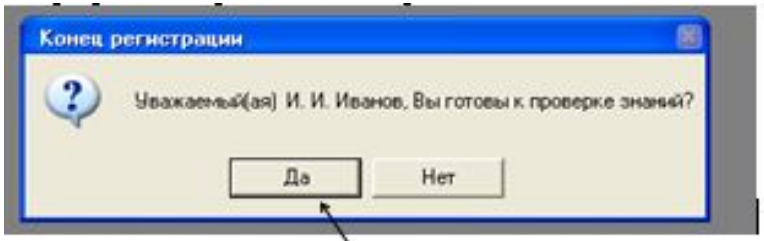
Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

- 7. Кликнуть мышью на кнопке “ЗАПУСК” для запуска программы.
- Появится диалоговое окно программы

- Вписывае
- Данное о ируется следующим оператором: strA = InputBox("Введите Ваше имя:", "Регистрация").

- 8. Программа открывает окно с приветствием.

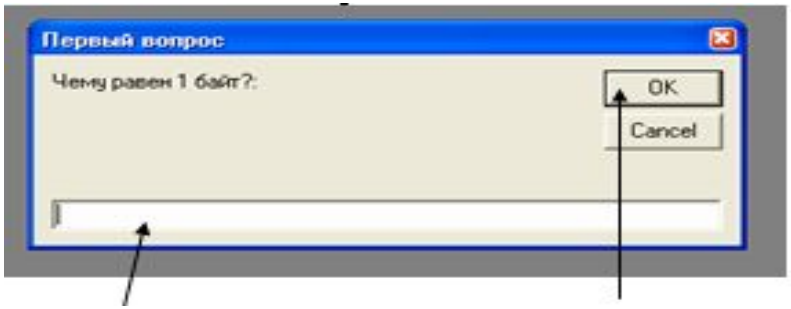


- Клика
- Данное окно в конструкторе программного кода формируется следующим оператором: bytB = MsgBox("Уважаемый (ая) " + strA + ", Вы готовы к проверке знаний?", 36, " Конец регистрации")
- If bytB = 7 Then End

Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

- 9. Появляется окно с вопросом и окном ввода ответа.

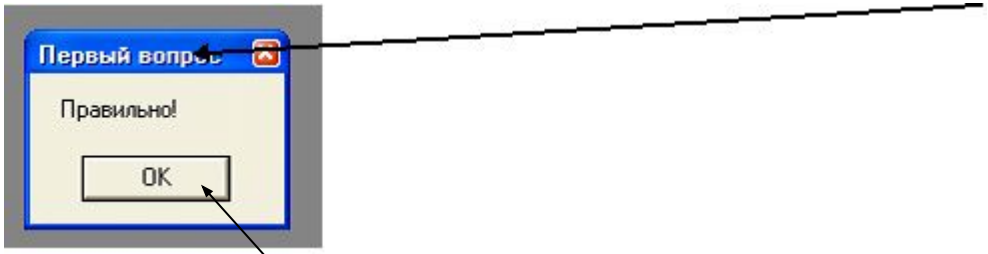


- Вводим ответ и кликаем мышью на кнопке “OK”.
- Данное окно в конструкторе программного кода формируется следующим оператором: strC = InputBox("Чему равен 1 байт?:", "Первый вопрос")

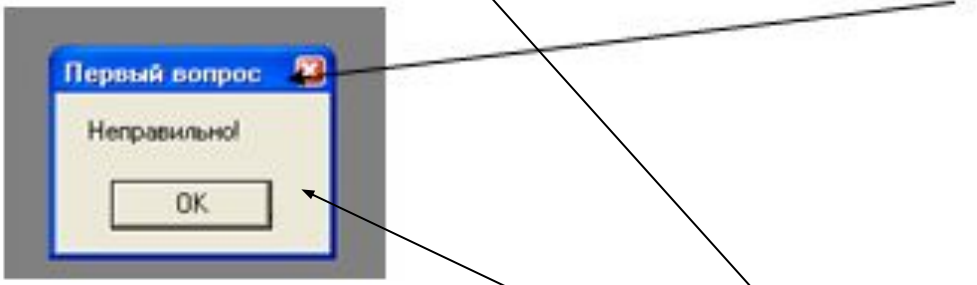
Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

- 10. (а) Если ответ правильный, тогда появляется окно с надписью “Правильно!”.



- Данное окно в конструкторе программного кода формируется следующим оператором: If strC = "8 бит" Then MsgBox "Правильно!", 0, "Первый вопрос"
- (б) Если ответ не верный, тогда появляется окно с надписью “Неправильно!”



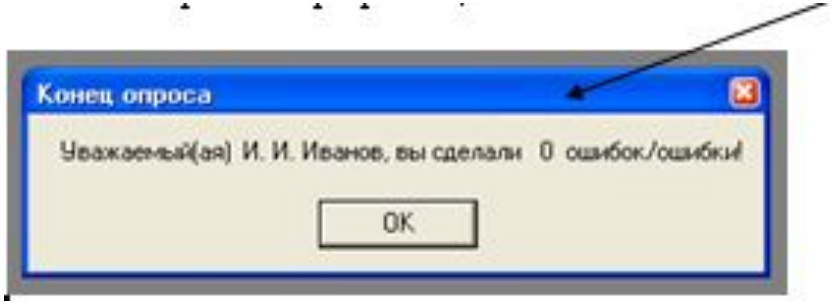
- Данное окно в конструкторе программного кода формируется следующим оператором:
- Else MsgBox "Неправильно!", 0, "Первый вопрос": bytN = bytN + 1
- В любом из перечисленных случаев кликаем мышью на кнопке “OK”.

Задача №1

Создание самостоятельного теста виде диалога пользователя и компьютера.

- 11. Появится диалоговое окно второго вопроса. Далее действия происходят по аналогичному алгоритму.
- Далее программа формируется следующими операторами:
- strC = InputBox("Переведите десятичное число 5 в двоичную систему счисления", "Второй вопрос")
- If strC = "101" Then MsgBox "Правильно!", 0, "Второй вопрос" _
- Else MsgBox "Неправильно!", 0, "Второй вопрос ": bytN = bytN + 1

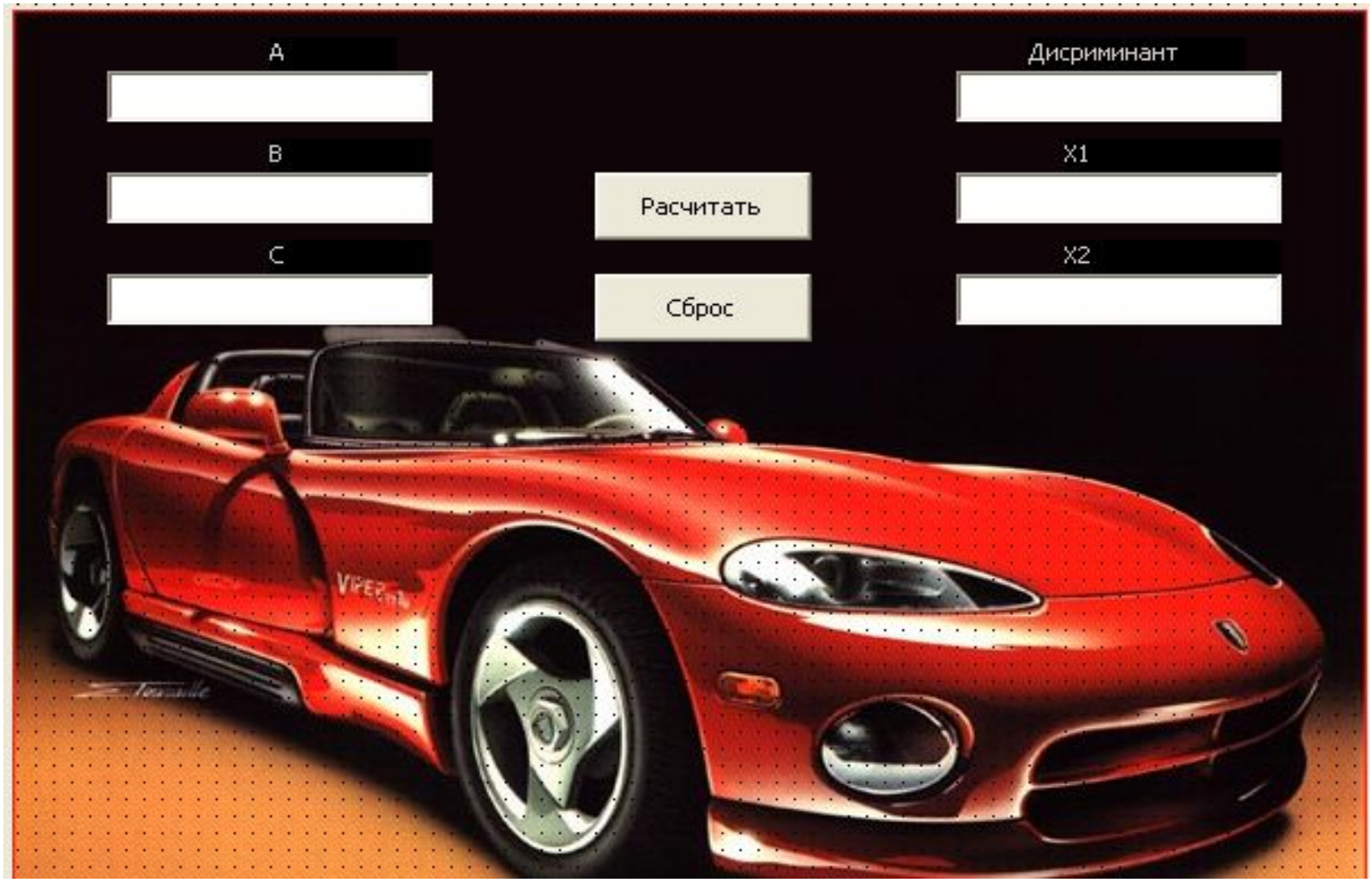
- 12. В конце работы программы, выдаётся диалоговое окно с результатом работы теста.



- Данное окно в конструкторе программного кода формируется следующим оператором: MsgBox "Уважаемый(ая) " + strA + ", вы сделали " + Str(bytN) + " ошибок/ошибки! ", 0, "Конец опроса "

Задача №2

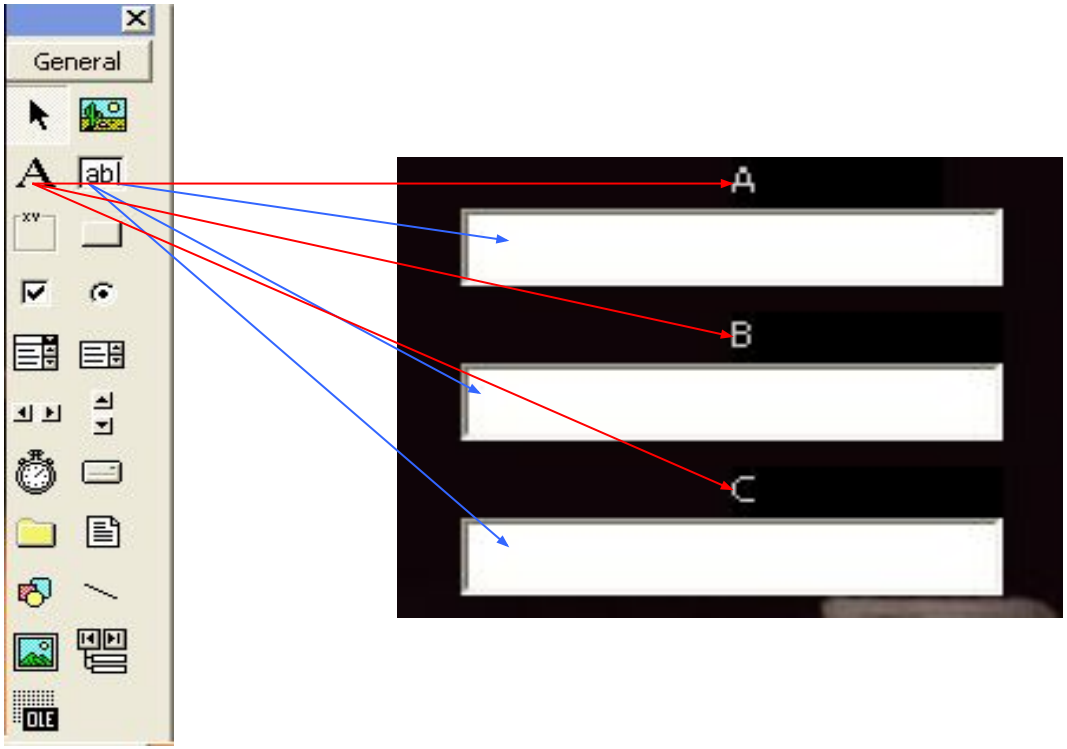
Создание программы по решению Корней квадратного уравнения.



Задача №2

Создание программы по решению Корней квадратного уравнения.

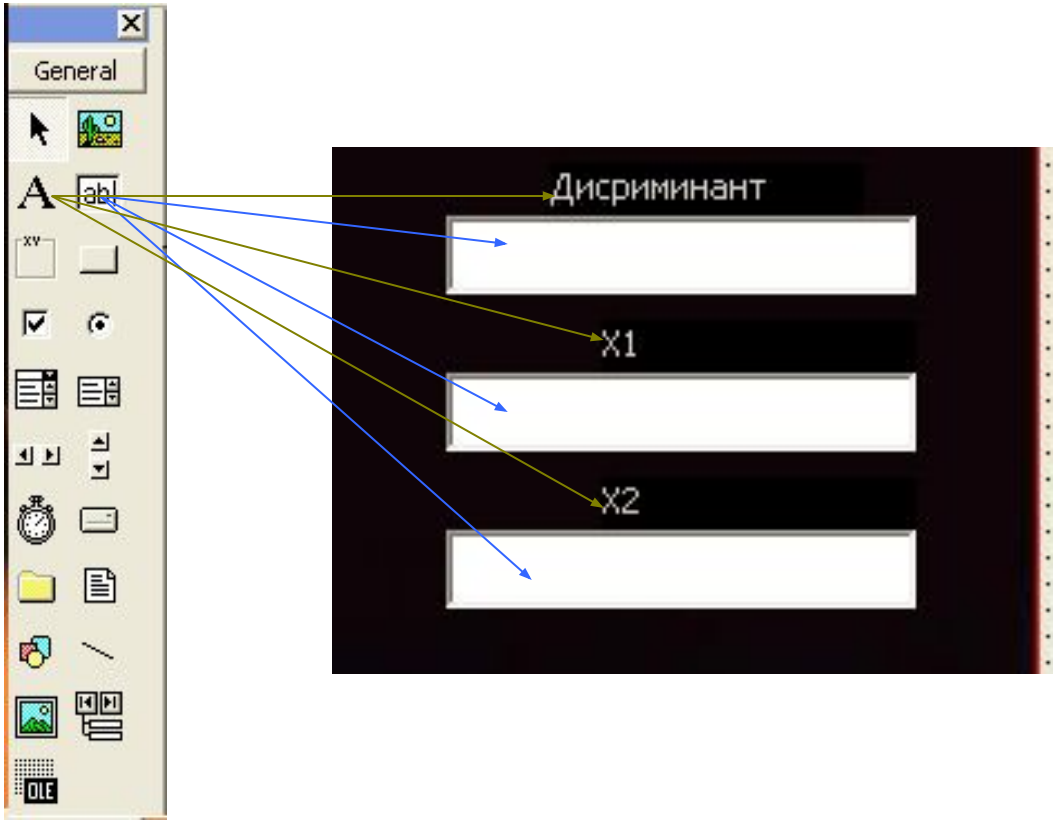
- 1.На форме построить сначала строим три текстовых объекта(Text1, Text2, Text3).
- 2. В свойствах каждого текстового объекта ищем строку “Name” и вписываем последовательно в каждый объект английские буквы “A”, ”B”, ”C”.
- 3. Над каждым тестовым объектом установить объект “Label” , и подписать буквы соответствующие названию окна, т.е. “A”, ”B”, ”C”.



Задача №2

Создание программы по решению Корней квадратного уравнения.

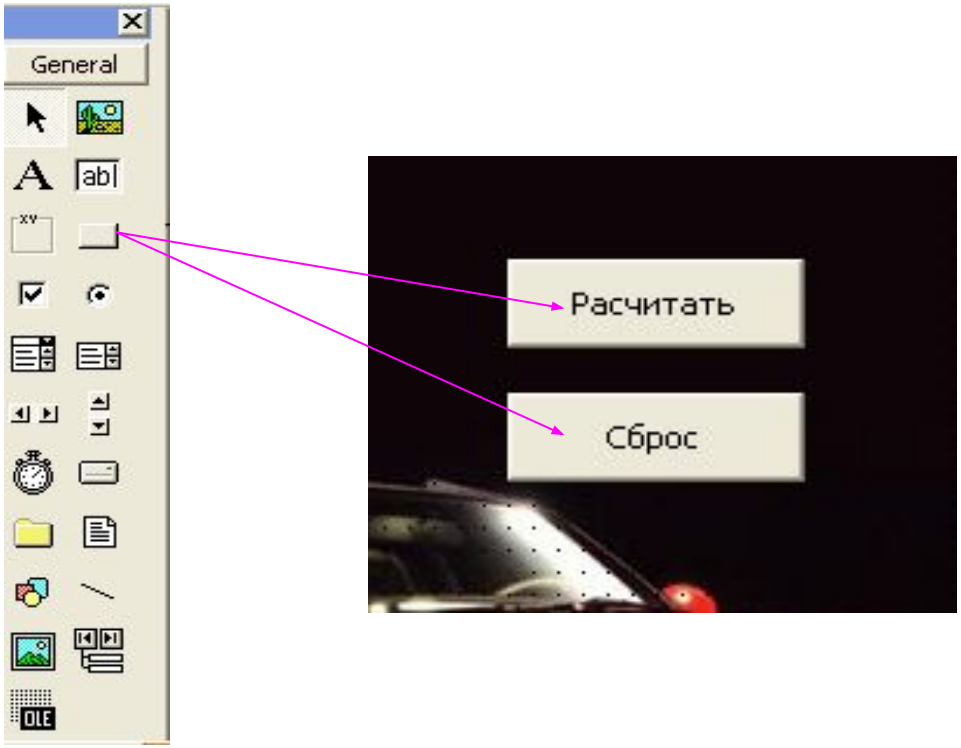
- 4. Аналогично создаём ещё три текстовых объекта для переменных “Дискриминант”, “X1”, “X2”.
- 5. Так же над каждым текстовым объектом устанавливаем объекты “Label” и подписываем соответствующие обозначения, т.е. “Дискриминант”, “X1”, “X2”.



Задача №2

Создание программы по решению Корней квадратного уравнения.

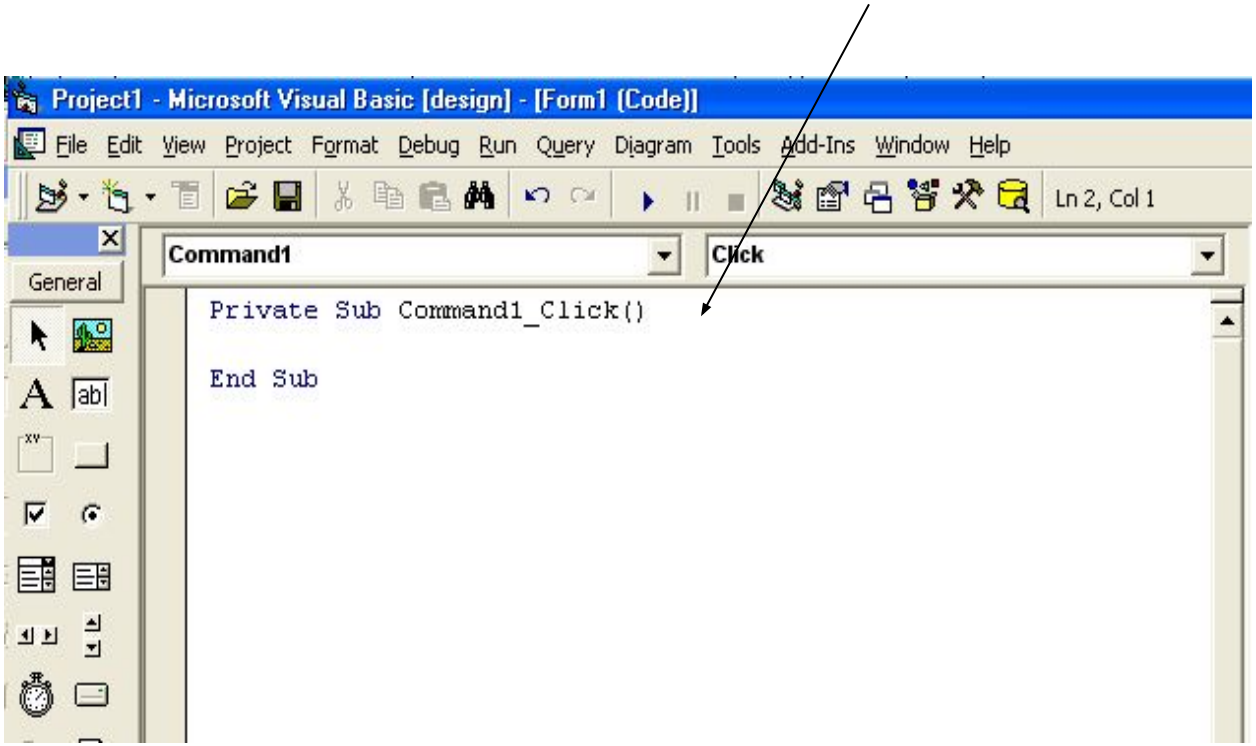
- 6. Строим два объекта “Кнопка”. На кнопках прописываем названия “Рассчитать”, ”Сброс”.



Задача №2

Создание программы по решению Корней квадратного уравнения.

- 7. Двойным щелчком левой кнопкой мыши кликнем по кнопке “Рассчитать” в режиме конструктора формы. После этого откроется конструктор программного кода.

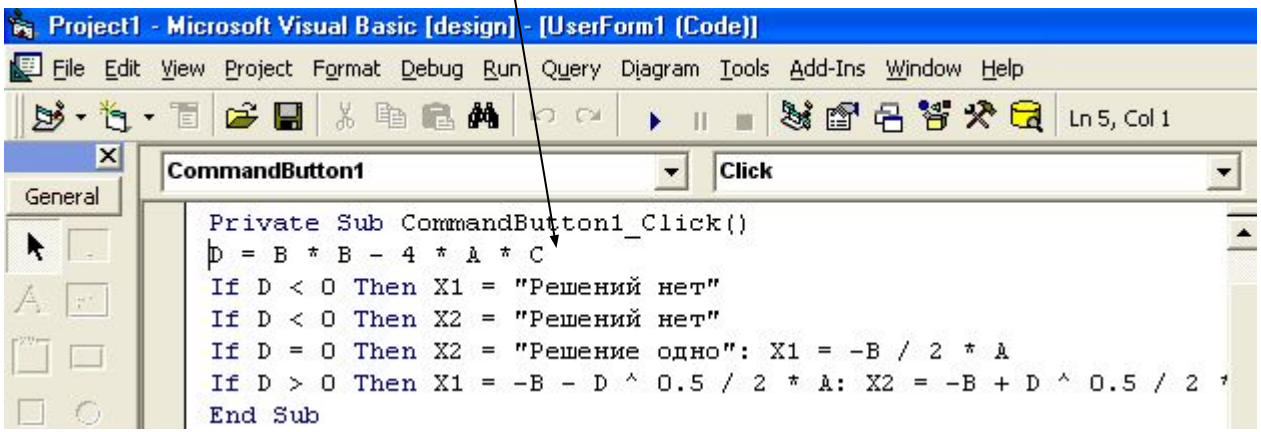


Задача №2

Создание программы по решению Корней квадратного уравнения.

- 8. Вписываем текст, программного кода:

```
D = B * B - 4 * A * C
If D < 0 Then X1 = "Решений нет"
If D < 0 Then X2 = "Решений нет"
If D = 0 Then X2 = "Решение одно": X1 = -B / 2 * A
If D > 0 Then X1 = -B - D ^ 0.5 / 2 * A: X2 = -B + D ^ 0.5 / 2 * A
```



Задача №2

Создание программы по решению Корней квадратного уравнения.

- 9. Двойным щелчком левой кнопкой мыши кликнем по кнопке “Сброс” в режиме конструктора формы. После этого откроется конструктор программного кода.

```
Private Sub CommandButton2_Click()

End Sub
```

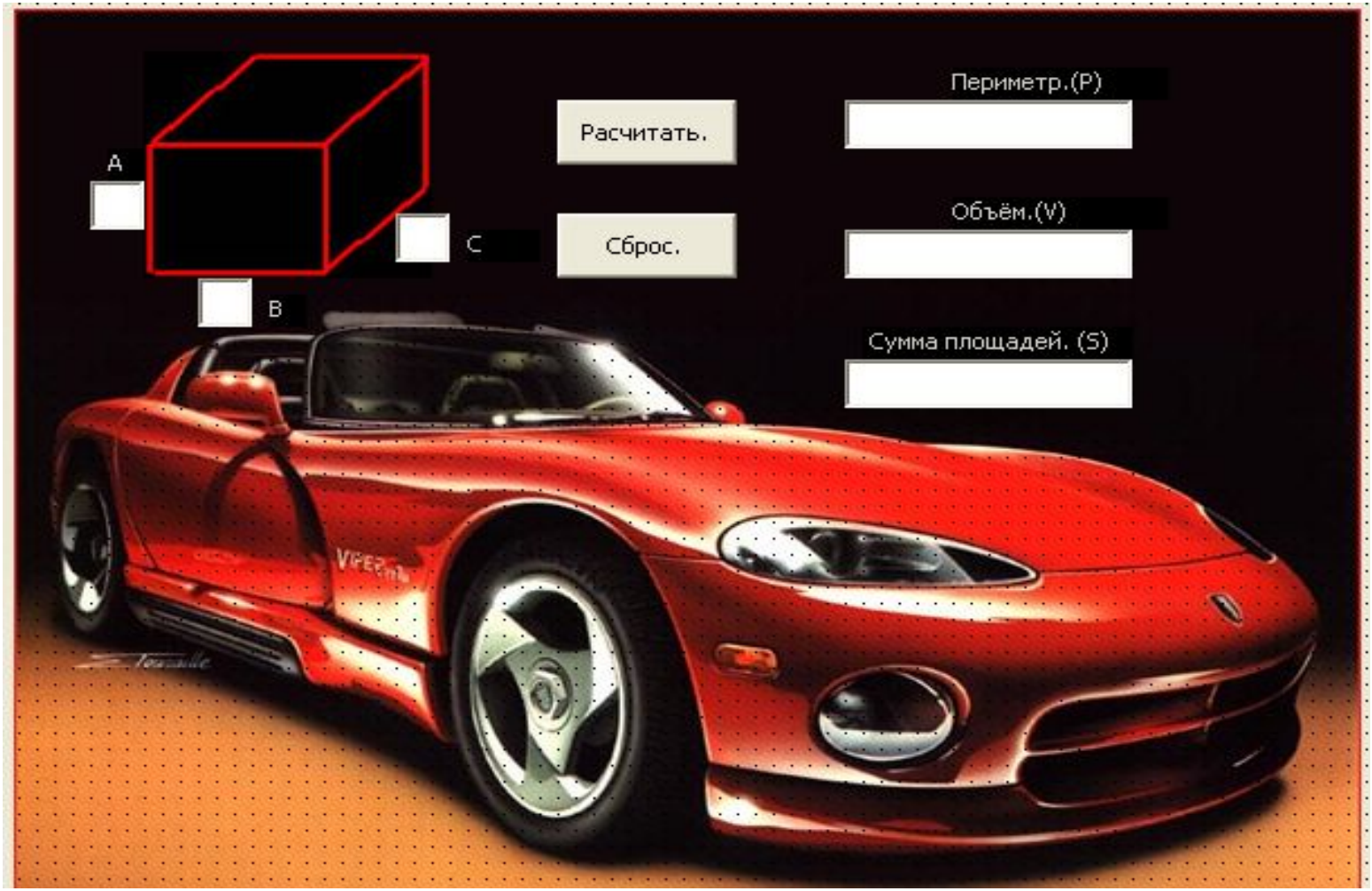
- 10. Вписываем текст, программного кода:

```
X1 = ""
X2 = ""
D = ""
A = ""
B = ""
C = ""
```

- После нажатия этой кнопки, значения всех переменных заменяются пустым значением.
- Запускаем получившуюся программу кнопкой “Start” .

Задача №3

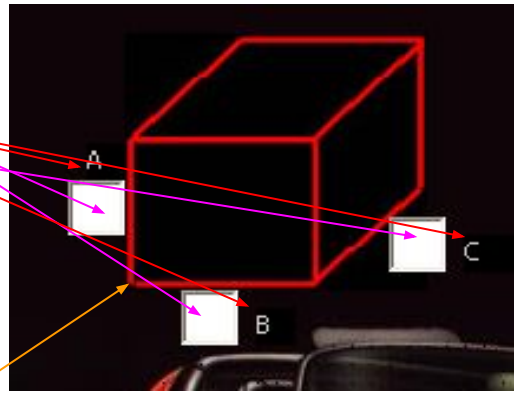
Создание программы по рассчёту
геометрических параметров паралелипипеда.



Задача №3

Создание программы по рассчёту
геометрических параметров паралелипипеда.

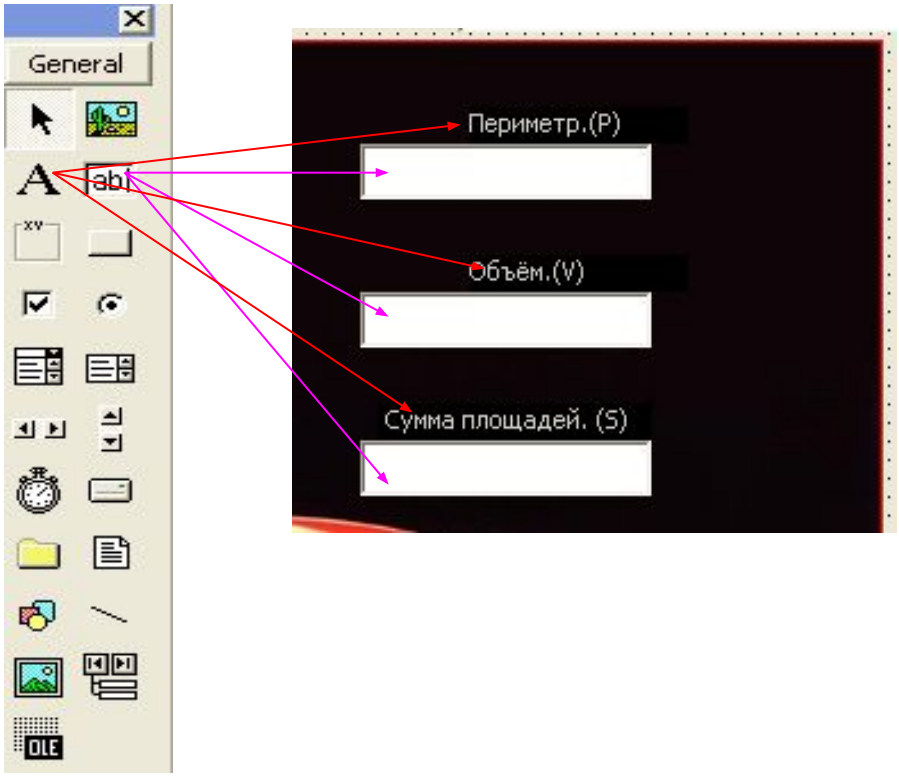
- 1. С помощью объекта “Line” рисуем паралелипипед на форме.
- 2. Рядом с линиями его длины, ширины, и высоты создаём три текстовых окна.
- 3. В свойствах каждого текстового объекта ищем строку “Name” и вписываем последовательно в каждый объект английские буквы “A”, ”B”, ”C”
- Высота – “A”
- Длина – “C”
- Ширина – “B”
- 4. Над каждым тестовым объектом установить объект “Label” , и подписать буквы соответствующие названию окна, т.е. “A”, ”B”, ”C”.



Задача №3

Создание программы по рассчёту
геометрических параметров паралелипипеда.

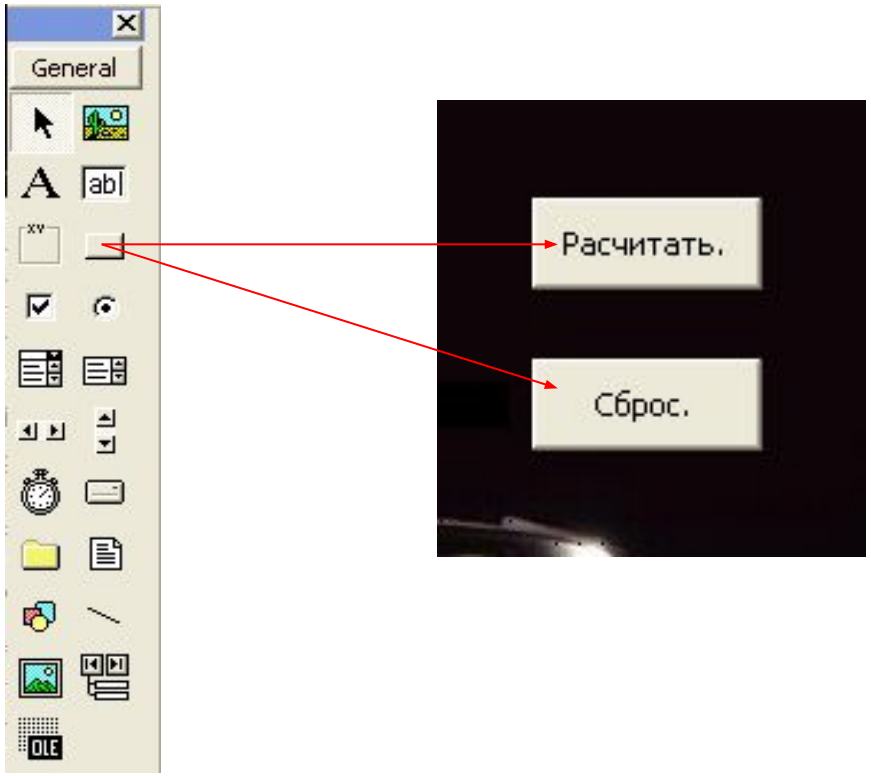
- 5. Создаём три текстовых окна для вывода результатов расчёта периметра, объёма и суммы площадей.
- 6. В свойствах каждого текстового объекта ищем строку “Name” и вписываем последовательно в каждый объект английские буквы “P”, ”V”, ”S”
- 7. Над каждым тестовым объектом установить объект “Label” , и подписать соответствующие названия окон: “Периметр.(P)”, ”Объём.(V)”, ”Сумма Площадей.(S)”



Задача №3

Создание программы по рассчёту
геометрических параметров паралелипипеда.

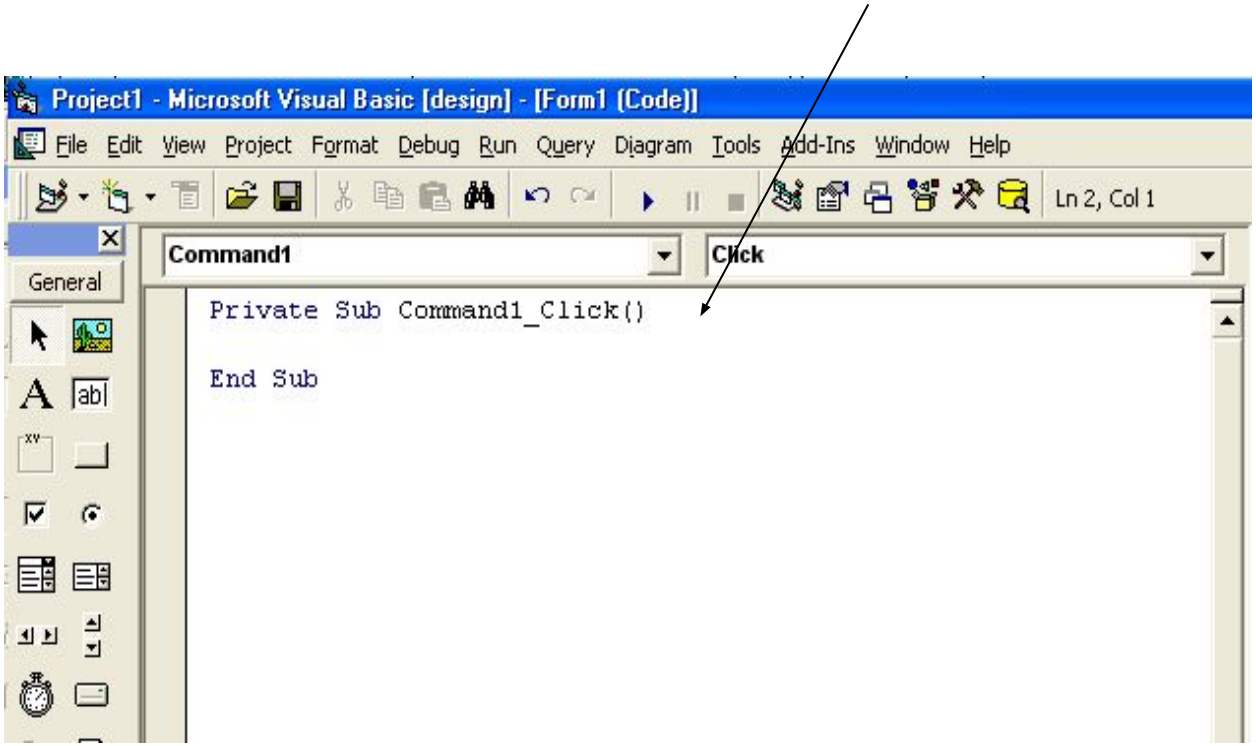
- 8. Строим два объекта “Кнопка”. На кнопках прописываем названия “Рассчитать”, ”Сброс”.



Задача №3

Создание программы по рассчёту
геометрических параметров паралелипипеда.

- 9. Двойным щелчком левой кнопкой мыши кликнем по кнопке “Рассчитать” в режиме конструктора формы. После этого откроется конструктор программного кода.



Задача №3

Создание программы по рассчёту
геометрических параметров паралелипипеда.

- 10. Вписываем текст, программного кода:
- P = 4 * A + 4 * B + 4 * C
- S = 2 * (A * B) + 2 * (A * C) + 2 * (B * C)
- V = A * B * C
- 11. Двойным щелчком левой кнопкой мыши кликнем по кнопке “Сброс” в режиме конструктора формы. После этого откроется конструктор программного кода.

- Private Sub CommandButton2_Click()

- End Sub

- 12. Вписываем текст, программного кода:

```
P = ""
S = ""
V = ""
z = ""
x = ""
n = ""
```

- 13. После нажатия этой кнопки, значения всех переменных заменяются пустым значением.
- Запускаем получившуюся программу кнопкой “Start” .

Примечание

- В процессе изучения объектно-ориентированного программирования рекомендуется установить с CD-ROM свободно распространяемую версию системы программирования Visual Basic (VB5.0 CCE – Visual Basic 5.0 Control Edition)

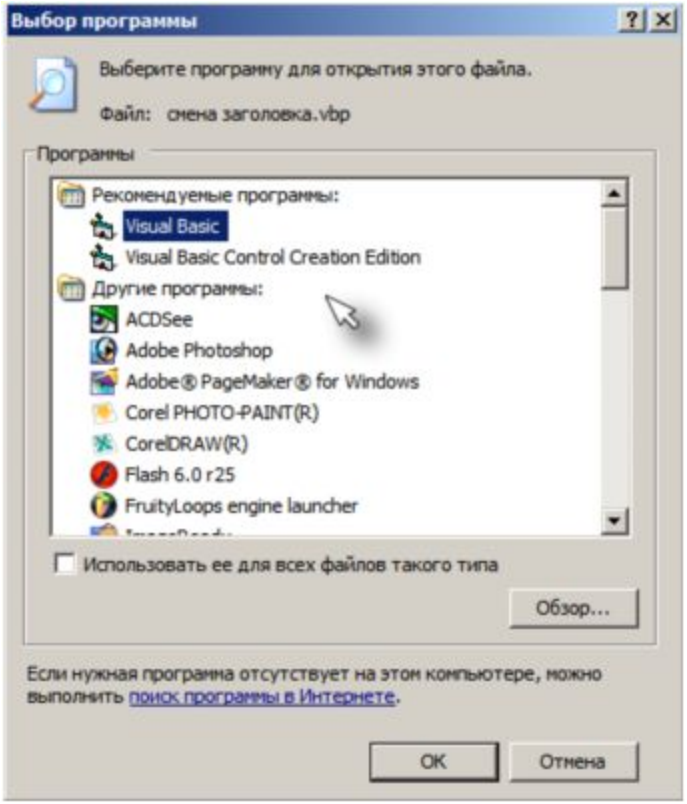
Установка системы программирования Visual Basic

1. Запустить систему программирования VB5.0 CCE
2. Создать новый проект. Для этого ввести команду [File –New Project]. На открывшейся диалоговой панели *New Project* выбрать тип создаваемого проекта *Standard. exe*.
3. После щелчка по кнопке Открыть появится окно интегрированной разработки *Visual Basic*

Внимание!

1. Убедитесь в том, что вы имеете базовые знания о системе программирования Visual Basic.
2. Обязательно перед просмотром задач, ознакомтесь с графой “О программе”.

- Для изучения объектно-ориентированного программирования рекомендуется установить версию системы Visual Basic 6.0
- Для автоматического вызова системы программирования при щелчке по имени файла проекта или при активизации ссылки на него необходимо зарегистрировать тип файлов проектов .VBP.
- Для этого необходимо при нажатой клавише {Shift} щелкнуть правой кнопкой мыши по имени файла. Появится диалоговая панель Открыть с помощью, содержащая список установленных на данном компьютере программ. Необходимо выбрать в этом списке программу, которая будет запускаться при активизации файлов выбранного типа.Щелчок по кнопке ОК завершит регистрацию. В ряде проектов (проект 5.2 «Компьютер в картинках», проект 5.5 «Сортировка строкового массива», в заданиях 5.9, 5.10, 5.11, 5.17, 5.18, 5.19, 5.20) в процессе выполнения проектов происходит обращение к файлам, хранящимся на CD-ROM. Если проекты загружаются из среды Visual Basic или по гиперссылке, то в программном коде нужно указывать полное имя внешнего файла (включая путь к этому файлу). В хранящихся на CD-ROM готовых проектах считается, что CD-ROM имеет логическое имя E:. Если на вашем компьютере CD-ROM имеет другое логическое имя, то необходимо внести соответствующие изменения в программные коды проектов.



Основные понятия алгоритмизации

- Алгоритм и его формальное исполнение.
- Основные типы алгоритмических структур.
  - *Линейный алгоритм*. Команды в одной строке на VB будем разделять двоеточием.
  - *Ветвление*. Выражение, проверяемое в этой структуре назовем логическим выражением. Можно использовать логические связки для получения составных логических выражений. На VB после первого ключевого слова (IF) должно следовать условное выражение, затем ключевые слова THEN, ELSE, за которыми следуют соответствующие последовательности команд. Опреатор условного перехъода может быть записать в одной строке, тогда строка может выглядеть так: **IF a > 0 THEN a = 2 * a ELSE a= a/2**. Если инструкции не умещаются в одной строке, то можно разбить строку на несколько, но ПК должен понимать, что это одна строка. Для этого в конце разрываемой строки, после пробела, ставится знак подчеркивания
- **IF.условие THEN Вариант1 _**
- **ELSE Вариант2**
-

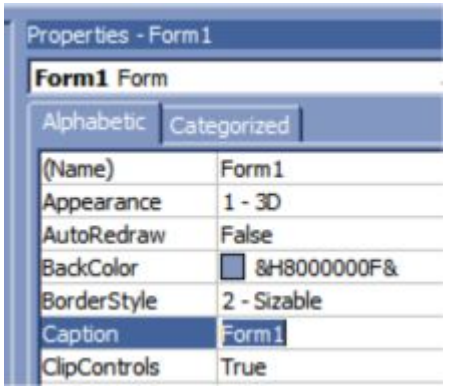
Если же оператор записывают в многострочной форме (полной или неполной форме условия), то на оформление накладывается ряд условий на структуру и еще одно ключевое слово:

- **IF.условие THEN**
- **Вариант1**
- **ELSE**
- **Вариант2**
- **END IF**

- *Выбор. Применяется для реализации ветвлениями со многими вариантами выбора. Выполняет один из нескольких* блоков операторов в зависимости от значения выражения. В эту структуры входит несколько условий, относящихся к одному тест-выражению.
  - **Select Case** тест_выражение **Case** список выражений 1 [блок операторов 1] **Case** список выражений 2 [блок операторов 2]. . . **Case Else** список выражений n [блок операторов n]**End Select**где: тест_выражение - любое числовое или строковое выражение, список выражений - одно или несколько выражений для сравнения с тест_выражением. блок операторов - один или несколько операторов в одной или нескольких строках. **Внимание!** 1) Если список выражений содержит знак отношения (<, >, <=, >=, <> или =), то перед знаком должно стоять ключевое слово IS.
    (например, строка case IS > 7 из примера 1).2) Если список выражений содержит значения из определенного интервала, то этот интервал записывают с помощью ключевого слова TO (например, строка CASE 2 TO 4 из примера 2).*Циклические структуры.*
- В алгоритмическую структуру «цикл» входит серия команд, многократно повторяемая. Такая последовательность называется «телом цикла». Будем разделять циклические структуры на 2 типа:
- - циклы со счетчиком повторений;
- - циклы с условием, выполняемые пока условие истинно.
- На VB5 будем создавать циклы с использованием особых операторов:
- **For V=e1 To e2 Step e3**Тело цикла**Next**Где: e- имя счетчика, e1- начальное значение счетчика, e2- конечное значение счетчика, . Step - в переводе с английского – шаг, e3- величина шага. В операторе FOR...NEXT всегда имеется шаг цикла. По умолчанию он равен единице. Часто бывает так, что действия необходимо повторить, но их количество зависит от каких-либо условий. Такой цикл можно реализовать через две основные структуры, в двух модификациях.
- **Do While** УсловиеТело цикла**LoopDo Until** УсловиеТело цикла**LoopDo** Тело цикла**Loop While** Условие**Do** Тело цикла**Loop Until** Условие*Цикл с предусловием**Цикл с постусловием*Проверка условия выхода из цикла производится с помощью ключевых слов While и Until, но сами ключевые слова придают условию противоположный смысл (пока да и пока нет соответственно)Условие можно поставить в конце, после тела цикла. Реализуют его такие же конструкции, но размещение условия меняет особенности цикла (цикли хотя бы один раз выполнится точно, до проверки условия)

- Приложение на VB5 строятся из объектов, подобно тому, как из блоков и различных деталей строятся дома. Программные библиотеки готовых объектов входят в эти системы программирования. Системы ООВП дают возможность визуализировать процесс создания графического интерфейса разрабатываемого приложения (т.е. работать с объектами, их свойствами, методами и действиями с помощью диалоговых окон).
- Взаимодействие программных объектов между собой и их изменения описываются с помощью программного кода. Создание такого кода в ООВП базируется на использовании алгоритмических структур различных типов, а исполнителями служат программные объекты.
  - ***Классы объектов, экземпляры класса и семейства объектов.***
- Программный объект (ПО) - основная единица в ООВР. ПО вбирает в себя: описывающие его данные (свойства), средства обработки этих данных (методы) . Аналогию можно провести такую же как между существительным- прилагательным и глаголом. ПО обладают свойствами и реагируют на события.
- Классы объектов- это шаблоны, определяющие наборы свойств и событий. В ЯООВП VB5 основными является классы объектов, реализующие графический интерфейс приложения, но в полной версии имеется еще более ста различных классов. Например, ПО «документ» обладает наборами:
- - свойств: имя (Name), полное имя (Full Name) и т.д.
- - методов: открыть документ (Open), сохранить документ (Save), напечатать документ (Print Out)
- - событий: открытие док-та (Document_New ()), закрытие док-та (Document_Close ()) /
- Экземпляр класса – это объект, созданный по шаблону класса объектов и он наследует весь набор свойств, методов, событий данного класса. Каждый экземпляр имеет уникальное для данного класса имя, которое указывается в скобках после названия класса . Например: Document («Проба»)
- Семейство классов- это объект, содержащий несколько объектов, экземпляров одного класса. Например, все открытые в данный момент в приложении Word документов образуют семейство Documents(). Обратиться к объекту, входящему в семейство, можно по его имени.
  - *Объекты: свойства, методы, события.*

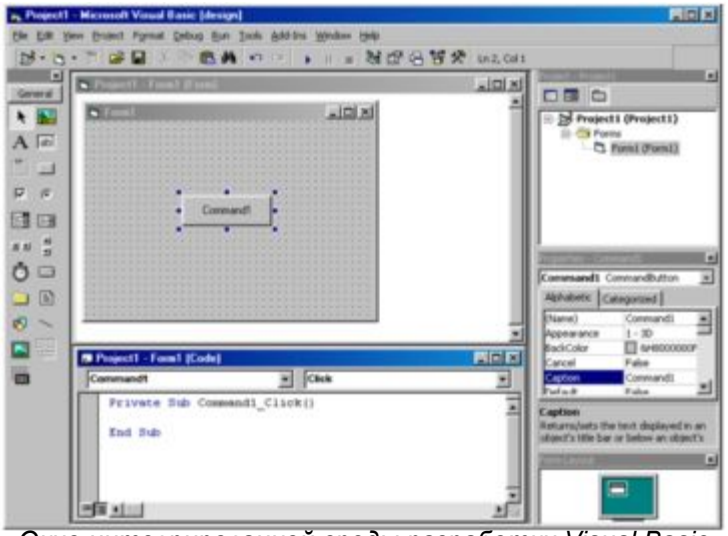
- Свойства объектов (Properties). Каждый объект обладает определенным набором свойств, первоначальное значение которых можно установить с использованием диалогового окна. Значения свойств ПО можно изменять в программном коде. Присвоение свойству объекта нового значения выполняется по следующей команде:
- **Объект.Свойство = ЗначениеСвойства**
- Например, **Selection.Characters(1).Bold = True**
- (Selection- ОП-выделенный фрагмент, Characters – семейство символов, 1- первый символ, Bold- свойство- жирный, True- включено). Можно задать сразу несколько свойств объекта с помощью инструкций
- **With** Объект
- .Свойство1 = ЗначениеСвойства1
- .Свойство2 = ЗначениеСвойства2
- …..
- .Свойство3 = ЗначениеСвойства3
- **End With**
- *Методы объектов (Methods)* применяется для того, чтобы выполнить какую-либо операцию. Многие методы требуют аргументов- параметров выполняемых действий. Обратиться к методам можно по правилу:
- **Объект.Метод арг1: = значение, арг2:=значение**
- *Событие (Events)* представляет собой действие, распознаваемое ПО. Может создаваться пользователем или быть результатом воздействия других ПО. В качестве реакции на события вызывается определенная процедура, которая может изменять значения свойств ПО, вызывать его методы и так далее. ПО Document реагирует на события Open, New, Close
- ПО Selection - на события Cut, Copy, Paste, Delete



Интегрированная среда разработки языка Visual Basic

- 1. Запустить систему программирования VB5.0 CCE
- 2. Создать новый проект. Для этого ввести команду **[File-New Project]**. На открывшейся диалоговой панели New Project выбрать тип создаваемого проекта Standart.exe.
- После щелчка по кнопке Открыть появится окно интегрированной среды разработки Visual Basic

Окно при загрузке



Окно интегрированной среды разработки Visual Basic

- В центре окна *Visual Basic* расположено окно *Конструктор форм*, в котором размещается главный объект проекта *Форма* (Form1*)*. *Форма* является основой для создания графического интерфейса создаваемого проекта и на ней размещаются различные *Управляющие элементы (*Controls*)*.
- Выбор размещаемых на форме *Управляющих элементов* производится с помощью *Панели инструментов* (*ToolBox*), которая обычно размещается в левой части окна приложения. *Стандартная* (*General*) панель инструментов содержит 21 класс управляющих элементов: *Метка* (Label), *Текстовое поле* (TextBox), *Командная кнопка* (CommandButton) и др.

- Окно *Свойства объекта* (*Properties*) предоставляет возможность просмотра и редактирования значений свойств выбранного объекта. В верхней части окна имеется раскрывающийся список всех объектов проекта. В нижней части окна содержится список всех свойств выбранного объекта, а для каждого свойства список его возможных значений. Окно С*войств объекта* располагается в середине правой части окна приложения.
- Окно *Программный код* (*Code*) позволяет просматривать и редактировать программный код проекта. Вызывается окно *Программный код* командой [View-Code] и располагается под окном *Формы*.
- Окно *Проводник проекта* (*Project Explorer*) представляет содержимое проекта, т.е. входящие в него файлы, в форме дерева файлов. В состав проекта входит собственно файл проекта (имеет расширение vbp), файлы форм, которых может быть несколько (имеют расширение frm) и файлы программных модулей (имеют расширение bas). Это окно располагается в верхней правой части окна приложения.
- Окно *Расположение формы* (*Form Layout*) позволяет установить положение формы (фактически окна созданного приложения) на экране монитора в процессе выполнения программы. Окно *Расположение формы* размещается в нижнем правом углу окна приложения.
- Наконец, командой [View-Object Browser] можно вызвать окно *Просмотра объектов* (*Object Browser*], которое содержит в левом списке все доступные классы объектов и объекты, текущего проекта (выделены жирным шрифтом), а в левом списке для выбранного объекта показывает перечень его свойств, методов и событий.

Окно *Свойства объекта*

Указатель Pointer

Рисунок Picture Box

Надпись Label

Текст Text Box

Группа Frame

Кнопка Command Button

Флажок Check Box

Переключатель Option Button

Combo Box

Список List Box

Гориз-ная полоса

Вертикальная полоса прокрутки

Timer

Drive List Box

Dir List Box