Содержание

[Аннотация 5](#_Toc452022794)

[Введение 6](#_Toc452022795)

[1 Теоретические основы обучения учащихся средних общеобразовательных учреждений объектно-ориентированному программированию 9](#_Toc452022796)

[1. 1 Основы объектно-ориентированного программирования средствами Visual Basic 9](#_Toc452022797)

[1. 2 Особенности изучения объектно-ориентированного программирования в курсе информатики общеобразовательной школы 14](#_Toc452022798)

[1. 3 Роль элективных курсов в обучении объектно-ориентированному программированию учащихся средних общеобразовательных учреждений 20](#_Toc452022799)

[2 Методические аспекты обучения учащихся средних общеобразовательных учреждений объектно-ориентированному программированию 25](#_Toc452022800)

[2. 1 Особенности обучения объектно-ориентированному программированию учащихся средних общеобразовательных учреждений 25](#_Toc452022801)

[2. 2 Элективный курс «Основы объектно-ориентированного программирования средствами Visual Basic» 46](#_Toc452022802)

[Заключение 65](#_Toc452022803)

[Список использованных источников 68](#_Toc452022804)

[Приложение 1 74](#_Toc452022805)

# Аннотация

Выпускная квалификационная работа на степень бакалавра содержит 72 страниц, 58 использованных источников, 1 приложение.

Данная бакалаврская работа посвящена рассмотрению особенностей обучения учащихся средних общеобразовательных учреждений объектно-ориентированному программированию.

Бакалаврская работа включает описание теоретических основ обучения учащихся объектно-ориентированному программированию, интерфейса и функционала среды разработки Visual Basic, анализ содержания школьного курса информатики на предмет обучения школьников объектно-ориентированному программированию, уточнение роли элективных курсов в обучении объектно-ориентированному программированию.

В работе предложены методические рекомендации по обучению учащихся объектно-ориентированному программированию с использованием возможностей среды Visual Basic, реализованные в технологической карте урока и элективном курсе.

Объектом исследования является процесс обучения информатике учащихся средних общеобразовательных организаций.

Цель работы – разработка методических рекомендаций по обучению учащихся объектно-ориентированному программированию с использованием возможностей визуальной среды Visual Basic.

В процессе работы использовались методы: целостного анализа, синтеза, обобщения, сравнения, моделирование, анкетирование и др.

В результате исследования определены роль и значение обучения объектно-ориентированному программированию в процессе обучения информатике.

Степень внедрения - частичная.

Область применения - использование в школьной практике на уроках информатики и элективных курсах.

Эффективность - повышение качества знаний учащихся по данной теме.

# Введение

Обучение основам алгоритмизация и программирование является важной составляющей процесса обучения информатике в школе. Особенно актуальным это становится в последнее время, когда программирование становится «новым стандартом грамотности». Изучая основы программирования в школе, ученики не только получают представление о таких понятиях как алгоритм, языки программирования, программа и т.п., о том, как создается программа на разных языках программирования и в различных средах и др. Они могут представить себя в роли программиста и сформировать начальное представление о данной профессии, которая с каждым годом становится все популярнее, а специалисты по программированию становятся все более востребованы [15, с. 18]. Не случайно в последнее время появилось множество различных программ, фондов и конкурсов («IT-START», «У.М.Н.И.К», «IT-прорыв» и др.), организованных при поддержке министерства образования и науки Российской Федерации, направленных на выявление и поддержку молодых программистов и молодежных проектов в сфере информационных технологий.

Следует отметить, что на современном этапе развития информационных технологий совершенствуются методология и технология разработки программного обеспечения, которые, преимущественно, базируются на объектно-ориентированном подходе, поэтому его изучение целесообразно начинать еще в школе. Объектно-ориентированное программирование (ООП) – это методика разработки программ, в основе которой лежит понятие объекта как некоторой структуры, описывающий объект реального мира, его поведение. Задача, решаемая с использованием методики ООП, описывается в терминах объектов и операций над ними, а программа при таком подходе представляет собой набор объектов и связей между ними [21, с. 5].

Анализ учебно-методической литературы, а также учебных программ школьного курса информатики позволяет сделать вывод о том, что на современном этапе развития школьной информатики статус объектно-ориентированного программирования в курсе информатики общеобразовательной школы до сих пор не определен [9, с. 4].

Вышесказанное определяет наличие следующих противоречий:

– между высокими темпами развития науки информатики и относительно медленными темпами создания и внедрения методик обучения, в частности объектно-ориентированному программированию;

– между потребностями современного общества в молодых специалистах в области программирования и недостаточной проработанностью вопросов обучения программированию в школе;

– между потенциальными возможностями развития мыслительной деятельности учащихся в процессе обучения объектно-ориентированному программированию и недостатком методик для реализации этого процесса.

Таким образом, актуальность исследования определяется современными тенденциями в развитии информатики как науки, в частности, переходом к программированию на основе объектно-ориентированной методологии и отсутствием достаточного количества методических разработок, раскрывающих особенности обучения школьников объектно-ориентированному программированию.

В качестве объекта исследования выступает процесс обучения информатике учащихся средних общеобразовательных организаций.

Предметом исследования является методические аспекты обучения учащихся средних общеобразовательных организаций объектно-ориентированному программированию с использованием среды Visual Basic.

Целью исследования являлась разработка методических рекомендаций по обучению учащихся объектно-ориентированному программированию с использованием возможностей визуальной среды Visual Basic.

Для достижения поставленной цели потребовалось решить следующие задачи:

– рассмотреть основы объектно-ориентированного программирования;

– изучить характеристики и компоненты интегрированной среды разработки приложений Visual Basic;

– выявить особенности изучения объектно-ориентированного программирования в курсе информатики общеобразовательной школы;

– охарактеризовать роль элективных курсов в обучении объектно-ориентированному программированию учащихся средних общеобразовательных учреждений;

– выявить особенности обучения объектно-ориентированному программированию на уроках информатики;

– разработать элективный курс «Основы объектно-ориентированного программирования средствами Visual Basic».

Практическая значимость исследования заключена в разработке методических рекомендаций по обучению объектно-ориентированному программированию с использованием среды Visual Basic.

В процессе анализа и обобщения информации в рамках проводимого исследования применялись следующие методы: анализ программных кодов, сравнение, синтез, обобщение, моделирование, анкетирование, анализ литературы и др.

Выпускная квалификационная работа состоит из введения, двух глав, заключения и списка использованных источников.

# 1 Теоретические основы обучения учащихся средних общеобразовательных учреждений объектно-ориентированному программированию

## 1 Основы объектно-ориентированного программирования средствами Visual Basic

В настоящее время объектно-ориентированное программирование является одной из тех технологий программирования, которые позволяют достичь простоты структуры и управляемости достаточно объемных программных систем.

Объектно-ориентированное программирование (ООП) *–* это методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса; а классы образуют иерархию наследования. Объектно-ориентированное программирование использует в качестве базовых элементов объекты, а не алгоритмы [31].

Следует отметить, что появление ООП в свое время произвело настоящую революцию в программировании: до появления ООП программа представляла собой логически единый код с ярко выраженными функциональными зависимостями (процедурный подход). ООП предполагает, что приложение строится из набора независимых по своему внутреннему устройству модулей, поэтому объектно-ориентированного программирование стало  большим шагом вперёд в развитии технологий программирования. В частности, программный код перестал быть «плоским», что позволяет разработчику оперировать не просто процедурами и функциями, а объектами, обладающими совокупностью некоторых свойств. Таким образом, объектно-ориентированное программирование – это методика разработки программ, в основе которой лежит понятие объекта как некоторой структуры, описывающей объект реального мира, его поведение. Задача, решаемая с использованием методики объектно-ориентированного программирования, описывается в терминах объектов и операций над ними, а программа при указанном подходе представляет собой набор объектов и связей между ними. Объектно-ориентированное программирование является естественной эволюцией более ранних нововведений в разработке языков программирования. Оно является более модульным и более абстрактным, чем предыдущие попытки абстрагирования данных и переноса деталей программирования на внутренний уровень [21, с. 16]. Также подчеркнем, что одним из существенных преимуществ объектно-ориентированного программирования является возможность многократного использования программного кода, например, при создании класса можно порождать от него новые классы и изменять их свойства и функциональное назначение.

Системы ООП дают возможность визуализировать процесс создания графического интерфейса разрабатываемого приложения, то есть позволяют создать объекты и задавать значения их свойств с помощью диалоговых окон системы программирования. Взаимодействия объектов между собой и их изменения описываются с помощью программного кода. Создание программного кода базируется на исполнении алгоритмических структур различных типов. Основной единицей в ООП является объект, который объединяет в себе как описывающие его данные, так и средства обработки этих   
данных, т.е. программные объекты обладают свойствами, могут, использовать методы и реагирует на события. Классы объектов являются шаблонами, определяющими наборы свойств, методы и событий. По этим шаблонам создаются объекты. Каждый из классов обладает специфическим набором свойств, методов и событий [23].

Объект, созданный по «шаблону» класса объектов, является экземпляром класса и наследует весь набор свойств, методов и событий данного класса. Значение свойств можно изменить в программном коде. Для того чтобы объект выполнил какую-либо операцию, необходимо применить метод, которым он обладает. Методы имеют аргументы, позволяю задать параметры выполняемых действий. Событие представляет собой действие, распознаваемое объектом. Событие может создаваться пользователем или быть результатом взаимодействия других программных объектов. В качестве реакции на событие вызывается определенная процедура. Отметим, что каждый экземпляр класса имеет уникальное для данного класса имя. Экземпляры обладают одинаковым набором свойств, но значения свойств у них могут отличаться.

Остановимся на рассмотрении ключевых моментов объектно-ориентированного программирования на основе использования системы программирования Visual Basic.

Visual Basic (VB) развился из первоначального языка Basic и в настоящее время содержит несколько сотен команд, функций и ключевых слов, многие из которых связаны непосредственно с графическим интерфейсом пользователя GUI (Graphical User Interface) операционной системы Windows [7]. Интегрированная среда разработки приложений Visual Basic, обладая простыми в обращении средствами визуального проектирования, позволяет в полной мере использовать преимущества графической среды Windows и быстро конструировать эффективные приложения. Причем процесс разработки заключается не столько в написании программного кода, сколько в проектировании приложения. Приложение формируется средствами графического редактирования, что позволяет свести процесс создания программного кода к минимуму.

Указанная среда программирования входит в состав пакета Microsoft Visual Studio. В настоящее время последней версией данного пакета является Microsoft Visual Studio 2015. Express версия является официально бесплатной и ее можно скачать, перейдя по ссылке https://www.visualstudio.com/vs-2015-product-editionsю. Также можно использовать систему программирования Basic, входящую в OpenOffice.org, доступную по ссылке http://ru.openoffice.org/.

Все объекты в языке VB, включая формы и элементы управления, имеют собственные атрибуты, описывающие их. Эти атрибуты называются свойствами. Свойства формы и любых элементов управления на форме можно задавать в интегрированной среде разработки с помощью окна «Свойства» или с помощью написания кода в редакторе кода [8, с 11].

Система программирования Visual Basic, интерфейс которой представлен на «Рис. 1.1», состоит из текстового редактора для написания кода программ и конструктора форм. Разработка интерфейса программы реализуется с помощью конструктора форм. Для создания интерфейса Visual Basic предоставляет готовые объекты, обладающие определенными свойствами, которые можно изменять. Код программы и ее интерфейс объединяется в общий проект, который запускается на выполнение непосредственно из среды программирования Visual Basic или как исполняемый файл, компилируемый данной средой.

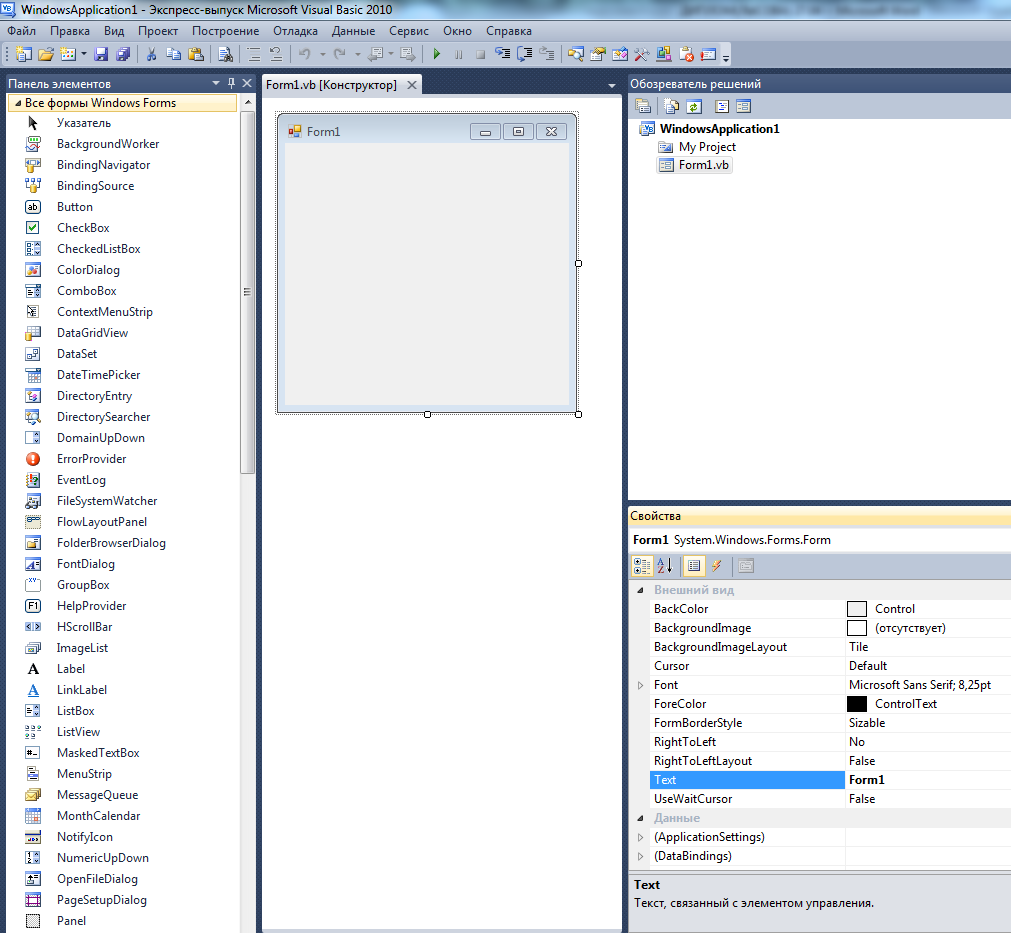


Рис. 1.1 Окно программы Visual Basic

Окно конструктора форм находится в центре экрана и содержит форму (Form). В новый проект сразу добавляется форма с именем Form1. Команды меню «Вид» открывают окна, если оно было закрыто и выводит его поверх других окон. Окно свойств содержит список свойств и их значений для выделенного объекта (в данном случае для формы). Объект можно выбрать также в списке под строкой заголовка.

Окно проводника (Обозреватель решений) отображает все составные части проекта: формы и модули, не связанные с формами.

Окно с набором управляющих элементов (Панель элементов) находится обычно слева от формы. В процессе проектирования выбираются необходимые элементы и перемещаются на форму. Приведем в таблице основные элементы управления и их функциональное предназначение (Таблица 1.1).

Таблица 1.1

**Названия и функциональное предназначение элементов**

|  |  |
| --- | --- |
| Название элемента | Назначение |
| Button | При щелчке возникает событие |
| CheckBox | Позволяет пользователю выбрать или удалить соответствующий параметр |
| ComboBox | Отображает редактируемое текстовое поле и раскрывающийся список допустимых значений |
| Label | Предоставляет элементу управления текст описания либо информацию во время выполнения |
| ListBox | Отображает список, из которого пользователь может выбирать элементы |
| PictureBox | Отображает изображение |
| ProgressBar | Отображает индикатор выполнения операции |
| RadioButton | Позволяет выбрать определенный параметр из группы в комбинации с другими флаговыми кнопками |
| RichTextBox | Обеспечивает дополнительные возможности ввода и редактирования текста |
| TextBox | Позволяет вводить текст и обеспечивает редактирование нескольких строк |

Для создания приложения средствами Visual Basic необходимо:

1. Подготовить интерфейс окна, с которым будет работать пользователь.

2. Определить, какие события будут обрабатывать элементы управления внутри окна.

3. Разработать процедуры обработки для этих событий (и вспомогательные процедуры, если это необходимо).

Остановимся на рассмотрении проблемы обучения объектно-ориентированному программированию в курсе информатики общеобразовательной школы.

## 2 Особенности изучения объектно-ориентированного программирования в курсе информатики общеобразовательной школы

Исследованию методологии объектно-ориентированного программирования посвящены многие научные работы известных зарубежных ученых: Буча Г., Грэхема И., Кея А., Максимчука Р., Мейера Б., Рамбо Дж., Хьюстона К., Энгла М., Якобсона А., Янга Б. и др. Исходя из основных положений объектной модели, разработанной Бучем Г., под термином «объектно-ориентированное проектирование» понимают метод, сочетающий процесс объектно-ориентированной декомпозиции и систему обозначений для представления логической и физической, статической и динамической модели проектирования системы [1, с. 13].

Вопросам обучения основам ООП посвящены научно-методические исследования многих отечественных и зарубежных ученых, анализ которых позволил выделить три основных подхода: изучение принципов ООП на примере построения графического интерфейса пользователя (Бабушкина И.А., Иванова Г.С., Ничушкина Т.Н., Пугачев Е.К.); рассмотрение ООП как дополнения к структурному программированию (Аржанов И.Н., Мещерякова Н.А.); обучение ООП на основе объектно-ориентированного проектирования (Бадд Т., Буч Г.). Первые два подхода достаточно подробно рассмотрены в научно-методической литературе, в то время как третий подход – обучение ООП на основе объектно-ориентированного проектирования и использование визуального языка моделирования в отечественных исследованиях – представлен недостаточно [9].

Отметим, что «школьная информатика» в России начиналась с алгоритмизации и программирования как с основной темы курса. В то время даже был провозглашен лозунг: «Программирование – это вторая грамотность». Эта тема изучалась и в безмашинном варианте, и с компьютерной поддержкой на БК (бытовом компьютере) и Yamaha - компьютерах, первыми появившимися в отечественных школах. Основным программным обеспечением данных компьютеров был встроенный язык программирования Бейсик [13, с. 5]. Однако последующие годы в обучении информатике в школе характеризовались уменьшением количества часов на изучение основ алгоритмизации и программирования в старшей школе, что было связано с развитием школьной информатики как самостоятельного предмета и бурным развитием информационных технологий. Чрезмерное увлечение «пользовательской компонентой» вытеснило изучение указанных вопросов не только из некоторых профильных курсов, но даже из ряда учебников базового курса, например учебника информатики и ИКТ Макаровой Н.В. (2009 г.). При явном улучшении оснащения школ компьютерной техникой уровень знаний выпускниками основ алгоритмизации и программирования заметно снизился.

Похожие проблемы появились во многих странах. Например, несмотря на перенасыщенность американских школ компьютерной техникой, на всеобщую доступность компьютеров и сети Интернет, нет положительных сдвигов в уровне общей подготовки учащихся, в том числе и по информатике. Полное отсутствие представлений об алгоритмизации и технологиях программирования у выпускников школ вызывает беспокойство преподавателей колледжей и университетов и приводит к изменению учебных планов в сторону продолжительности обучения на вводных курсах. По мнению зарубежных ученых и специалистов в области образования вопросы, связанные с алгоритмизацией и программированием, являются фундаментальными и обязательно должны изучаться на вводных курсах информатики вне зависимости от дальнейшего профиля обучения [15, с. 65]. Более подробно рассмотрим вопросы, связанные с обучением объектно-ориентированному программированию в школьном курсе информатики.

Программирование, как предметная область – это раздел информатики, задача которого – разработка программного обеспечения ЭВМ. В узком смысле слово «программирование» обозначает процесс разработки программы на определенном языке программирования. Разработку средств системного программного обеспечения и систем программирования принято называть системным программированием, а разработку прикладных программ называют прикладным программированием [28, с. 7].

Методика преподавания программирования на разных ступенях образования, в том числе в старшей и средней школе, отрабатывалась многие годы. Существуют различные парадигмы программирования, и преподавание каждой из них имеет свои особенности. К основным парадигмам программирования относят:

– процедурное программирование (языки и системы программирования, реализующие данную парадигму: Паскаль, Бейсик, Фортран, Си, Ассемблеры и др.);

– логическое программирование (языки и системы программирования, реализующие данную парадигму: Пролог и др.);

– функциональное программирование (языки и системы программирования, реализующие данную парадигму: Лисп и др.);

– объектно-ориентированное программирование (языки и системы программирования, реализующие данную парадигму: Visual Basic, Смолток, Си++, Делфи и др.) [22, с. 35].

Процесс изучения и практического освоения программирования можно разделить на три части:

1. Изучение методов построения вычислительных алгоритмов;
2. Изучение языка программирования;
3. Изучение и практическое освоение определенной системы программирования.

Основы объектно-ориентированного программирования рассматриваются в учебнике «Информатика и ИКТ» для 9 класса (базовый уровень) автора Угринович Н. Д. Тема «Алгоритмизация и основы объектно-ориентированного программирования» занимает особое место в данном учебнике. В данной теме рассматриваются все основные алгоритмические структуры и их кодирование на трех языках программирования:

– алгоритмическом языке OpenOffice Basic, который входит в свободно распространяемое интегрированное офисное приложение OpenOffice Basic в операционных системах Windows и Linux;

– объектно-ориентированном языке Visual Basic, который бесплатно распространяется по лицензии корпорации Microsoft;

– объектно-ориентированном языке Gambas (аналог – Visual Basic в операционной системе Linux), который распространяется по лицензии компании AltLinux.

На изучение темы «Основы алгоритмизации и объектно-ориентированного программирования» рекомендовано 20 учебных часов, причем рассматриваются три языка программирования: OpenOffice.org Basic в операционных системах Windows и Linux, объектно-ориентированный язык Visual Basic в операционной системе Windows и объектно-ориентированный язык Gambas в операционной системе Linux. Использование языка OpenOffice.org Basic согласуется с ОГЭ и ЕГЭ, а объектно-ориентированные языки Visual Basic и Gambas используют современную технологию программирования, хотя процедурное программирование входит в технологию объектно-ориентированного программирования [42, с. 45].

Освоение данной темы предполагает изучение главы «Алгоритм и его формальное исполнение», в которой рассматриваются свойства алгоритма и его исполнители, блок-схемы алгоритмов, выполнение алгоритмов компьютером; главы «Кодирование основных типов алгоритмических структур на объектно-ориентированных языках и алгоритмическом языке», где рассматриваются такие понятия, как линейный алгоритм, алгоритмическая структура «ветвление», алгоритмическая структура «выбор», алгоритмическая структура «цикл», а так же глав «Переменные: тип, имя, значение», «Кодирование основных типов алгоритмических структур на языках объектно-ориентированного и процедурного программирования», «Арифметические, строковые и логические выражения», «Функции в языках объектно-ориентированного и алгоритмического программирования», «Основы объектно-ориентированного визуального программирования», «\*Графические возможности объектно-ориентированного языка программирования Visual Basic» [42, с. 47].

К теме «Алгоритмизация и основы объектно-ориентированного программирования» также приведены практические задания, содержание которых представляет собой создание проектов приложений: «Переменные», «Калькулятор», «Строковый калькулятор», «Даты и время», «Сравнение кодов символов», «Отметка», «Коды символов», «Слово-перевертыш», «Графический редактор», «Системы координат», «Анимация».

Тема «Алгоритмизация и основы объектно-ориентированного программирования» также входит в содержание учебника «Информатика и ИКТ» того же автора для 10 класса, но для профильного уровня [43, с. 6].

В рамках данной темы учащиеся изучают основы и принципы алгоритмизации и программирования, а не какой-либо один язык программирования. Проекты могут выполняться на четырех языках объектно-ориентированного программирования (Visual Basic .Net, Delphi, Visual C# и Visual J#). Количество рекомендованных часов на данную тему – 50.

Методика преподавания по данному учебнику предполагает широкое использование метода проектов, что позволяет дифференцировать и индивидуализировать обучение [42, с. 36].

В главу «Алгоритмизация и основы объектно-ориентированного программирования» учебника «Информатика и ИКТ» для 10 класса Угриновича Н.Д. входят следующие разделы: «Алгоритм и кодирование основных алгоритмических структур», «Введение в объектно-ориентированное визуальное программирование», «Система объектно-ориентированного программирования Microsoft Visual Studio .NET», «Система объектно-ориентированного программирования Turbo Delphi», «Переменные», «Графический интерфейс», «Пространство имен .NET», «Процедуры и функции», «Итерация и рекурсия», «Делегаты, Алгоритмы перевода чисел и их кодирование на языках объектно-ориентированного программирования», «Графика в объектно-ориентированных языках программирования», «Графика в языках программирования Visual Basic .NET, Visual C# и Visual J#», «Модульный принцип построения решений (групп) и проектов», «Чтение и запись данных в файлы», «Массивы». В указанных разделах рассматриваются следующие основные понятия: алгоритм и его свойства, алгоритмические структуры, объекты и их свойства и методы, события, проекты и приложения, процедуры и функции, алгоритмы перевода целых и дробных чисел, компьютерная и математическая системы координат, анимация, массивы [43, с. 15].

Отметим, что на углубленном уровне обучения информатике тема, связанная с изучением объектно-ориентированного программирования, также изложена в учебнике «Информатика» для 11 класса авторского коллектива Семакина И. Г. и др. и в учебнике «Информатика» для 11 класса авторов Полякова К. Ю. , Еремина Е. А.

Раздел «Объектно-ориентированное программирование» входит в главу «Методы программирования» учебника авторского коллектива Семакина И. Г, на изучение которого рекомендовано 10 часов. Данная тема включает в себя следующие разделы: «Базовые понятия ООП», «Система программирования Delphi», «Этапы программирования на Delphi», «Программирование метода статистических испытаний», «Построение графика функции»[40, с. 16].

На изучение темы «Объектно-ориентированного программирования» по учебнику авторского коллектива Полякова К. Ю. и Еремина Е. А. отведено 13 ч. Содержание данной темы включает в себя следующие разделы: «Что такое ООП?», «Объекты и классы», «Создание объектов в программе», «Создание объектов в программе», «Скрытие внутреннего устройства», «Иерархия классов», «Программы с графическим интерфейсом», «Основы программирования в RAD-средах», «Использование компонентов», «Разработка компонентов» [35, с.10].

Таким образом, проанализировав учебники школьного курса информатики, рекомендованные Министерством образования и науки Российской Федерации на базовом уровне, можно сделать следующий вывод: тема объектно-ориентированного программирования встречается только в учебнике информатики Угриновича Н. Д. для 9 класса. Основным методом изучения данной темы является проектная деятельность на практических занятиях что позволяет индивидуализировать и дифференцировать обучение; в учебниках на углубленном и профильном уровнях изучения информатики данная тема предложена только у трех авторов, где также рассматриваются основные понятия ООП, его принципы, графический интерфейс и его инструментарий и др.

## 3 Роль элективных курсов в обучении объектно-ориентированному программированию учащихся средних общеобразовательных учреждений

Прилагательное «элективный» (electus) в переводе с латинского языка означает «избранный, отобранный» [58]. Отсюда следует, что любой курс, который в учебном плане называется элективным, должен выбираться. В Концепции профильного обучения на старшей ступени общего образования (Приказ Минобразования РФ от 18 июля 2002 г. № 2783) представлено следующее определение: «Элективные курсы – обязательные для посещения курсы по выбору учащихся, входящие в состав профиля обучения на старшей ступени школы» [26].

Элективные курсы связаны с удовлетворением индивидуальных образовательных интересов, потребностей и склонностей каждого учащегося, направленных на формирование определенных компетенций. В отличие от факультативов и кружков, элективные курсы обязательны для посещения, именно они позволяют школьникам развить интерес к тому или иному предмету и четко определиться со своим дальнейшим выбором профессии. «Элективные курсы принципиально отличаются от давно существующих у нас факультативов. Факультативный курс – это не обязательный, а только возможный для изучения. Идеология «изучаю – не изучаю» не предполагает, что выбор становится обязательным элементом общего образования. А введение элективных курсов направлено на решение именно этой задачи» [38, с. 29]. Элективные курсы реализуются за счет школьного компонента учебного плана, характеризуются минимальной численностью учебной группы (1– 15 человек). Каждый учащийся в течение двух лет должен выбрать и изучить 5–6 элективов, при этом количество предлагаемых курсов должно быть значительно выше. Примерное соотношение объемов базовых общеобразовательных, профильных общеобразовательных предметов и элективных курсов определяется пропорцией 50: 30: 20. Элективные курсы выполняют следующие функции:

– изучение ключевых проблем современности;

– ознакомление с особенностями будущей профессиональной деятельности, «профессиональная проба»;

– ­­ориентация на совершенствование навыков познавательной, организационной деятельности;

– дополнение и углубление базового предметного образования;

– компенсация недостатков обучения по профильным предметам [17, с.81].

Цель элективных курсов – ориентация учащихся на индивидуализацию обучения и социализацию, на подготовку к осознанному и ответственному выбору сферы будущей профессиональной деятельности. Перед элективными курсами ставится ряд важных задач: 1) расширить знания по изучаемым предметам; 2) обеспечить более высокий уровень сформированности компетенций; 3) способствовать активному самоопределению, в том числе и профессиональному; 4) формировать и развивать познавательный интерес к учебным предметам. Исходя из этого, тематика и содержание элективных курсов должны отвечать следующим требованиям:

– иметь социальную и личностную значимость, актуальность, как с точки зрения подготовки профессиональных кадров, так и личностного развития учащихся;

– способствовать социализации и адаптации, представлять возможность для выбора индивидуальной образовательной траектории, осознанного профессионального самоопределения;

– поддерживать изучение базовых и профильных общеобразовательных предметов, а также обеспечивать условия для внутрипрофильной социализации обучения;

– обладать значительным развивающим потенциалом;

– вносить вклад в формирование целостной картины мира;

– способствовать развитию общеучебных, интеллектуальных и профессиональных умений и навыков, ключевых компетенции учащихся [19, с. 38].

Методы и формы обучения определяются уровнем развития и саморазвития учащихся, уровнем их индивидуальных способностей и требованиями профилизации обучения. Ведущие методы, реализуемые в рамках элективных курсов: проблемно-поисковые и исследовательские, стимулирующие познавательную активность учащихся.

Виды элективных курсов определяются и задачами ступени школы, и видом учебного заведения, и местными условиями, и возможностями класса, группы, учителя, и др. Обсуждая вопрос о месте элективных курсов в старшей профильной школе, выделяют виды элективных курсов, содержание и способы работы в рамках элективных курсов, образовательные задачи элективных курсов.

Отметим, что в соответствии с одобренной Министерством образования и науки РФ «Концепцией профильного обучения на старшей ступени общего образования» дифференциация содержания обучения в старших классах осуществляется на основе различных сочетаний курсов трех типов: базовых, профильных, элективных. Каждый из курсов этих трех типов вносит свой вклад в решение задач профильного обучения. Однако можно выделить круг задач, приоритетных для курсов каждого типа [26].

Базовые общеобразовательные курсы отражают обязательную для всех школьников инвариантную часть образования и направлены на завершение общеобразовательной подготовки учащихся. Профильные курсы обеспечивают углубленное изучение отдельных предметов и ориентированы в первую очередь на подготовку выпускников школы к последующему профессиональному образованию.

Роль элективных курсов в системе профильного обучения определяет широкий спектр их функций и задач. Очевидно, что общие функции и задачи элективных курсов в системе профильного обучения будут во многом по-разному проявляться в курсах этого типа по различным учебным предметам.

В частности, по информатике, одной из тем, выносимых на изучение в рамках элективных курсов, является объектно-ориентированное программирование, поскольку в настоящее время ООП является одной из наиболее актуальных и интересующих учащихся тем: данная парадигма программирования широко используется для создания Windows-приложений, удачно сочетает в себе богатейшие возможности [21, с. 45]. Изучение объектно-ориентированного программирования дает учащимся возможность выйти за рамки школьных задач и самим создавать достаточно серьезные проекты, используемые для программной поддержки уроков, для обеспечения запросов учебно-воспитательного процесса и др.

В условиях существующей традиционной классно-урочной системы образования с ее жесткой структурой реализация возможностей данной предметной среды весьма затруднительна. Необходимо отметить, что на изучение объектно-ориентированного программирования выделяется недостаточное количество часов, а во многих учебниках отсутствует данный раздел. Решением данного недостатка являются элективные курсы по объектно-ориентированному программированию.

Данные элективные курсы помогут сформировать у учащихся представления о современных языках программирования высокого уровня, овладеть технологией решения задач с применением компьютера, развивать операционное (алгоритмическое) и логическое мышление учащихся, а также сформировать умения у учащихся самостоятельно получать информацию, пополнять свои знания, приобретение учащимися опыта творческой деятельности, повышение учебной мотивации и др.

# 2 Методические аспекты обучения учащихся средних общеобразовательных учреждений объектно-ориентированному программированию

## 2. 1 Особенности обучения объектно-ориентированному программированию учащихся средних общеобразовательных учреждений

Основы объектно-ориентированного программирования рассматриваются при изучении информатики как в основной, так и в старшей школе. Объем и содержание данного раздела обусловлены уровнем общего образования (базовый или профильный/углубленный) и спецификой учебников разных авторских коллективов. Так, в учебнике «Информатика и ИКТ» автора Угриновича Н. Д. изучение основ ООП реализовано в 9 классе на базовом уровне и на профильном уровне в 10 классе. На углубленном уровне тема «Объектно-ориентированного программирования» изложена в 11 классе в учебнике «Информатика» авторского коллектива Семакина И. Г. и др. и в учебнике «Информатика» авторов Полякова К. Ю. , Еремина Е. А.

Тема «Основы алгоритмизации и объектно-ориентированного программирования» на базовом уровне по учебнику «Информатика и ИКТ» автора Угриновича Н. Д. изучается в первом полугодии 9 класса [37]. Освоение данной темы следует за главой «Кодирование и обработка числовой информации». В данной главе изучение основ ООП рассматривается на примере трех языков, реализующих объектно-ориентированную парадигму программирования: OpenOffice.org Basic в операционных системах Windows и Linux, объектно-ориентированный язык Visual Basic в операционной системе Windows и объектно-ориентированный язык Gambas в операционной системе Linux. Использование языка OpenOffice.org Basic согласуется с ГИА и ЕГЭ, а объектно-ориентированные языки Visual Basic и Gambas используют современную технологию программирования и также согласуется с ГИА и ЕГЭ [42, с. 52].

На рассмотрение указанной темы отводится 20 учебных часов, в рамках которых изучается следующее содержание (таблица 2.1).

Таблица 2.1

**Содержание главы «Основы алгоритмизации и   
объектно-ориентированного программирования»**

|  |  |  |  |
| --- | --- | --- | --- |
| **Тема урока** | **Основные понятия** | **Планируемые результаты** | **Практические задания** |
| § 4.1  Алгоритм. Свойства алгоритма и его исполнители. | Алгоритм. Свойства алгоритма. Возможность автоматизации деятельности человека.  Исполнители алгоритмов (назначение, среда, режим работы, система команд). Компьютер как формальный исполнитель алгоритмов (программ). | Знать/понимать  Понятие алгоритма и его свойства. Понятие, свойства и назначение исполнителя. Области использования. Формализация действия исполнителя.  Уметь/применять  Приводить примеры из жизни. Описывать режим работы и систему команд исполнителя. |  |
| § 4.1  Блок-схемы алгоритмов. | Способы записи алгоритмов; блок-схемы. | Знать/понимать  Способы записи алгоритмов. Блок-схемы.  Уметь/применять  Строить простые структуры блок-схемы алгоритмов. |  |
| § 4.1  Выполнение алгоритмов компьютером. | Машинный язык. Ассемблер. Языки программирования, их классификация. | Знать/понимать  Классификация языков программирования. Назначение и области применения.  Уметь/применять  Находить различие между языками. Приводить примеры языков программирования. |  |
| § 4.2  Линейный алгоритм. | Линейный алгоритм. Блок-схема линейного алгоритма. | Знать/понимать  Как выполняются команды в линейном алгоритме.  Уметь/применять  Начертить блок-схему линейного алгоритма. |  |
| § 4.2  Алгоритмическая структура «ветвление» | Алгоритмы ветвления. Способ реализации разветвляющегося алгоритма.. | Знать/понимать  понятие разветвляющегося алгоритма..  Уметь/применять  Использовать сокращенную форму алгоритмического ветвления и сокращенную форму алгоритмической структуры «выбор». Начертить блок-схему ветвления и блок-схему алгоритмической структуры «ветвление». |  |
| § 4.3  Алгоритмическая структура «выбор». | Алгоритмическая структура «выбор»и способ ее реализации на языке программирования. | Знать/понимать  .Алгоритмическую структуру «выбор».  Уметь/применять  Начертить блок-схему ветвления и блок-схему алгоритмической структуры «выбор». |  |
| § 4.2  Алгоритмическая структура «цикл». | Алгоритмическая структура «цикл» и способ ее реализации на языке программирования. Виды: «цикл со счетчиком» и «цикл с условием». | Знать/понимать  Алгоритмическую структуру «цикл». Виды: «цикл со счетчиком» и «цикл с условием».  Уметь/применять  Начертить блок-схемы алгоритмической структуры «цикл»: «цикл со счетчиком», «цикл с условием». | Практическая работа №20«Знакомство с системами объектно-ориентированного и алгоритмического программирования» |
| § 4.3  Переменные: тип, имя, значение. | Переменные: типы, имя, значение. Объявление переменным значений. Присваивание переменным значений. Значение переменных в оперативной памяти. | Знать/понимать  В чем разница между типом, именем и значением переменной. Какие основные типы переменных используются в языках программирования. Почему рекомендуется объявлять переменные перед их использованием в программе.  Уметь/применять  Определять количество ячеек в оперативной памяти. | Практическая работа №21«Проект «Переменные»» |
| § 4.4  Арифметические, строковые и логические выражения. | Арифметические, строковые и логические выражения. Операция конкатенации. | Знать/понимать  Какие элементы входят в состав арифметических, логических и строковых выражений.  Уметь/применять  Составлять программу для линейного алгоритма в среде программирования. Записывфать операторы согласно правилам записи. | Практическая работа №22«Проект «Калькулятор»» |
| § 4.5  Функции в языках объектно-ориентированного и процедурного программирования. | Понятие функции. Математические, строковые и функции ввода/вывода данных. | Знать/понимать  Понятие функции. Математические, строковые и функции ввода/вывода данных. Типы данных аргументов, возвращаемых функциями.  Уметь/применять  Составлять программы в среде программирования с использованием изученных функций. | Практическая работа №23«Проект «Строковый калькулятор»» |
| § 4.5  Функции даты и времени. | Функции даты и времени. | Знать/понимать  Функции даты и времени. Типы данных аргументов, возвращаемых функциями. Уметь/применять  Составлять программы в среде программирования с использованием изученных функций. | Практическая работа №24«Проект «Даты и время»» |
| § 4.6  Основы объектно-ориентированного визуального программирования. | Системы программирования и проектирования. Графический интерфейс проекта. Элементы управления. Объекты и их свойства, методы обработки. | Знать/понимать  Как изменить выбранный объект. Какие объекты могут использоваться при конструировании графического интерфейса.  Уметь/применять  Создать проект определения больших из числовых кодов. | Практическая работа №25«Проект «Сравнение кодов символов»» |
| Способы применения оператора выбора. | Способы применения оператора выбора в программной среде. Примеры. | Знать/понимать  На какие события реагирует кнопка. Способы применения оператора выбора.  Уметь/применять  Создать проект выставления отметок . | Практическая работа №26«Проект «Отметка»» |
| Способы применения оператора цикла со счетчиком. | Способы применения оператора цикла со счетчиком в программной среде. Примеры. | Знать/понимать  Способы применения оператора цикла со счетчиком  Уметь/применять  Создать проект вывода в поле списка числовые коды символов. | Практическая работа №27«Проект «Коды символов»» |
| Способы применения оператора цикла с предусловием. | Способы применения оператора цикла с предусловием в программной среде. Примеры. | Знать/понимать  Способы применения оператора цикла с предусловием Уметь/применять  Создать проект введенного слова в слово-перевёртыш. | Практическая работа *№28* «Проект «Слово-перевертыш»» |
| § 4.7  Графические возможности объектно-ориентированного языка программирования Visial Basic. | Область рисования. Перо. Кисть. Графические методы. Цвет. Рисование текста. | Знать/понимать  Методы рисования графических фигур и их аргументы. Уметь/применять  Составить программу «Графический редактор». | Практическая работа №29 «Проект «Графический редактор»» |
| § 4.7  Системы координат в компьютерной системе. | Системы координат в компьютерной системе. | Знать/понимать  Каким образом можно изменить систему координат формы или графического поля. Уметь/применять  Создать проект рисования осей и печать шкалы в компьютерной системе координат. | Практическая работа *№30* «Проект «Системы координат»» |
| Анимация. | Этапы создания анимации движения объекта. | Знать/понимать  Основные этапы создания анимации движения объекта. Уметь/применять  Составить программу анимации объекта. | Практическая работа №31«Проект «Анимация»» |

Приведем в качестве примера разработку урока «Переменные: тип, имя, значение».

**Технологическая карта урока по информатике «Переменные: тип, имя, значение»**

|  |  |
| --- | --- |
| **Автор** | Лысова Татьяна Анатольевна |
| **Предмет** | Информатика |
| **Класс** | 9 класс |
| **Учебно-методический комплекс** | Угринович Н. Д. |
| **Тема** | Переменные: тип, имя, значение. |
| **Место урока в изучаемой теме, разделе, курсе:** | На изучение главы №4 - Основы алгоритмизации и объектно-ориентированного программирования - выделено 20 часов. «Переменные: тип, имя, значение» – это 8 урок это темы. |
| **Тип урока** | Урок «открытия новых знаний» |
| **Цель деятельности учителя** | Познакомить учащихся с понятием переменной, основными типами переменных, способами объявления переменных. |
| **Задачи** | 1. Формирование знаний понятий переменной в Visual Basic. 2. Формирование умений использовать переменные при создании проектов в Visual Basic. 3. Мотивировать учащихся к получению образования в ИТ-сфере посредством организации продуктивной творческой деятельности и создания ситуации успеха. |
| **Образовательные ресурсы** | ЭОР(http://metodist.lbz.ru/authors/informatika/1/umk10-11p.php), авторский ЭОР(Презентация «Переменные: тип, имя, значение»), учебник (Угринович Н. Д. Информатика и ИКТ: учебник для 9 класса; 2014 г.), раздаточный материал (карточки с заданием) |

**Планируемые результаты**

|  |  |
| --- | --- |
| **Образовательные** | **Личностные:** развитие логического мышления, анализа, алгоритмического мышления; развитие творческого потенциала.  **Предметные:** готовность применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов.  **Метапредметные:** умение самостоятельно планировать пути достижения цели, в том числе альтернативные, осознанно выбирать наиболее эффективные способы решения учебных и познавательных задач; умение оценивать правильность выполнения учебной задачи, собственные возможности ее решения. |
| **Владеют базовыми понятиями по теме** | **Универсальные учебные действия (УУД*)***  **Личностные УУД:** готовность и способность обучающихся к саморазвитию и личностному самоопределению, мотивация к целенаправленной познавательной деятельности с целью приобретения профессиональных навыков в ИТ-сфере.  **Познавательные УУД:** поиск и выделение необходимой информации; построение логической цепи рассуждений; самостоятельное создание способов решения проблем творческого и поискового характера.  **Коммуникативные УУД:** умения организовывать продуктивное сотрудничество и совместную деятельность с учителем и сверстниками.  **Регулятивные УУД:** умение самостоятельно планировать пути достижения целей, соотносить свои действия с планируемыми результатами, осуществлять контроль и коррекцию своей деятельности в процессе достижения результата. |

**Организация пространства**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Формы работы** | **Технологии, методы, приемы обучения** | | **Программно-технические ресурсы:** | | | | |
| фронтальная, индивидуальная, групповая | 1. словесный (объяснение), 2. наглядный (оформление доски, демонстрация готовой программы), 3. практический (работа по разработке проекта). | | *Необходимое техническое оборудование:* проектор, компьютеры*.*  *Программное обеспечение:* MS Visual Basic, MS Power Point. | | | | |
| **Организация структуры урока** | | | | | | | |
| **1 этап. Организационный момент (2 мин)** | | | | | | | |
| **Цель деятельности** | **Деятельность учителя** | | | **Деятельность ученика** | **Наглядные средства, ЭОР** | | |
| Включение в деловой ритм. Подготовка класса к работе. | Проверка отсутствующих учеников, обеспечение эмоционального рабочего настроения учащихся. | | | Подготовка к уроку, настрой на работу. |  | | |
| **2 этап. Актуализация знаний (3 мин)** | | | | | | | |
| Выявить уровень знаний по изученному материалу. Определить типичные недостатки и ошибки. Систематизировать теоретические знания.  Активизировать знания учащихся, необходимые для изучения нового материала. Сформировать познавательные мотивы.  Организация учащихся по принятию познавательной задачи. | | Самостоятельная работа (3 минуты) с последующей проверкой. | | Выполняют работу самостоятельно по вариантам.  Сдают готовые работы.  Отвечают на вопросы и проводят самоанализ. | | | Слайд 2 (авторский ЭОР (Презентация «Переменные в Visual Basic) |
| **3 этап. Открытие новых знаний (20 мин.)** | | | | | | | |
| Сформировать конкретные представления по теме и содержанию урока, побуждать учащихся самих искать решение задач. | **Учитель объясняет новый материал, вводит новые понятия, демонстрируя примеры на слайде.**  **Тип, имя и значение переменной**  **Переменная в программе представлена именем и служит для обращения к данным определенного типа, конкретное значение которых хранится в ячейках оперативной памяти.**  **Имя переменной**. Имя каждой переменной (идентификатор) уникально и не может меняться в процессе выполнения программы. Имя переменной может состоять из различных символов (латинские и русские буквы, цифры и т.д.), но должно обязательно начинаться с буквы и не должно включать точку «.». Число символов в имени не может быть более 255.  **Соглашение об именах**  Имя каждой переменной уникально и не может меняться в процессе выполнения программы.  Для большей прозрачности кода в среде Windows используют венгерскую нотацию*,* суть которой заключается в том, что имя переменной начинается с префикса, который обозначает тип переменной.  **Арифметические выражения.**  В состав арифметических выражений могут входить переменные числового типа и числа; над переменными и числами могут производиться различные арифметические и математические операции.   1. A=23\*B-(45+F)^2/5 2. S=S+1 3. B=(A+cos (X))/N   **Строковые выражения.**  В состав строковых выражений могут входить переменные строкового типа, строки и строковые функции.   1. strA="ИНН"&vntНомер 2. strC ="град"& "ус"& strЗнак 3. strФИО=Mid(strF,2,6) & "."   **Логические выражения.**  В состав логических выражений кроме логических переменных могут входить также числа, числовые или строковые переменные или выражения, которые сравниваются между собой с использованием операций сравнения (>, <, =, >=, <= и т.д.)  Логическое выражение может принимать только два значения: **«истина»** или **«ложь»**.  Над элементами логических выражений могут производиться логические операции, которые на языке Visual Basic обозначаются следующим образом: логическое умножение – **And**, логическое сложение – **Or** и логическое отрицание **Not**. При записи сложных логических выражений часто используются скобки.  (5 > 3) **And** (2\*2 = 5) - условие ложно;  (5 > 3) **Or** (2\*2 = 5) - условие истинно.  **Переменные типа Variant** имеют большое практическое значение, однако при их применении возникают проблемы. Во-первых, при чтении кода не видно, какой внутренний тип имеет переменная в данный момент. Это может крайне затруднить обнаружение логических ошибок программирования. Во-вторых, данные этого типа из-за частых внутренних преобразований занимают больше памяти, чем аналогичные данные, объявленные с указанием явного типа.  **Объявление переменных.**  Важно, чтобы не только разработчик программы понимал, переменные какого типа используются в программе, но это мог учесть и исполнитель программы (компьютер). Второе даже более важно, так как, если компьютер не будет знать, переменная какого типа используется в программе, он будет считать е переменной универсального типа Variant и отведет для ее хранения 16 или более ячеек (байт). Это будет приводить к неэффективному использованию памяти и замедлению работы программы.  Для объявления типа переменной можно воспользоваться оператором определения переменной. | | | Учащиеся конспектируют в тетрадях основные понятия и примеры использования переменных Visual Basic. | Слайд 3, 4, 5, 6, 7, 8, 9 (авторский ЭОР(Презентация «Переменные: тип, имя, значение) | | |
| **4 этап. Проверка усвоения новых знаний (15 мин)** | | | | | | | |
| Провести первичное обобщение. Организовать деятельность по применению новых знаний*.* | | Выполнение практической работы на компьютере.  **Практическое задание «Проект «Переменные»**  Создать проект, в котором объявить переменные различных типов, присвоить им значение и вывести значения в поле списка, размещенное на форме. Изменить свойства элементов управления.  Этапы создания проекта:  1. Создание графического интерфейса  2. Создание событийной процедуры  3.Компиляция и запуск проекта на выполнение  4. Анализ и оптимизация программы.  5. Совершенствование графического интерфейса. | | Выполняют практическое задание. Сохраняют проекты в папке «Переменные». | |  | |
| **5 этап. Итоги урока. Рефлексия (3мин.)** | | | | | | | |
| Анализ успешности усвоения материала. | | Предлагает учащимся определить, что нового они узнали на уроке, где это можно применять. | | Учащиеся обобщают знания, полученные на уроке. | |  | |
| **6 этап. Домашнее задание. (2мин.)** | | | | | | | |
| Дать информацию и инструктаж по д/з. | | Задает домашнее задание с комментариями по его выполнению | | Записывают домашнее задание | |  | |

В качестве задания учащимся на данном уроке было предложено разработать проект «Переменные». Подчеркнем, что реализация технологии проектной деятельности продиктовано не только особенностью данного предмета, но и особенностями данного раздела, поскольку выполнение заданий учащимися на компьютере (в частности разработка интерфейса программного приложения) предполагает не только учебную, но и творческую составляющую, самостоятельную деятельность школьников. Однако следует отметить, что в учебнике приведено лишь одно задание для всех учащихся с указанием строгой последовательности выполнения представленного алгоритма. В результате у всех школьников в качестве итогового проекта будет совершенно одинаковый файл, что не в полной мере согласуется с технологией организации проектной деятельности. Еще раз отметим, что именно проект указан в качестве задания.

В связи с вышеизложенным, мы предлагаем несколько изменить задание, внеся коррективы не только в его содержание и сделав его разноуровневым, но и дополнив его некоторыми составляющими, где оформление, способ реализации зависят от ученика и его возможностей.

Также отметим, что задание, предложенное в учебнике для выполнения, предусматривает работу только с числовыми переменными разных типов (целыми и вещественными). Однако на уроке рассматриваются также переменные других типов (логические и строковые (символьные)). В связи с этим, целесообразно изменить содержание задания, добавив в него действия по обработке логических и строковых данных. Итоговые задания по разработке проектов для учащихся с разным уровнем подготовки приведены ниже.

I. Проект «Переменные» (задание для учащихся с низким уровнем подготовки)

1. Создать новый проект.
2. Для создания графического интерфейса разместить на форме (form1) три *командные кнопки* (Button1, Button 2, Button 3). Изменить *основные свойства объектов* (Рис.2.1).
3. Перейти в окно Программного кода и объявить переменные для их использования в программе:

Dim A, B As Integer, C As Byte, D As Single, E As Double, F,G As String, X,Y,Z As Boolean

1. Произвести двойной щелчок по кнопке c Button 1, в окне Программный код появится заготовка событийной процедуры:

Sub Button 1\_Click()

End Sub

Создать событийную процедуру, реализующую алгоритм деления двух чисел. Вывод результатов осуществить на метке Label, добавив три указанных элемента управления на форму (Label1, Label2, Label3).

1. Ввести событийную процедуру:

A = 2

B = 3

C = A / B

D = A / B

E = A / B

Label1.Text = C

Label2.Text = D

Label3.Text = E

1. Создать заготовку *событийной процедуры*, в которой в качестве объекта будет использоваться *кнопка* Button 2 (см. п.4).
2. В окне «Программный код» ввести вторую событийную процедуру, реализующую конкатенацию двух строковых выражений и строковой переменной. Вывод результатов осуществить на метке Label, добавив указанный элемент(Label4) на форму:

F = “форма”

G = “ин” + F + “тика”

Label4.Text = G

1. Создать заготовку *событийной процедуры*, в которой в качестве объекта будет использоваться *кнопка* Button 3 (см. п.4).
2. В окне *Программный код* ввести третью событийную процедуру, реализующую логические операции с логическими переменными. Вывод результатов осуществить на метке Label, добавив указанный элемент(Label5) на форму:

F = “форма”  
X = 5 > 3  
Y = 2 \* 2 = 5  
Z = X **And** Y  
Label5.Text = Z

1. Запустить проект на выполнение и щелкнуть по всем трем кнопкам.

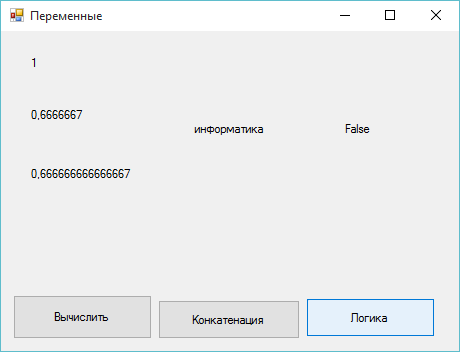


Рис. 2.1 Интерфейс проекта «Переменные»

На форму будут выведены значения переменных.

10. Сохранить проект в личной папке.

II. Проект «Переменные» (задание для учащихся со средним уровнем подготовки)

Кроме указанных выше заданий п. 1-9 дополнительно необходимо видоизменить интерфейс приложения путем изменения свойств элементов управления (рис. 2):

10. Изменить цвет фона формы и цвет фона кнопок (свойство BackColor), установив значение свойств по своему усмотрению.

11. Изменить вид выводимого текста: шрифт, размер шрифта, начертание (свойство Font), установив значение свойств по своему усмотрению.

12. Изменить цвет текста у элементов управления (свойство FontColor), установив значение свойств по своему усмотрению.

13. Запустить проект на выполнение и щелкнуть по всем трем кнопкам.

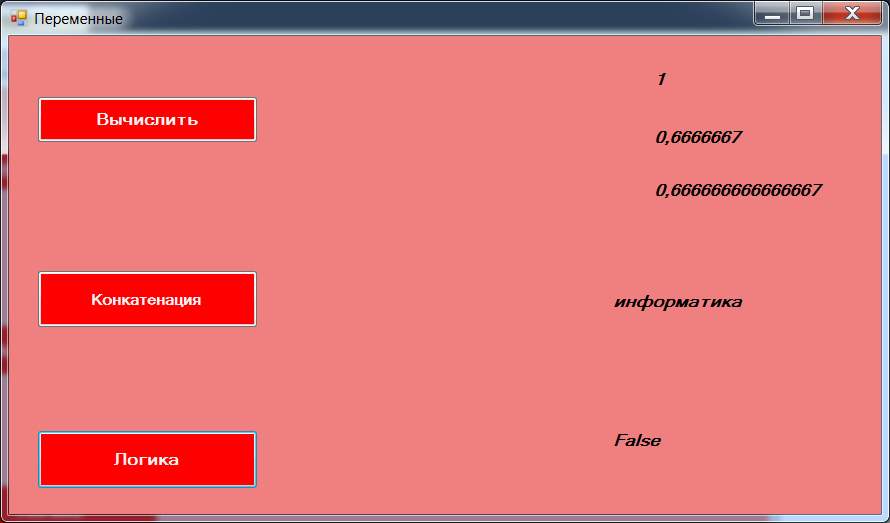


Рис. 2. 2 Интерфейс проекта «Переменные»

для учащихся со среднего уровнем подготовки

14. Сохранить проект в личной папке.

III. Проект «Переменные» (задание для учащихся с высоким уровнем подготовки)

Кроме указанных выше заданий п. 1-12 дополнительно необходимо дополнить интерфейс приложения иллюстрацией (рис. 4):

13. С использованием ресурсов интернет выбрать рисунок, иллюстрирующий науку информатику.

14. Сохранить выбранный рисунок в папке с проектом с именем «Мой рисунок».

15.Добавить на форму элемент управления PictureBox.

16. Изменить свойство Image элемента управления PictureBox1, выбрав в открывшемся диалоговом окне «Выбор ресурса» «Импорт локального ресурса» (рис. 3)

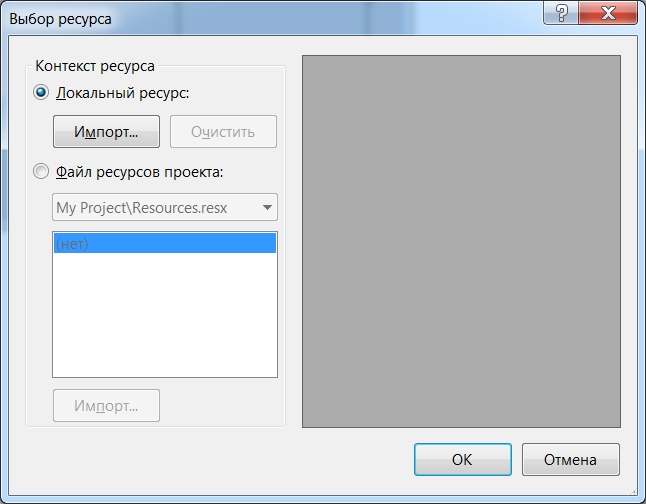


Рис. 2. 3 Диалоговое окно «Выбор ресурса»

17. В открывшейся папке с проектом выбрать файл «Мой рисунок».

18. Изменить свойство SizeMode элемента управления PictureBox1, выбрав значение StretchImage, что позволяет разместить рисунок по всей области элемента управления.

19. Запустить проект на выполнение и щелкнуть по всем трем кнопкам.

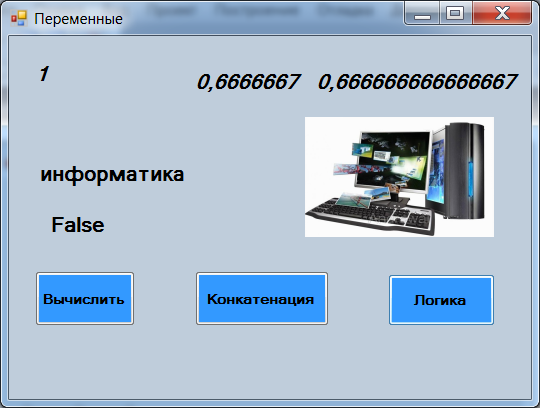


Рис. 2. 4 Интерфейс проекта «Переменные»

для учащихся с высоким уровнем подготовки

20. Сохранить проект в личной папке.

Таким образом, добавление дополнительных заданий позволяет индивидуализировать работу учащегося, приблизив ее к реальной проектной деятельности, поскольку в результате их выполнения каждый учащийся разрабатывает свой особый проект, который отличается индивидуальным интерфейсом. Ниже приведем проекты учащихся, которые были выполнены на уроке информатики в рамках апробации разработанного материала (рис. 5).

|  |  |
| --- | --- |
|  |  |
|  |  |

Рис.2. 5 Интерфейс проекта «Переменные»

разработанные учащимися «9 А»

Как показала апробация, выполнение данного задания вызвало интерес у учащихся к объектно-ориентированному программированию, позволило развить у них навыки разработки программного интерфейса, работы с элементами управления Label, Button, PictureBox, программирования свойств элементов управления и т.п. Все вышеперечисленное свидетельствует о том, что подобные задания способствуют развитию мотивации к изучению информатики и программирования, поскольку в результате получается завершенный проект, а как известно мотивация к изучению того или иного предмета возникает у ученика через успех (нравится обычно то, что получается).

В качестве самостоятельной работы учащимся было предложено по желанию усовершенствовать свой проект, сделав его межпредметным (например, разработать «Математический калькулятор», «Физический конвертер величин» для преобразования значений в систему «СИ» и др.). Пример подобного приложения приведен на рисунке 2. 6

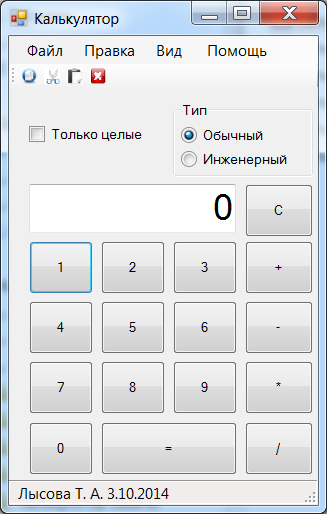


Рис. 2. 6 Приложение «Калькулятор»

Отметим, что выполнение подобного задания демонстрирует тесную связь объектно-ориентированного программирования с другими школьными дисциплинами, что способствуют развитию мотивации к изучению данного достаточно сложного раздела курса информатики и развитию творческого потенциала учащихся.

## 2 Элективный курс «Основы объектно-ориентированного программирования средствами Visual Basic»

Согласно пункту 3 Федерального государственного образовательного стандарта среднего (полного) общего образования освоение основной образовательной программы предусматривает обязательное изучение курсов по выбору (элективных курсов) [50]. Также следует отметить, что элективные курсы являются важным средством для построения индивидуальных образовательных программ, так как, в наибольшей степени близки к выбору каждым школьником элементов содержания образования в зависимости от собственных способностей, интересов, жизненных планов. С хорошо разработанной программой элективного курса каждый ученик может получить дополнительное образование с определенным желаемым уклоном в ту или иную область знаний [39, с. 18].

Мы предположили, что учащимся будет интересно дополнительно в рамках элективного курса изучить некоторые аспекты объектно-ориентированного программирования. Поэтому, было проведено анкетирование среди учащихся 10-го класса. Вопросы анкеты приведены ниже.

1. Хотели бы вы уметь разрабатывать Window-приложения (например, калькулятор, графический редактор и др.)?

А) да;

Б) нет;

В) затрудняюсь ответить.

2. Достаточно ли вам знаний и навыков для самостоятельной разработки Window-приложений?

А) да;

Б) нет;

В) думаю, что не совсем достаточно.

3. В рамках какого раздела информатики рассматривается разработка Window-приложений?

А) информационные технологии;

Б) алгоритмизации и программирования;

В) информации и информационных процессов.

4. Считаете ли вы, что изучение программирования является сложным и неинтересным?

А) да;

Б) нет;

В) затрудняюсь ответить.

5. Какой раздел программирования занимается проектированием визуальных интерфейсов?

А) объектно-ориентированное;

Б) процедурное;

В) функциональное.

6. Какие средства визуального проектирования вы знаете?

А) Paskal;

Б) Visual Basic;

В) C.

7. Каким способом можно разработать визуальный интерфейс приложения?

А) путем вызова стандартных элементов управления и программирования их свойств;

Б) путем поэтапного создания каждого элемента;

В) путем создания шаблонов элементов.

8. Можно ли осуществить графическое построение в среде программирования?

А) да;

Б) нет;

В) затрудняюсь ответить.

9. Можно ли запрограммировать в приложении организацию визуального выбора (как в онлайн-тестах) и с помощью каких элементов это реализуется?

А) флажок;

Б) переключатель;

В) нельзя реализовать.

10. Хотели бы вы уметь дополнительно более углублено изучать язык визуального программирования Visual Basic?

А) да;

Б) нет;

В) затрудняюсь ответить.

Вопросы анкеты были направлены на выявление следующих параметров: наличия мотивации у учащихся к разработке программных приложений средствами систем программирования, изучению объектно-ориентированного программирования, уровня знаний в области объектно-ориентированного программирования.

В качестве анкетируемых участвовали ученики 10-го класса (20 учени-ков). Анкета проводилась дважды: первый раз (1 этап) –до проведения некоторых занятий элективного курса, повторно (2 этап) –после проведения занятий курса.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Рис. 2. 7 Диаграмма результатов анкетирования 1этапа

Анализ результатов анкетирования показал, что у большинства учащихся (60%) отмечается устойчивая мотивация к изучению объектно-ориентированного программирования и созданию программных приложений, однако более половины из них (55%) считают его сложным для изучения. Также отмечается достаточно низкий уровень знаний школьников в области объектно-ориентированного программирования: на вопросы, касающиеся данного раздела, многие учащихся дали неправильные ответы (на 5–65%, 7–65%, 8–50% неправильных ответов).

После проведения занятий элективного курса было повторно проведено анкетирование учащихся, которые посещали занятия элективного курса. Данные анкетирования приведены на рис. 2.8

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Рис. 2. 8 Диаграмма результатов анкетирования 2этапа

Как показал анализ результатов, после проведения занятий элективного курса наблюдалось увеличение количества учащихся (80% против 60% по результатам первого анкетирования), имеющих устойчивую мотивацию к изучению объектно-ориентированного программирования и созданию программных приложений, существенно понизился процент учащихся (20% против 55% по результатам первого анкетирования), которые считают его сложным для изучения. Также повысился уровень знаний школьников в области объектно-ориентированного программирования: на вопросы, касающиеся данного раздела, большинство учащихся дали правильные ответы.

Приведем разработку соответствующего элективного курса. Составленная документация соответствует всем требованиям, предъявляемым к программам учебных курсов.

*Пояснительная записка*

Объектно-ориентированное программирование, в частности, язык Visual Basic, является в настоящее время одной из наиболее актуальных и интересующих учащихся тем. Этот современный, широко используемый для создания Windows-приложений язык удачно сочетает в себе богатейшие возможности и достаточный уровень легкости в изучении. Овладение языком такого уровня дает учащимся возможность выйти за рамки «ученических» задач и самим создавать достаточно серьезные проекты, используемые для программной поддержки уроков, для обеспечения запросов учебно-воспитательного процесса. Кроме того, язык Visual Basic является встроенным языком широко изучаемых в школьном курсе информатики приложений Microsoft Office. Поэтому знание Visual Basic позволит учащимся глубже понимать процессы, происходящие при выполнении основных операций в компьютерной среде (работа с командами меню, нажатие кнопок панелей инструментов, выбор значения из списка и другое).

*Целевая аудитория:* учащиеся 10-11 классов. Предполагается, что учащиеся, осваивающие данный курс, имеют некоторый опорный уровень знаний языка программирования Basic (назначение и типы переменных, основные операторы, базовые алгоритмические конструкции и их реализация в Basic), и практические навыки работы с Windows-приложениями.

Целью данного курса является формирование у учащихся представлений об одном из современных языков программирования высокого уровня и умения применять его для решения задач.

Задачи курса:

• практическое овладение учащимися технологией решения задач с применением компьютера;

• развитие операционного (алгоритмического) и логического мышления учащихся;

• формирование умения учащихся самостоятельно получать информацию, пополнять свои знания;

• приобретение учащимися опыта творческой деятельности;

• повышение учебной мотивации.

*Структура курса:* теоретический материал (прилагается список литературы и материал для учащихся), практикум, задания для самоконтроля, тесты, задания для итогового контроля в конце семестра (приведены в Приложении 1).

*Принципы отбора содержания и организации учебного материала:*

• принцип практикоориентированности, предполагающий отбор содержания курса, построение логики изложения материала исходя из востребовательности вновь получаемых знаний для решения поставленных перед учащимися задач.

• принцип вариативности, предполагающий построение содержания курса, изменение соотношения теоретической и практической его составляющих с учетом начального базового уровня знаний и умений учащихся.

*Метод обучения:* основным методом обучения в данном курсе является метод проектов. Проектная деятельность позволяет развить исследовательские и творческие способности учащихся, повышает мотивацию освоения материала.

*Форма обучения:* очно-заочная. Предполагается изучение материала двумя способами: индивидуальное самостоятельное изучение или в составе малой группы под руководством учителя.

Роль учителя во время занятия состоит в объяснении нового материала и постановке задачи, а затем консультировании учащихся в процессе выполнения практического задания. Преподаватель также осуществляет различные формы контроля качества освоения программы, проверяет и оценивает созданные учащимися проекты.

Учащиеся посещают занятия, изучают необходимый теоретический материал, содержащий примеры готовых проектов, и выполняют практические задания, самостоятельно и под руководством учителя. В течение учебного полугодия учащиеся должны самостоятельно выполнить итоговое практическое задание и предоставить его на проверку.

*Организация самостоятельной работы:* самостоятельная работа учащихся предполагает: самостоятельное изучение содержания отдельных тем курса, выполнение текущих и итоговых практических заданий, выполнение тестов.

*Организация консультирования:* консультирование осуществляется как в ходе занятий с педагогом во время выполнения самостоятельных практических и итоговых работ, так и путем применения коммуникационных технологий (участие в работе форума на сетевом ресурсе, электронная почта и др.).

*Формы контроля*

Преподавателю перед изучением нового материала необходимо, используя разные формы (фронтальная беседа, небольшая практическая работа) выявлять базовый уровень знаний и умений учащихся, своевременно их дополнять, при необходимости исправлять, актуализировать опорные знания и способы выполнения действий, тем самым готовить ребят к сознательному усвоению новых понятий и приемов деятельности, повышать мотивацию учения.

Текущая аттестация качества освоения программы осуществляется учителем по результатам выполнения учащимися практических заданий и тестирования. Выполнение практических работ после изучения отдельных тем программы предполагает комплексное творческое применение учащимися полученных знаний, навыков, умений, их углубление и закрепление.

Итоговый контроль качества освоения программы реализуется в конце учебного полугодия. Учащиеся защищают самостоятельно выполненные проекты. Работа оценивается, отмечаются положительные и отрицательные моменты и даются необходимые рекомендации. Курс считается освоенным, если учащийся успешно выполнил все задания.

*Планируемые результаты курса. Требования к знаниям и умениям учащихся.*

Учащиеся должны знать:

• основные понятия объектно-ориентированного визуального программирования;

• этапы разработки приложения, понятие проекта;

• основные элементы управления Visual Basic, их свойства, методы, события;

• основные операторы и запись основных алгоритмических конструкций на изучаемом языке;

• типы переменных, используемые в языке, способы их описания и особенности применения в зависимости от типа, времени жизни и зоны видимости;

• способ организации данных в виде массива переменных;

• способы создания и особенности использования массивов элементов управления;

• назначение и способы создания диалоговых окон;

• типы и назначение стандартных констант языка Visual Basic;

• графические возможности среды.

Учащиеся должны уметь:

• выполнять основные операции создания интерфейса и программного кода будущего приложения;

• выбирать элементы управления, наиболее рациональные для решения задачи;

• применять в интерфейсе проекта готовые или самостоятельно созданные графические образы;

• правильно использовать в проекте операторы и алгоритмические конструкции;

• использовать массивы переменных и элементов управления;

• создавать диалоговые окна для взаимодействия с пользователем во время работы приложения;

• проводить отладку программ;

• проводить компьютерный эксперимент разработанного проекта и анализировать полученные результаты.

*Учебно-методический комплекс*

1. Бабушкина, И.А. Практикум по объектно-ориентированному программированию / Бабушкина И.А., Окулов С.М. М.: БИНОМ. Лаборатория знаний, 2012.– 366 с.
2. Бородин, М. Н. Информатика. 10 – 11 классы. Методическое пособие для учителя. Углублённый уровень. – БИНОМ. Лаборатория знаний, 2013. – 114 с.
3. Голубев, О. А.Visual Basic. Самоучитель для начинающих / под ред. О.А. Голубев. – М. : Диалог-МИФИ, 2012. – 448 с
4. Златопольский, Д. М. Сборник задач по программированию. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2013. — 304 с.
5. Зыков, С. В. Введение в теорию программирования. Объектно-ориентированный подход / С.В. Зыков. – 2-е изд., испр.– М. : Национальный Открытый Университет «ИНТУИТ», 2016. – 189 с.
6. Интерактивный учебник по Visual Basic [Электронный ресурс]– Режим доступа: https://msdn.microsoft.com/ru-ru/library/90h82b3x(v=vs.90).aspx
7. Полежаева, О. А., Цветкова, М. С. Информатика. 10 – 11 классы. Методическое пособие для учителя. Углублённый уровень. – БИНОМ. Лаборатория знаний, 2014. - 114 с.
8. Поляков, К. Ю., Еремин, Е. А. Информатика. Углубленный уровень: учебник для 11 класса: в 2 частях, Ч. 2./ К. Ю. Поляков, Е. А. Еремин – М. БИНОМ. Лаборатория знаний, 2014. – 312с.
9. Семакин, И. Г. Информатика. Углубленный уровень : учебник для 11 класса: в 2 частях, Ч.1. / И. Г. Семакин, Е. К. Хеннер, Л. В. Шестакова. –БИНОМ. Лаборатория знаний, 2014. – 176 с.

10. Угринович, Н. Д. Информатика и ИКТ. 8–11 классы : методическое пособи. - 2 изд. БИНОМ. Лаборатория знаний , 2014. – 149 с.

11. Угринович, Н. Д. Информатика и ИКТ. Профильный уровень: учебник для 11 класса / Н. Д. Угринович. – М.: БИНОМ. Лаборатория знаний, 2013.– 308 с.

12. Угринович, Н.Д. Практикум по информатике и информационным технологиям. Учебное пособие для общеобразовательных учреждений/ Н.Д. Угринович, Л.Л. Босова, Н.И. Михайлова. – М.: Бином. Лаборатория Знаний, 2014. – 400 с.

13. Фомин, Г.В. Двуязычная компиляция из MSDN Visual Studio. Методическое пособие. – Южный федеральный университет, 2013. – 165 стр.

*Основное содержание программы*

Предлагаемая программа разбита на 5 модулей. Каждая тема сочетает лекционный материал и материалы практических заданий. Предполагается обязательная самостоятельная работа учащихся по приобретению и закреплению знаний, практических навыков и умений по каждой теме.

*1. Введение в Visual Basic*

Основные виды программирования. Объектно-ориентированное программирование. Визуальное программирование. Понятие объект, свойства объекта, методы, события.

История создания языка Visual Basic, назначение, сферы применения. Основные достоинства языка.

Среда Visual Basic: запуск, основные элементы среды (строка заголовка, главное меню, стандартная панель инструментов, панель элементов управления, окно проводника проекта, конструктор форм, окно свойств, окно просмотра объектов, окно редактора кода).

Объекты среды Visual Basic. Свойства, общие для всех объектов.

Понятие проекта, операции с проектом: создание, сохранение, добавление/удаление формы, выполнение проекта.

Этапы разработки приложения. Понятие процедуры. Общий вид событийной процедуры.

*2. Формы и основные элементы управления*

Понятие графического интерфейса приложения. Форма как основной элемент интерфейса: назначение, основные свойства, методы, события, настройка параметров формы.

Технология создания программного кода: вызов редактора кода, создание заготовки процедуры, ввод команд и способы автоматизации ввода, запуск.

Понятие элементов управления. Действия, выполняемые с объектами формы: размещение на форме, выделение, перемещение, изменение размеров, выравнивание, удаление.

Метка, текстовое поле и командная кнопка как наиболее часто используемые элементы управления: назначение, основные свойства, события, методы. Графические свойства формы и командной кнопки.

Флажки, переключатели, фреймы: назначение, основные свойства, события, методы. Объединение элементов управления в группу.

Типы и назначение списков, основные свойства, события, методы.

Ввод данных с помощью полос прокрутки.

*3.Основные элементы программирования*

Понятие переменной в Visual Basic: правила образования имен, основные типы и особенности их применения, способы объявления переменных. Динамические и статические переменные, их описание и применение. Локальные и глобальные переменные: понятие, способы описания, применение.

Числовые переменные: понятие, типы. Арифметические операции, приоритет операций. Математические функции. Запись математических выражений на языке Visual Basic.

Символьные переменные. Функции обработки строк, функции преобразования типов данных.

Константы Visual Basic. Типы встроенных констант, их назначение и доступ к ним с помощью обозревателя объектов.

Использование в проектах диалоговых окон: понятие диалогового окна, типы диалоговых окон и их назначение. Окно ввода, технология создания окна и ввода информации. Виды окон сообщений, технология их создания, применение встроенных констант для создания интерфейса окна сообщения.

Основные типы алгоритмических структур. Линейные алгоритмы.

Алгоритмы ветвления: понятие, применение. Условные выражения, простые и сложные условия. Управляющие конструкции IF … Then и IF … Then … Else. Применение оператора ELSEIF. Оператор выбора SELECT CASE.

Циклические алгоритмы. Алгоритм цикл со счетчиком: назначение, особенности применения. Понятия счетчик цикла, тело цикла. Конструкция FOR … NEXT. Условный цикл: назначение, типы. Конструкция DO WHILE … LOOP.

Массивы переменных как способ организации данных: понятие массива, объявление, способы заполнения.

*4.Графические возможности языка Visual Basic*

Назначение графики в приложениях. Способы применения графики в Visual Basic. Понятие координатной системы. Единицы измерения.

Графические методы: понятие, основные методы (построение точки, линии, прямоугольника, окружности, эллипса, дуги, сектора, вывод текста, определение цвета точки, очистка), для каких объектов применяются. Способы задания цвета. Свойства объектов, влияющие на работу графических методов (цвет границ, толщина линий, цвет и стиль заполнения контура, размер шрифта, координаты текущей точки).

Элементы управления, применяемые для работы с графикой. Графические элементы Line и Shape. Размещение на форме графических изображений: элементы Image и PictureBox.

*5.Создание проектов в среде Visual Basic*

Обобщение изученного материала: создание проектов, требующих комплексного применения знаний, навыков и умений.

*Основные понятия:* объект, свойство, событие, метод, объектно-ориентированное программирование, визуальное программирование, проект, графический интерфейс, форма, элементы управления, процедура, оператор, программный код, константа, переменная, массив, диалоговое окно, основные алгоритмические структуры, графические методы.

*Учебно-тематический план*

Учебно-тематическое планирование предполагает, что при распределении материала необходимо выделять часы на изучение теоретического материала; выполнение практических заданий; консультирование: непосредственное общение учащихся с педагогом; тематический контроль.

*Учебно-тематический план*

| Темы | Содержание | Количество часов | | |
| --- | --- | --- | --- | --- |
| Изучение теоретического материала | Выполнение практических заданий | Тематический контроль |
| I. Введение в Visual Basic | Введение: история создания и назначение среды программирования Visual Basic. Основы объектно-ориентированного визуального программирования. Объекты: свойства, методы, события. Знакомство со средой Visual Basic. Понятие проекта. Этапы разработки приложения. | 3 | 1 | 1 |
| II. Формы и основные элементы управления. | Форма: основные свойства, методы, события. Инструмент вывода информации «Поле метки». Инструмент управления «Командная кнопка». Графические свойства формы и командной кнопки. Функция LoadPicture. Инструмент ввода/вывода информации «Текстовое поле». Разработка простейших проектов в среде программирования Visual Basic. Флажки, переключатели, фреймы. Объединение элементов управления в группу. Списки как элементы управления. Полосы прокрутки. | 7 | 7 | 1 |
| Ш. Основные элементы программирования | Понятие переменной в Visual Basic: имя, основные типы, способы объявления. Числовые переменные. Арифметические операции. Математические функции. Понятие переменной: время жизни переменных. Зона видимости переменных. Символьные переменные. Строковые функции. Функции преобразования типов данных. Встроенные константы Visual Basic. Использование в проектах диалоговых окон. Основные типы алгоритмических структур: линейный алгоритм, алгоритмы ветвления. Оператор выбора. Циклические алгоритмы: цикл со счетчиком. Циклические алгоритмы: условный цикл. | 12 | 12 | 1 |
|  | Массивы в Visual Basic: назначение, описание, способы заполнения одномерных массивов. Массивы элементов управления. |  |  |  |
| IV. Графические возможности языка Visual Basic | Графические методы. Графические элементы Line и Shape. Размещение на форме графических изображений: элементы Image и PictureBox. | 6 | 6 | 1 |
| V. Создание проектов в среде Visual Basic | Обобщение изученного материала. Создание проектов. | 2 | 8 | - |
| Итого: | 68 часов | 30 | 34 | 4 |

По результатам апробации мы сделали следующие выводы. Первоначальный уровень знаний учащихся в области объектно-ориентированного программирования был достаточно низок, у большинства анкетируемых учащихся наблюдалась устойчивая мотивация к изучению объектно-ориентированного программирования и созданию программных приложений, однако большинство из них считали его сложным для изучения. На вопросы, касающиеся данного раздела, при проведении анкетирования на первом этапе, многие учащихся отвечали неверно.

Проведение элективного курса по объектно-ориентированному программированию способствовало созданию условий для реализации сотрудничества между учителем и учеником и совместной творческой деятельности. Также положительными моментами данного элективного курса являются развитие способностей активной практической деятельности, освоение учащимися знаний (основные понятия объектно-ориентированного визуального программирования, этапы разработки приложения, понятие проекта, графические возможности среды, основные элементы управления Visual Basic, их свойства, методы, события и т.д.), что подтверждает достижение планируемых результатов курса.

После проведенных занятий электива отмечается повышение внимания к данной визуальной среде программирования. Таким образом, данный курс вызвал интерес у учащихся к основам объектно-ориентированного программирования, позволил развить у них навыки составления программных приложений и решения задач других предметных областей средствами систем программирования и формирования общей алгоритмической культуры.

# Заключение

Изучение объектно-ориентированного программирования является сегодня одним из приоритетных направлений при изучении программирования в школьном курсе информатики. От того, как будет реализовано изучение этой парадигмы программирования в школе и от того, насколько правильно будут сформированы у школьников основные понятии ООП во многом зависит не только уровень развития алгоритмического и логического мышления и мотивации к дальнейшему изучению программирования, но и выбор будущей сферы деятельности [54, с. 45].

В рамках данной выпускной квалификационной работы были рассмотрены современные подходы к изучению объектно-ориентированного программирования на уроках информатики, а также определены роль и место объектно-ориентированного программирования в курсе информатики и ИКТ. Анализ школьных учебников по информатике, направленный на выявление методических аспектов обучения объектно-ориентированному программированию на уроках информатики, позволил сделать вывод о том, что объектно-ориентированное программирование в школьном курсе не нашло должного отражения. Данная тема раскрыта только в одном учебнике по информатике базового уровня. Однако объектно-ориентированное программирование рассматривается на профильных и углубленных уровнях изучения информатики у многих ведущих авторов, таких как Угринович Н. Д., Семакин И. Г., Поляков К. Ю. и Еремин Е. А. Во всех указанных учебниках рассмотрение основ объектно-ориентированного программирования реализуется на базе среды разработки приложений Visual Basic, именно это и послужило основанием выбора данной программной среды в нашей работе.

Для достижения цели работы были решены следующие задачи:

1. Рассмотрены основы обучения учащихся средних общеобразовательных учреждений объектно-ориентированному программированию. Так, объем и содержание данного раздела обусловлены уровнем общего образования (базовый или профильный/углубленный) и спецификой учебников разных авторских коллективов.
2. Охарактеризованы особенности изучения объектно-ориентированного программирования в курсе информатики общеобразовательной школы. Сделан вывод, что одной из основных технологий обучения в рамках данной темы является проектная деятельность, что позволяет индивидуализировать и дифференцировать обучение.
3. Выявлена роль элективных курсов в обучении объектно-ориентированному программированию учащихся средних общеобразовательных учреждений. Показано, что в рамках часов, отведенных на изучение данной темы, невозможно даже овладение основами ООП, а углубленное изучение возможно в рамках элективного курса. Исходя из этого, элективные курсы могут оказать необходимую помощью в обучении объектно-ориентированному программированию в условиях того количества часов, которое определено в рамках школьного курса информатики.
4. Изучены характеристики интегрированной среды разработки приложений Visual Basic, ее интерфейс, описан функционал основных элементов управления.
5. Рассмотрены особенности обучения объектно-ориентированному программированию учащихся средних общеобразовательных учреждений. Выделены основные понятия, изучаемые в рамках данного раздела, а также планируемые результаты обучения для каждого урока.
6. Приведены методические разработки к проведению урока по теме «Переменные: тип, имя, значение». Разработана система заданий к данному уроку в целях индивидуализации и дифференциации обучения с учетом уровня подготовки учащихся по информатике. Как показала апробация, выполнение данных заданий вызвало интерес у учащихся к объектно-ориентированному программированию, позволило развить у них навыки разработки программного интерфейса, работы с элементами управления Label, Button, PictureBox, программирования свойств элементов управления и т.п.
7. Разработан элективный курс «Основы объектно-ориентированного программирования средствами Visual Basic». В работе приводится программа курса с указанием цели курса, его содержания, перечня используемого учебно-методического комплекса и др. В приложении приведены разработанные нами теоретический материал (лекции), задачи и тестовые задания для самоконтроля, проекты, методические рекомендации для учителей.

Элективный курс был апробирован, после проведенных занятий электива отмечалось повышение внимания к данной визуальной среде программирования, данный курс позволил развить у учащихся навыки составления программных приложений и решения задач других предметных областей средствами систем программирования и формирования общей алгоритмической культуры.

# Список использованных источников

1. Андреева, Е.В. Методика обучения основам программирования на уроках информатики // Материалы курса «Методика обучения основам программирования на уроках информатики» : Лекции 1-8. – М. : Педагогический университет «Первое сентября», 2006. – 60 с.
2. Бабина, Н. Ф. Выполнение проектов : учеб.-метод. пособие [Электронный ресурс] / Н. Ф. Бабина – М.-Берлин : Директ-Медиа, 2015. – 77с. – Режим доступа: http://biblioclub.ru/index.php?page=book\_view&book\_id=276774.
3. Бабушкина, И.А. Практикум по объектно-ориентированному программированию / Бабушкина И.А., Окулов С.М. М.: БИНОМ. Лаборатория знаний, 2012.– 366 с.
4. Бережнова, Е. В. Основы учебно-исследовательской деятельности студентов : учебник. – М. : Просвещение, 2006. –94 с.
5. Бешенков, С. А. Курс информатики в контексте новых образовательных результатов / С. А. Бешенков // Информатика и образование. – 2013. – №9. – С. 17-22.
6. Бородин, М. Н. Информатика. 10 – 11 классы. Методическое пособие для учителя. Углублённый уровень. – БИНОМ. Лаборатория знаний, 2013. – 114 с.
7. Бурденко, Е. Э. Visual Basic в задачах. [Электронный ресурс]. – Режим доступа: http://videouroki.net/filecom.php?fileid=98683584
8. Голубев, О. А.Visual Basic. Самоучитель для начинающих / под ред. О.А. Голубев. – М. : Диалог-МИФИ, 2012. – 448 с
9. Городской методический центр [Электронный ресурс]. – Режим доступа: http://mosmetod.ru.
10. Демонстрационный вариант контрольно измерительных материалов единого государственного экзамена 2016 года по информатике и ИКТ [Электронный ресурс]. – Режим доступа : http://www.fipi.ru/ege-i-gve-11/demoveRsii-sPeciFikacii-kodiFikatoRy
11. Джуринский, А. Н. Развитие образования в современном мире : учебное пособие / А. Н. Джуринский. – М. : Дрофа, 2008. – 200 с.
12. ЕГЭ портал [Электронный ресурс]. – Режим доступа: http://4ege.ru/analitika/52278-osnovnye-rezultaty-ege-proshlyh-let.html
13. Ершов, А. П. Введение в теоретическое программирование: Беседы о методе / А. П. Ершов. – М. : Книга по Требованию, 2012. – 288 с.
14. Закон Об образовании[приянят Гос. Думой 29 дек. 2012 г.]. [Электронный ресурс] – Режим доступа: минообрнауки.рф/документы/2974.
15. Захарова, Т.Б. Профильная дифференциация обучения информатике в старшей ступени школы / Т.Б. Захарова. – М. : МЦНТИ, 2005. – 213 с.
16. Златопольский, Д. М. Сборник задач по программированию. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2013. — 304 с.
17. Зубрилин, А. А. Виды и средства контроля знаний, умений и навыков обучаемых, применяемые на элективных курсах по информатике / А. А. Зубрилин, Е. М. Юртанова // Информатика и образование. – 2006. – № 10. – С. 79–83.
18. Зубрилин, А. А. О некоторых проблемах внедрения элективных курсов [Электронный ресурс]. – Режим доступа : http://www.portalus.ru.
19. Зубрилин, А. А. Особенности организации деятельности обучаемых на элективных курсах по информатике / А. А. Зубрилин // Информатика и образование. – 2010. – № 5. – С. 37–43.
20. Зубрилин, А. А. Оценка готовности обучаемых к участию в элективных курсах по информатике / А. А. Зубрилин // Информатика и образование. – 2009. – № 11. – С. 66–72.
21. Зыков, С. В. Введение в теорию программирования. Объектно-ориентированный подход / С.В. Зыков. – 2-е изд., испр.– М. : Национальный Открытый Университет «ИНТУИТ», 2016. – 189 с.
22. Иванова, Л.В. Методика обучения программированию будущих учителей информатики / Л.В. Иванова // Ученые записки ЕГПУ. Том 18. Серия «Физико-математические науки». – Издательство ЕГПУ, 2012 г. – 138 с.
23. Интерактивный учебник по Visual Basic [Электронный ресурс]– Режим доступа: https://msdn.microsoft.com/ru-ru/library/90h82b3x(v=vs.90).aspx
24. Каспржак, А. Г. Проблема выбора : элективные курсы в школе / А. Г.Каспржак. – М. : Новая школа, 2011. – 160 с.
25. Каспржак, А. Г. Элективные курсы – ответ на запросы ученика и учителя, семьи и государства / А. Г. Каспржак // Директор школы, 2013. – № 1. – С. 3–9.
26. Приказ Минобразования РФ от 18 июля 2002 г. N 2783 "Об утверждении Концепции профильного обучения на старшей ступени общего образования" [Электронный ресурс]. – Режим доступа: http://base.garant.ru/184895/#ixzz49k9y3mNo.
27. Кузнецов, A. A. Изучение ИКТ в курсе информатики: методические проблемы и пути их решения / А. А. Кузнецов, А. С. Захаров, Т. Н. Суворова // Информатика и образование. – 2013. – №12. – С.3–9.
28. Лапчик, М.П. Теория и методика обучения информатике: учебник / М.П. Лапчик, И.Г. Семакин, Е.К. Хеннер, М.И. Рагулина и др.; под ред. М.П. Лапчика. – М. : Издательский центр «Академия», 2012. – 592 с.
29. Ломачкина, Н. В. Обучение программированию на элективных курсах по информатике / Н. В. Ломачкина // Проблемы и перспективы развития информационных и коммуникационных технологий в образовании и науке: межвузовский сборник научно-методических трудов студентов, аспирантов и молодых ученых / Редкол.: Н. В. Вознесенская, Е. В. Черемухина, МордГПИ. – Саранск, 2013. – . 123-128.
30. Методическая служба. Лаборатория знаний [Электронный ресурс]. - Режим доступа : http://metodist.lbz.ru. 18. Нигма – Википедия [Электронный ресурс]. - Режим доступа : http://ru.wikipedia.org.
31. Основные подходы к обучению студентов объектно–ориентированному программированию и проектированию [Электронный ресурс]. – Режим доступа: http://www.rae.ru/fs/?section=content&op= show\_article&article\_id=7780826
32. Официальный информационный портал единого государственного экзамена [Электронный ресурс]. – Режим доступа: http://ege.edu.ru/ru/main/satistics-ege/
33. Официальный информационный портал единого государственного экзамена [Электронный ресурс]. – Режим доступа: http://ege.edu.ru/ru/main/satistics-ege/
34. Полежаева, О. А., Цветкова, М. С. Информатика. 10 – 11 классы. Методическое пособие для учителя. Углублённый уровень. – БИНОМ. Лаборатория знаний, 2014. - 114 с.
35. Поляков, К. Ю., Еремин, Е. А. Информатика. Углубленный уровень: учебник для 11 класса: в 2 частях, Ч. 2./ К. Ю. Поляков, Е. А. Еремин – М. БИНОМ. Лаборатория знаний, 2014. – 312с.
36. Поляков, С. Д. В поисках педагогической инновации / С. Д. Поляков. – М. : Дрофа, 2003. – 176 с.
37. Примерная основная образовательная программа основного общего образования (одобрена решением федерального учебно-методического объединения по общему образованию (протокол от 8 апреля 2015 г. № 1/15))
38. Савицкая, Д. Н. Элективные курсы в профильном обучении / Д. Н. Савицкая // Народное образование. – 2011. – № 12. – С. 25–27.
39. Сафонов, Г. А. Элективные курсы в профильных классах / Г. А. Сафонов // Народное образование. – 2012. – № 9. – С. 17–20.
40. Семакин, И. Г. Информатика. Углубленный уровень : учебник для 11 класса: в 2 частях, Ч.1. / И. Г. Семакин, Е. К. Хеннер, Л. В. Шестакова. –БИНОМ. Лаборатория знаний, 2014. – 176 с.
41. Синько, Т. П. Элективные курсы [Электронный ресурс]. – Режим доступа : http://den-za-dnem/page.php?article=34.
42. Угринович, Н. Д. Информатика и ИКТ. 8–11 классы : методическое пособи. - 2 изд. БИНОМ. Лаборатория знаний , 2014. – 149 с.
43. Угринович, Н. Д. Информатика и ИКТ. Профильный уровень: учебник для 11 класса / Н. Д. Угринович. – М.: БИНОМ. Лаборатория знаний, 2013.– 308 с.
44. Угринович, Н. Д. Информатика и информационные технологии : примерное поурочное планирование с применением интерактивных средств обучения / Н. Д. Угринович. – М. : Школьная пресса, 2001. – 48 с.
45. Угринович, Н.Д. Практикум по информатике и информационным технологиям. Учебное пособие для общеобразовательных учреждений/ Н.Д. Угринович, Л.Л. Босова, Н.И. Михайлова. – М.: Бином. Лаборатория Знаний, 2014. – 400 с.
46. Учеба.ру: Все образование в России и за рубежом, высшее и среднее образование за рубежом [Электронный ресурс]. – Режим доступа : http://www.ucheba.ru/for-abiturients/ege/stat
47. Федеральный базисный учебный план для образовательных учреждений Российской Федерации, реализующих программы общего образования (утвержден приказом 1312 Министерства образования РФ от 09.03.2004). / Министерство образования Российской Федерации. – М. : 2004. – 186 с.
48. Федеральный государственный образовательный стандарт высшего образования по направлению подготовки 44.03.01 Педагогическое образование (уровень бакалавриата) (утвержден приказом Минобрнауки России от 4 декабря 2015 г. № 1426) [Электронный ресурс]. – http://fgosvo.ru/uploadfiles/fgosvob/440301.pdf
49. Федеральный государственный образовательный стандарт начального общего образования (утвержден приказом Минобрнауки России от 6 октября 2009 г. № 373) [Электронный ресурс]. – Режим доступа: http://минобрнауки.рф/документы/922.
50. Федеральный государственный образовательный стандарт основного общего образования (утвержден приказом Минобрнауки России от 17 декабря 2010 г. № 1897) [Электронный ресурс]. – Режим доступа: http://минобрнауки.рф/документы/938.
51. Федеральный государственный образовательный стандарт среднего (полного) общего образования (утвержден приказом Минобрнауки России от 17 мая 2012 г. № 413) [Электронный ресурс]. – Режим доступа: http://минобрнауки.рф/документы/2365.
52. Федеральный компонент государственного стандарта общего образования. Часть II. Среднее (полное) общее образование / Министерство образования Российской Федерации. – М. : 2004. – 266 с.
53. Фомин, Г.В. Двуязычная компиляция из MSDN Visual Studio. Методическое пособие. – Южный федеральный университет, 2013. – 165 стр.
54. Формирование универсальных учебных действий в основной школе : от действия к мысли. Система заданий : пособие для учителя / [А. Г. Асмолов, Г. В. Бурменская, И. А. Володарская и др.] ; под ред. А. Г. Асмолова. – М. : Просвещение, 2011. – 159 с.
55. Цветкова, М. С. Элективный учебный проект как новая форма профильного обучения школьников / М. С. Цветкова // Профильная школа, 2011. – № 8. – С. 23–25.
56. Цветкова, М. С. Информатика. УМК для старшей школы: 10–11 классы. Базовый уровень. Методическое пособие для учителя. М. : БИНОМ. Лаборатория знаний. 2013.
57. Экзамен.ру: портал для абитуриентов и их родителей [Электронный ресурс]. – Режим доступа: http://ege.edu.ru/ru/main/satistics-ege/
58. Электронная энциклопедия – Википедия [Электронный ресурс]. – Режим доступа : http://ru.wikipedia.org/wiki.

# Приложение 1

*Теоретический материал для учащихся*

*I семестр*

1. *Введение: история Visual Basic, назначение среды программирования Visual Basic*

Обычно BASIC ассоциируется с простым в освоении и ис­пользовании средством программирования. Действительно, этот язык долго использовался в качестве учебного языка при изучении основ программирования. Это было заложено в название BASIC — Beginners All-purpose Symbolic Instructional Code, то есть многоцеле­вой код символьных инструкций для начинающих.

Visual Basic является продуктом многолетней эволюции языка BASIC. С появлением системы Windows фирма Microsoft разработала графический вариант языка BASIC. Он предназначался для создания программ, работающих в режиме диа­лога с пользователем, т.е. в визуальном режиме. По мере раз­вития компьютерной технологии Visual Basic постепенно превращался из простого языка для на­чинающих в мощный инструмент для разработки приложений.

*Visual Basic* – это система программирования, предназначенная для написания программ, рабо­тающих под управлением операционной системы Windows. Используя Visual Basic, можно разрабатывать очень сложные приложения практически для любой области современных компьютерных технологий: бизнес-приложе­ния, игры, мультимедиа, базы данных.

Причины столь широкой популярности и у Visual Basic, и у системы Win­dows примерно одинаковы: фирма Microsoft сумела такую сложную технологию, как написа­ние компьютерных программ, сделать доступной широкому кругу пользователей путем применения графического интерфейса. В визуальной среде программирования можно быстро создать дизайн диалоговых окон, с которыми будет работать пользователь, нарисовав и упорядочив их элементы точно так же, как помещаются рисунки в текст, например, при верстке газеты. При написании программ традици­онным способом (без использования графического интерфейса) система взаимодействия с пользова­телем создается на основе команд языка программирования. Очевидно, что начинающим гораздо проще освоить метод визуального программирования.

Одним из типов объектов Visual Basic являются элементы управления – это элементы, которые используются при разработке пользовательского интерфейса. С их по­мощью можно дополнять программы новыми функциями, не вникая при этом в суть их работы. Например, чтобы пользователь мог ввести в программу некоторые исходные данные, достаточно просто поместить в форму соответствующий элемент управления (например, текстовое поле). Таким образом, языки визуального программирования обла­дают неоспоримым преимуществом — можно сконцентрироваться на том, что вы хотите полу­чить от программы, а не на том, как это все запрограммировать.

Одним из основных преимуществ языка Visual Basic является возможность очень быстрого создания работоспособных приложений. Мастера, включенные в состав Visual Basic, дают воз­можность быстро создавать прототипы приложений, готовых для обсужде­ния и согласования с заказчиком. Данная реализация языка ставит его практически в один ряд с такими средства­ми разработки, как Visual C++, Delphi и другими.

Простота и мощность языка Visual Basic позволили сделать его встроен­ным языком для приложений Microsoft Office. Многие независимые разра­ботчики, например, известная своими программами в области бухгалтерского учета фирма "1C", приобретают лицензии на использование языка Visual Basic в качестве внутреннего языка своих приложений.

Фирма Microsoft интегрировала также специальную версию Visual Basic, известную под име­нем Visual Basic for Application (VBA) во все компоненты пакета Microsoft Office, Microsoft Project и некоторые другие программы. Кроме того, фирма Microsoft продала лицензию на VBA очень многим фирмам — про­изводителям программного обеспечения. Поэтому в на­стоящее время Basic уже не считается учебным языком — знание Visual Basic и его диалектов (VBA, VBScript) становится необходимостью для современ­ного программиста любого уровня.

1. *Основные понятия объектно-ориентированного визуального программирования*

В Visual Basic реализована модель объектно-ориентированного событийно – управляемого программирования. Основным понятием в объектно-ориентированном программировании является объект. *Объект* – программная конструкция, объединяющая наборы данных, программы для их обработки и интерфейс для взаимодействия с внешней средой. Важным компонентом Visual Basic является наличие библиотек типовых объектов. Эти объекты можно вставлять в программу без каких-либо изменений или изменив некоторые их свойства. Видимые или визуальные объекты Visual Basic используются для создания средства общения пользователя и приложения. Это средство принято называть пользовательским интерфейсом. Большинство объектов служат для организации ввода данных в программу, вывода данных на печатающие устройства или экран, для изменения хода работы программы. Каждый из объектов передает в приложение сообщение о произошедшем событии – нажатии клавиши, щелчке мыши, т.е. о действии оператора. В связи с этим их называют еще элементами управления. Это, например, командные кнопки, клавиши, текстовые поля, рамки и окна для рисунков и пр. Объекты обладают свойствами, могут использовать методы и реагируют на события.

*Свойства объекта* – характеристики объекта, определяющие его внешний вид и выполняемые им функции. Свойства задаются при проектировании (в окне Свойства объекта или во время выполнения приложения (команда *объект.свойство = значение свойства*в программном коде). Изменяя свойства, мы можем воздействовать на эти объекты и, тем самым, использовать их для назначенной цели. В Visual Basic свойства большинства объектов представлены в окне свойств и доступны для изменений. Свойства объектов можно разбить на несколько групп. Одни из свойств управляют внешним видом самих объектов – высота Height, ширина Width, цвет BackColor и ForeColor, шрифт надписи и т.д. Другие свойства управляют расположением объектов на окне форм: Тор - расстояние от верхнего края формы и Left – расстояние от левого края, взаимным расположением на форме и т.д. Наконец есть группа свойств, предназначенных для воздействия одних объектов на другие. Так, например свойство Picture объекта PictureBox позволяет поместить в рамку рисунок из выбранного файла.

Свойства, общие для всех объектов:

*Name:* имя объекта при обращении к нему в программе

*Enabled:* доступность для пользователя

*Visible:* видимость на экране

*Left:* расстояние от левой границы экрана (формы) до левой границы объекта

*Top:* расстояние от верхней границы экрана (формы) до верхней границы объекта

*Height, Width*: высота и ширина объекта

По умолчанию в среде VB расстояния и размеры измеряются в *твипах*(1 твип ≈ 0,018 мм).

*События объекта* (– действия, на которые объект реагирует. Эти события могут быть порождены операционной системой (системные события) или действиями пользователя. Реакция на происшедшее событие описывается разработчиком в процедуре обработки события с помощью операторов языка в окне Редактора кода.

Процедура – изолированная часть программы, предназначенная для решения локальной задачи.

Параметры – список переменных с указанием типа. При вызове процедуры параметры получают конкретные значения.

*Общий вид процедуры обработки события:*

*Sub Объект\_Событие(параметры)*

*Действия (команды языка VB)*

*End Sub*

*Методы объекта* (Methods) – набор действий, которые можно выполнить с объектом. В программном коде используют команду **объект.метод (аргументы):** метод **Move** (переместить объект), **Show** (сделать видимым объект), **Print** (вывод информации в объект), *SetFocus* (передать управление объекту). Например: *Text1. SetFocus*(курсор будет помещен в текстовое поле).

*Фрагмент программного кода:*

Private Sub command1\_Click()

End

End Sub

Private Sub HScroll1\_Scroll()

HScroll1\_Change

End Sub

Private Sub VScroll1\_Change()

Shape1.FillStyle= VScroll1.Value

Label3.Caption = VScroll1.Value

End Sub

1. *Среда Visual Basic 2010 Express*
2. Начало работы: среда Visual Basic может быть запущена из Главного меню (как и другие приложения Windows).

Пуск  Программы  MicrosoftVisualBasic 2010 Express; для начала работы над новым проектом необходимо ввести команду:

Создать проект Приложение WindowsForms (Рис. 1).

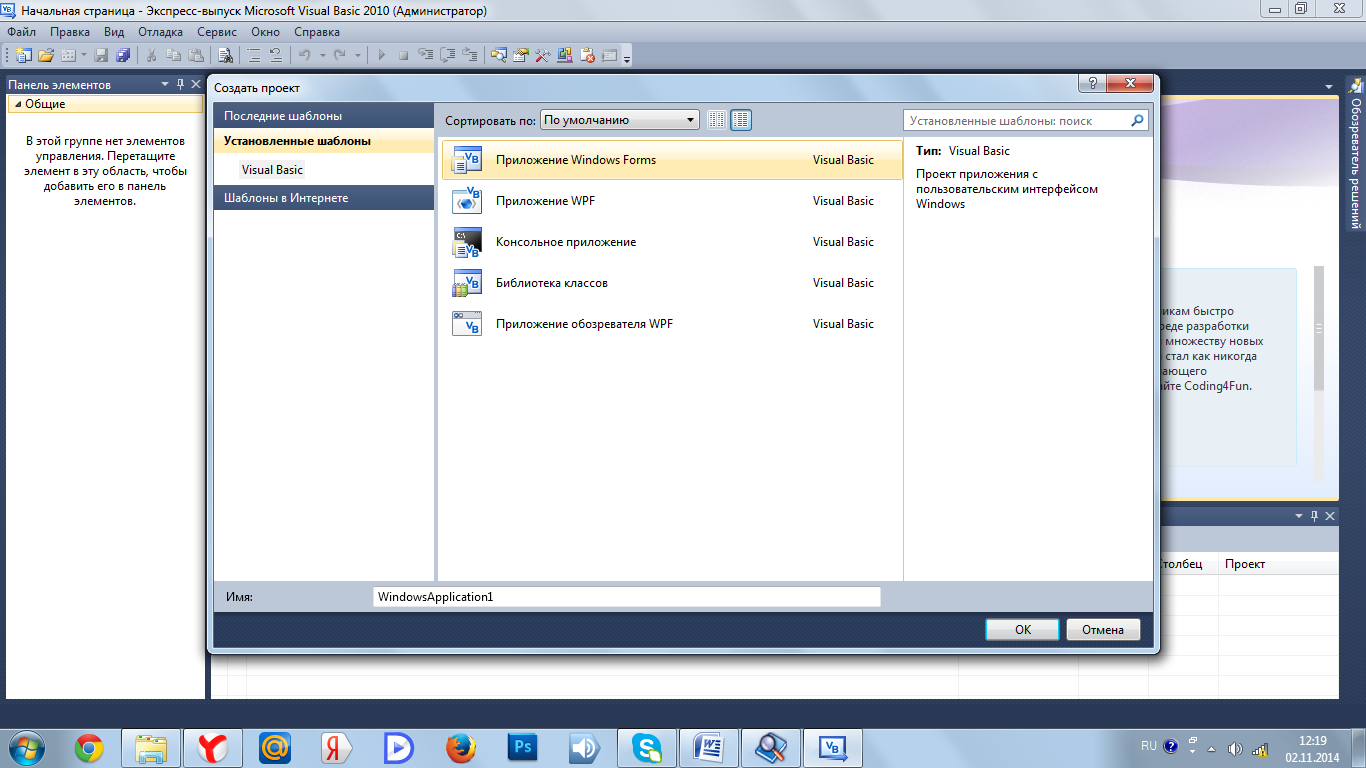


Рис. 2 Окно создание проекта

Обычно программы на алгоритмических языках программирования выполняются от начала до конца согласно алгоритму. Событийно – управляемое приложение на Visual Basic работает совершенно по-другому: выполнение приложения не идет предопределенным путем, а зависит от сложившейся ситуации. Программа является набором независимых частей кода, активизирующихся в ответ на события, происходящие при работе пользователя с приложением. Поэтому вместо создания программы, отражающей мысли программиста, получается приложение, управляемое пользователем. Большая часть кода программы, отвечающая за обработку таких событий, как щелчки мыши, нажатие клавиш, размещается в процедурах обработки событий. Программы с такой структурой легче и создавать, и отлаживать.

*Сохранение проекта*

Для сохранения проекта VB необходимо в меню *Файл*  выбрать команду   *Сохранить все*. В Lazarus сохранение проекта происходит аналогичным способом, воспользовавшись командой *Проект → Сохранить проект как…,* откроется диалоговое окно сохранения программного кода *Сохранить WindowsApplication1(Рис.2)*

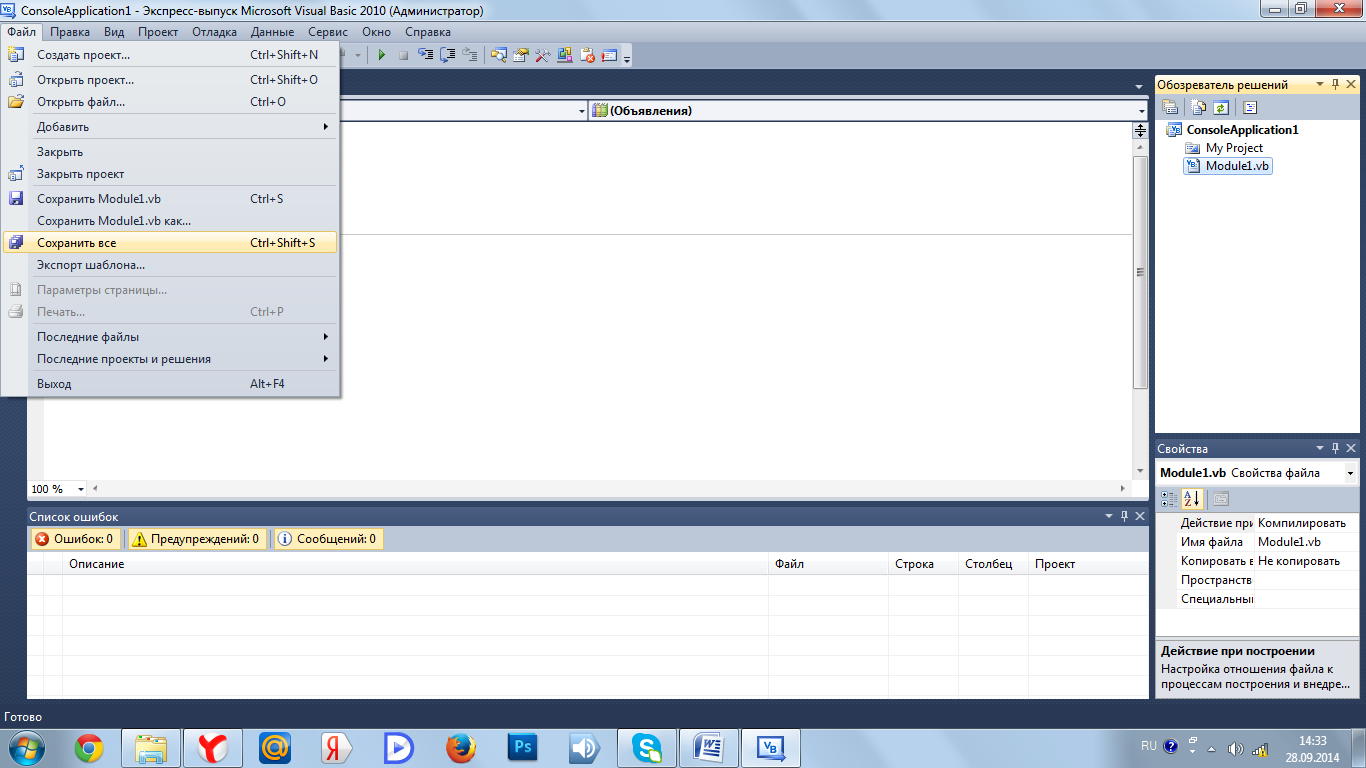


Рис. 3 Окно меню Visual Basic

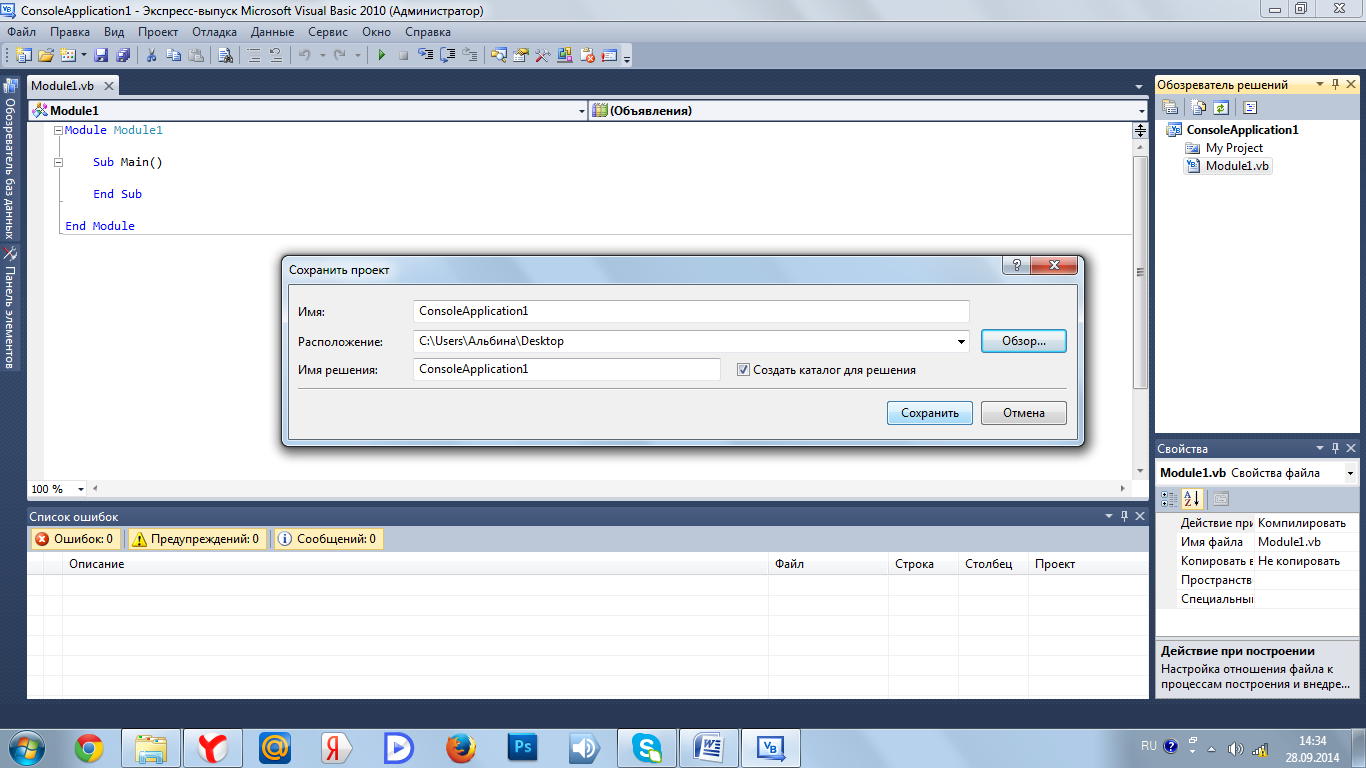


Рис. 4 Сохранение проекта Visual Basic

*Проект* – приложение на стадии разработки.

1. *Интерфейс*

После выполнения выше перечисленных действий на экране откроется окно среды разработки: *WindowsApplication1.*

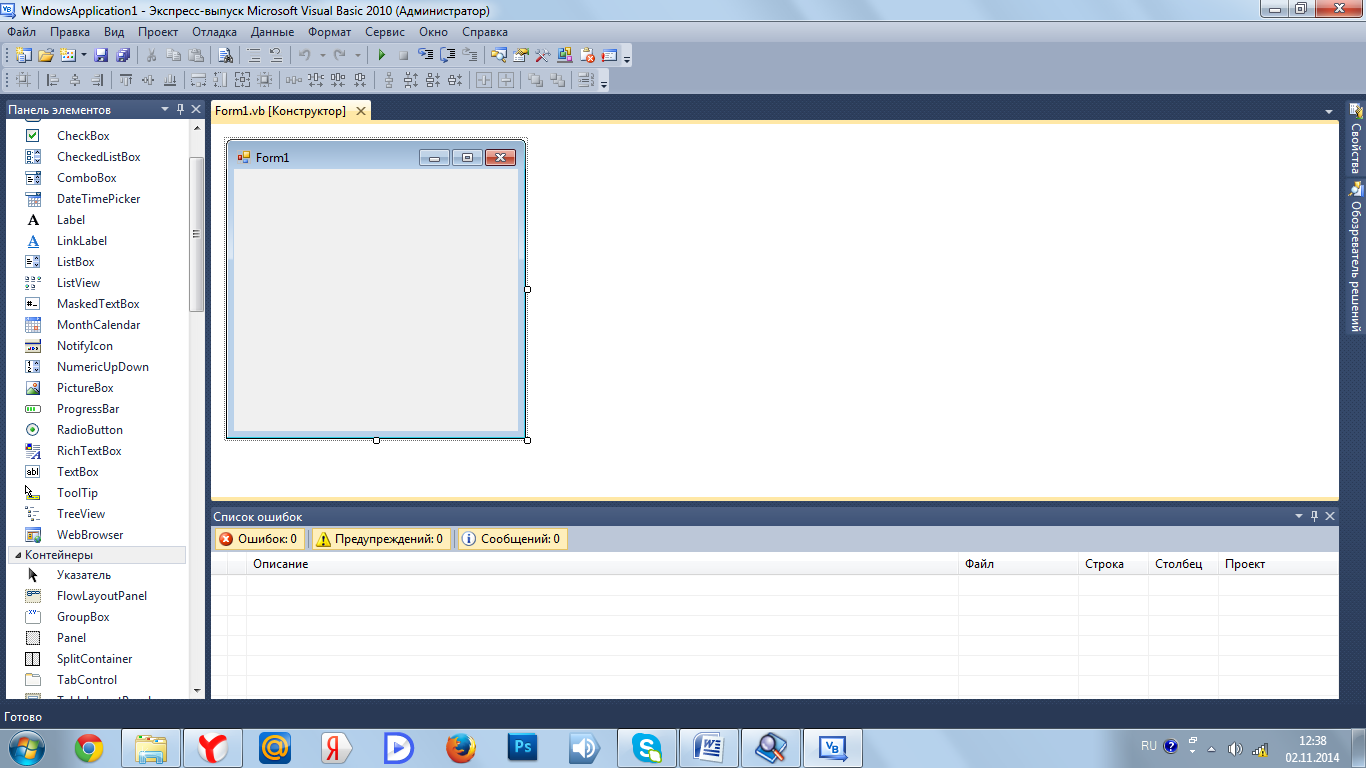


Рис. 5 Окно среды разработки *WindowsApplication1*

Среда Visual Basic является интегрированной: она предоставляет разработчику широкие возможности конструирования графического интерфейса приложения, редактирования методов и свойств объектов, отладки, тестирования и выполнения проекта.

Основными элементами среды являются:

А) *Главное окно среды* напоминает окна Windows: есть строка заголовка, главное меню и панель инструментов.

*Меню.* В состав этого элемента входят все команды, которые используются при создании приложений в среде разработки . Среди пунктов меню есть как стандартные, присущие большинству окон Windows*(Файл, Правка, Вид, Справка, Сервис)*, такие пункты, при помощи которых разработчик может создавать, запускать или отлаживать своё приложение *(Проект, Откладка, Данные)*. Меню расположено в верхней части MicrosoftVisualBasic под заголовком окна.

*Контекстное меню MicrosoftVisualBasic* cодержит команды, доступные для данного объекта в данный момент времени. Чтобы открыть такое меню, необходимо щелкнуть правой кнопкой мышки на активном объекте. Список доступных команд определяется самим объектом.

*Панель инструментов.* Эти элементы представляют собой различные группы кнопок с изображенными на них иконками. При помощи данных кнопок можно выполнять те или иные операции, предусмотренные в среде разработки, не используя при этом меню; то есть выполнять одно действие нажатие соответствующей кнопки вместо нескольких, в случае выполнения этой же операции при помощи меню. Панели инструментов расположены в верхней части, под элементом меню.

##### А

Таблица 1.

**Панель инструментов Visual Basic**.

.

|  |  |  |
| --- | --- | --- |
| **Кнопка** | **Название** | **Значение** |
|  | Создать новый проект | Используется для создания нового проекта. |
|  | Добавить новый элемент | Предназначен для добавления нового элемента в проект. |
|  | Сохранить Module1.vb | Используется для сохранения проекта с расширением .vb |
|  | Сохранить все | Предназначен для сохранения файлов, входящих в состав открытого проекта. |
|  | Найти | Используется в случае необходимости поиска фрагмента текста в коде программы |
|  | Закомментировать выделенные строки | Используется для отображения списка доступных объектов в данном проекте |
|  | Отменить преобразование выделенных строк в комментарии | Используется для отмены выделенных строк комментарий. |
|  | Начать откладку | Служит для запуска приложения на выполнение |
|  | Остановить откладку | Применяется в случае необходимости прерываниявыполнения программы (пауза) |
|  | Шаг с заходом | Используется для вывода программы с заходом в саму программу |
|  | Шаг с обходом | Используется шаг для обхода сомой программы. |
|  | Шаг с выходом | Используется шаг для выходы из программы. |
|  | Обозреватель решений | Применяется для обзора решений. |
|  | Окно свойства | Применятся в том случае, если нужно для  выделенного элемента посмотреть или изменить какое-либо свойство |
|  | Панель элементов | Применяется с целью отображения панели  инструментов в среде разработки |
|  | Диспетчер расширений | Применяется для выбора расширения. |
|  | Список ошибок | Используется для вывода на экран ошибок. |
|  | Интерпретация | Связана с получением переменными значений в процессеработы программы. |

Б) *Дизайнер форм* служит в качестве окна, которое можно настроить при разработке интерфейса приложения. Программист добавляет элементы управления, графику и картинки на форму для придания, ей желаемого вида. Каждая форма приложения имеет окно дизайнера форм.

##### В

##### ДА

##### Г

##### Б

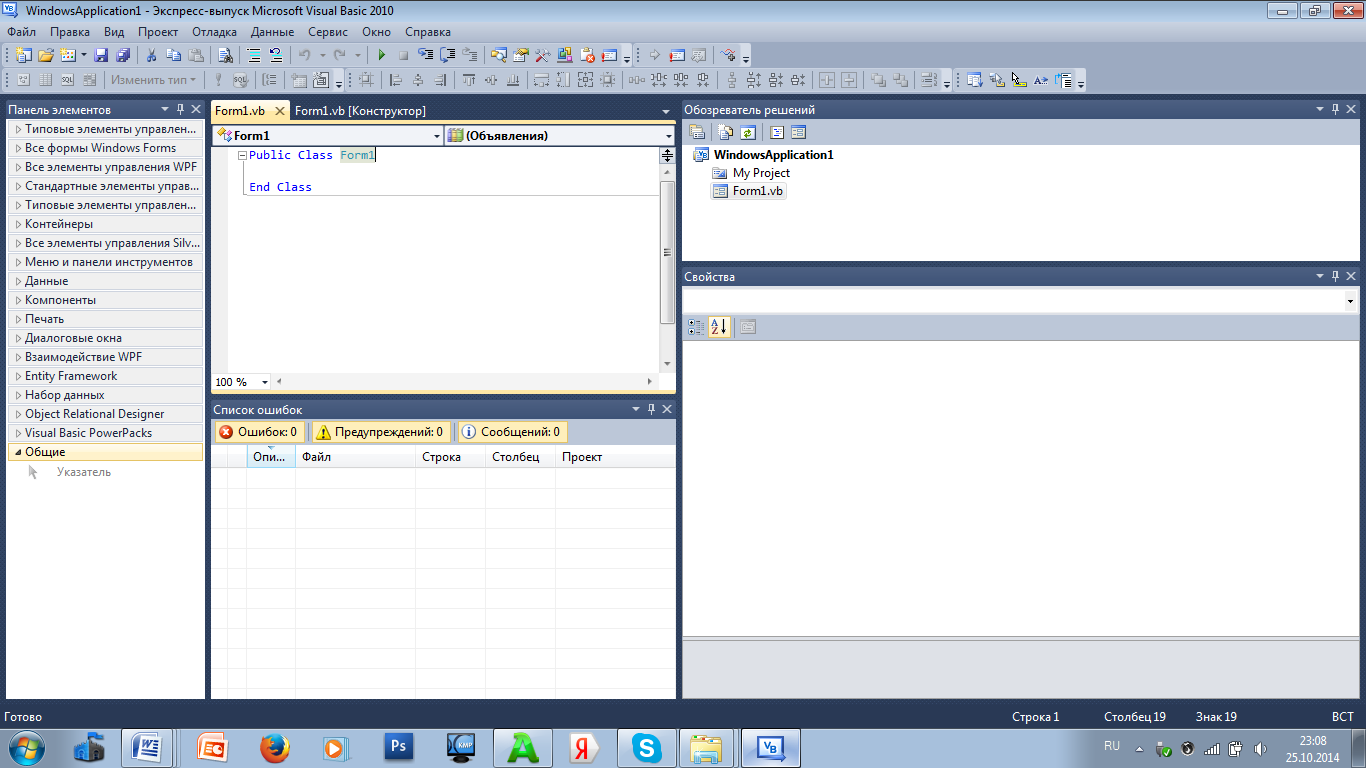
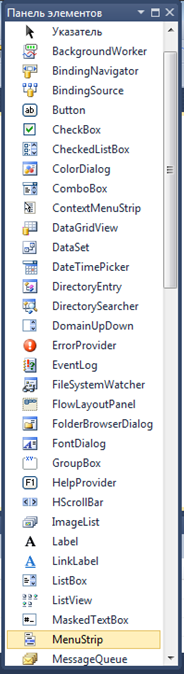
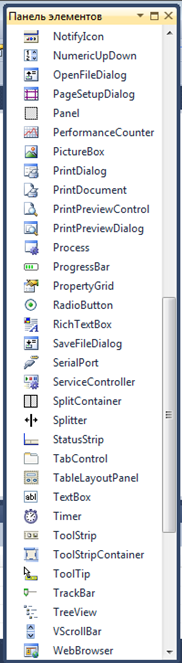
При запуске редактора кода и в VB (Рис. 6)появляется окно, в котором можно вводить текст программы (программный код). Это окно служит для редактирования вводимого *программного кода приложения.* Для каждой формы или модуля кода приложения создается отдельное окно редактора кода. В какой-то мере редактор похож на специализированный текстовый процессор с набором функций, облегчающих ввод текста программы.

Рис. 6 Окно редактора программного кодаVisuailBasic

В) *Панель элементов управления* MicrosoftVisualBasic. При помощи данного элемента визуальной среды разработчик может размещать на форме те или иные элементы управления, которые необходимы для создания приложения.

Ниже представлены объекты, находящиеся на панели элементов Mi-crosoft Visual Basic.

В таблице 2 приведены названия и функциональное предназначение основных элементов управления в MicrosoftVisualBasic.

Таблица 2.

**Элементы управления Visual Basic**

|  |  |  |
| --- | --- | --- |
| **Элемент** | **Название** | **Назначение** |
|  | Указатель | Выбор элемента управления |
|  | Border (границы) | Выбор границ |

|  |  |  |
| --- | --- | --- |
|  | Button (кнопка) | Выполнение приложением определённых действий |
|  | Checkbox (флажок) | Установка/отключение настроек |
|  | Combobox (поле со списком) | Выбор элемента из раскрывающегося Списка |
|  | DataGrindVien(данные) | Отображает строки и столбцы данных в сетки, которые пользователь может изменить |
|  | Image (изображение) | Вывод графического элемента в форме |
|  | Label (метки) | Отображение надписей в форме |
|  | ListBox (список) | Выбор пользователем какого-либо элемента из списка |
|  | RadioButton (радио кнопки) | Выбор пользователем одного из нескольких возможных пунктов |
|  | Rectangle(прямоугольник) | Объединение в группу различных элементов управления (такие объекты называются контейнерами) |
|  | StackPanel (стек понели) | Выбор пользователем какого-либо элемента из тех, которые имеются в списке |
|  | TabControl (табло контроля) | Выбор пользователем каталога на диске |
|  | TextBlock (текстовый блок) | Отображение надписей в форме |
|  | TextBox (текстовое поле) | Ввод текста |
|  | Timer (время) | Компонент, создающий события с интервалом, определенным пользователем |
|  | Dateset | Представляет кэш данных в памяти |
|  | Groupbox | Отображает рамку вокруг группы элементов управление с возможностью включения заголовка |
|  | Panel | Позволяет пользователю группировать коллекции элементов управления |

Г) *Окно Свойства объекта* используется, чтобы задать свойства формы и размещенных на ней объектов на этапе проектирования. Это окно содержит перечень тех свойств объекта, которые пользователь может изменить. Окно выполнено в виде таблицы, каждая строка которой содержит два поля. В левом поле записано название свойства, в правом - значение свойства. Значения свойств можно изменять непосредственно в окне Свойства.

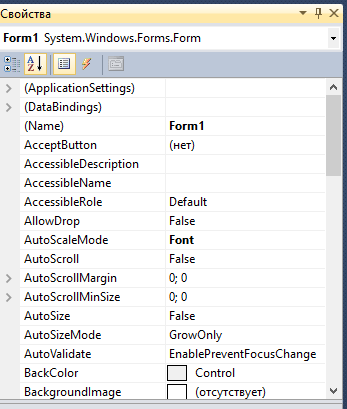


Рис. 7 Окно свойств Visual Basic

1. *Понятие проекта*

Программы, работающие в среде Windows, называются приложениями. На этапе разработки приложения в среде Visual Basic называются *проектами*. Проект содержит набор взаимодействующих форм, программных модулей, модуль проекта и вспомогательные файлы.

*Форма* – основной элемент внешнего интерфейса проекта, аналог окна Windows. Она имеет строку заголовка с кнопками управления и системным меню, а также возможности управления мышью. Все это не надо программировать, форма, включённая в проект, обладает этими свойствами. Форма хранится в файле с расширением frm, она предназначена для размещения элементов настройки и управления проекта, для обмена информацией с пользователем при работе приложения. Это сложный программный объект, имеющий характерные свойства, методы, события. Кроме того, в файле формы хранится описание помещенных на нее элементов управления, программный код процедур, относящихся к данной форме. Обычно в проекте бывает несколько форм.

Программный модуль не имеет внешнего интерфейса, на него нельзя поместить элементы управления. Он содержит программный код процедур, которые могут использоваться в разных формах и в самом программном модуле. Обычно в проекте один программный модуль, он хранится в файле с расширением bas.

Модуль проекта содержит код, описывающий связи между отдельными элементами проекта и определяющий порядок их выполнения. Хранится в файле с расширением vbp.

1. *Этапы разработки приложения в среде Visual Basic*
2. *Создание интерфейса (этап проектирования)*

На этом этапе необходимо сначала продумать ожидаемый алгоритм работы пользователя с будущим приложением, возможные события при работе приложения, порядок их возникновения. Кроме того, надо выявить будущих пользователей, максимально точно описать предъявляемые ими требования к проекту.

На форму с помощью мыши помещаются те или иные управляющие элементы, а в окне Properties задаются их свойства. После того как разработка интерфейса завершена, кнопки управления, поля и другие элементы, размещенные на форме, автоматически будут распознавать действия пользователя, такие, например, как движение мыши или щелчок ее кнопки.

1. *Создание программного кода*

Теперь начинается процесс, похожий на традиционное программирование: создание программного кода для активизации визуального интерфейса, подготовленного на первом этапе. Смысл состоит в том, что объекты Visual Basic только распознают события типа щелчков мыши, а то, как они реагируют на эти события, зависит от программного кода, написанного программистом. При этом, алгоритмическая часть Visual Basic практически ничем не отличается от традиционного языка Basic.

Для создания или редактирования кода надо вызвать *окно Редактора кода.* Окно используется для размещения, просмотра и редактирования всех текстовых элементов программы - описания констант, переменных, массивов, процедур и пр.

1. *Отладка, тестирование, компиляция*

Для выполнения отладки (проверки правильности функционирования проекта и исправления найденных ошибок) в Visual Basic существует набор специальных инструментов, в первую очередь команды меню Debug. Отладка и тестирование – непременные этапы работы над любым проектом, особенно большим и сложным. Проект может функционировать лишь в среде Visual Basic. После его создания, отладки и тестирования выполняется компиляция: создается исполняемый модуль с расширением exe, независимый от среды Visual Basic. Так получается готовое приложение.

Форма и некоторые стандартные элементы управления

*Графический интерфейс* необходим для реализации интерактивного диалога пользователя с работающим приложением. Основой для создания графического интерфейса разрабатываемого приложения являются форма, представляющая собой окно, в котором размещаются управляющие элементы. Графический интерфейс проекта может включать в себя несколько форм.

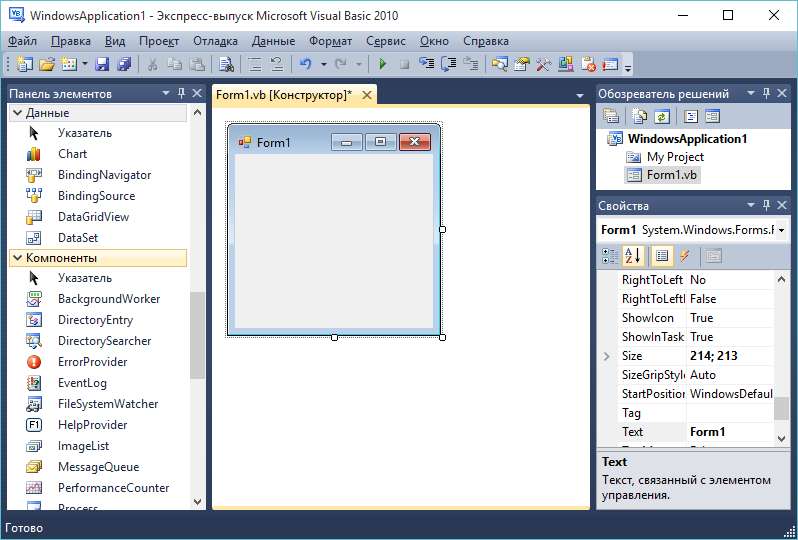


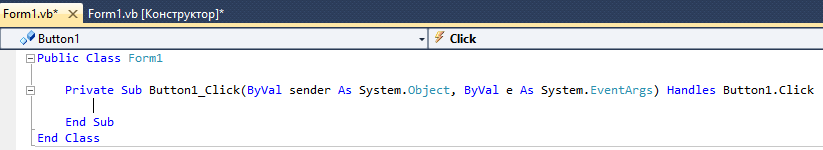
Рис. 8 Интерфейс среды разработки VB

*Форма (Form)* – контейнер, содержащий другие элементы управления, аналог окна Windows.

Основное внимание при создании данного проекта уделим технологии создания кода.

Вызов окна Редактора кода:

* выполнить двойной щелчок по объекту, для которого создаем процедуру. Ниже представлен текст в окне Form1.vb\*, образовавшийся в результате двойного щелчка по командной кнопке с именем Button1:



Первая строка *Public Class* – служебная, она открывает раздел описаний. Следующая пара строк представляет собой заготовку процедуры, определяет ее начало и конец. Между этими двумя строчками размещают тело процедуры.

Полезно обратить внимание на очень удобный сервис, предоставляемый редактором окна кодов: как только справа от названия объекта будет поставлена точка, на экране появляется всплывающее меню с перечнем всех доступных изменению свойств данного объекта. Достаточно выбрать из списка двойным щелчком название нужного свойства, как оно вставляется в текст процедуры. Использование этого сервиса помогает избежать орфографических ошибок при написании незнакомых слов.

1. *Метка*

*Метка (Label)* – элемент управления, применяемый для вывода информации. Другое название объекта - этикетка.

1. *Командная кнопка*

*Командная кнопка (Button)* – элемент управления, применяемый для интерактивного управления работой приложения.

1. *Текстовое поле*

*Текстовое поле (TextBox)* – элемент управления, применяемый для ввода и вывода информации.

1. *Переменные*

Имя: латинские и русские буквы, цифры, символ подчеркивания. Начинается с буквы. Максимально 255 символов.

Тип:

1) Числовые:

*Byte:*целые числа от 0 до 255

*Integer:* целые числа от -32768 до 32767

*Long:* длинные целые

*Single:* вещественные одинарной точности

*Double:* вещественные двойной точности

*Currency:* денежный формат (числа с фиксированной точкой)

2) Символьные:

*String:* переменной длины

*String\*n:* фиксированной длины

3) Особые типы:

*Boolean:* логический тип; *Date*: тип дата/время

*Object:* объектный тип; *Variant*: стандартный тип (по умолчанию)

Объявление переменных:

Явное объявление (с помощью специального оператора *Dim, Private, Static, Public): Dim имя переменной [As тип переменной]*

Например: *Dim f As Integer*

Неявное объявление (с помощью специального символа *%, &, !, #, @, $*после имени переменной): S% (тип Integer), Y# (тип Double)

Время жизни переменных:

*Динамические:* существуют, пока выполняется породившая их процедура.

*Статические*: сохраняют свои значения после окончания работы процедуры. Их необходимо объявлять: *Static имя переменной [As тип переменной].* Такие переменные часто используют для подсчета количества.

1. *Зона видимости переменных:*

*Локальные:*

А) локальные для процедуры: описываются внутри конкретной процедуры оператором *Dim* или *Private.* Такие переменные нельзя использовать из других процедур.

Б) локальные для формы: описываются в разделе общих объявлений General Declaration этой формы оператором *Dim*. Такие переменные можно использовать из всех процедур кода для данной формы.

*Глобальные:* описываются в разделе общих объявлений General Declaration программного модуля проекта (файл с расширением bas) оператором *Public*. Такие переменные можно использовать из всех процедур проекта.

*Пример:* Психологи утверждают, что для того, чтобы брак был счастливым, должна быть определенная разница в возрасте между женихом и невестой: возраст невесты должен быть равен половине возраста мужчины плюс семь, а возраст жениха – удвоенному возрасту невесты минус четырнадцать. Разработать приложение для определения подходящего возраста будущего супруга.

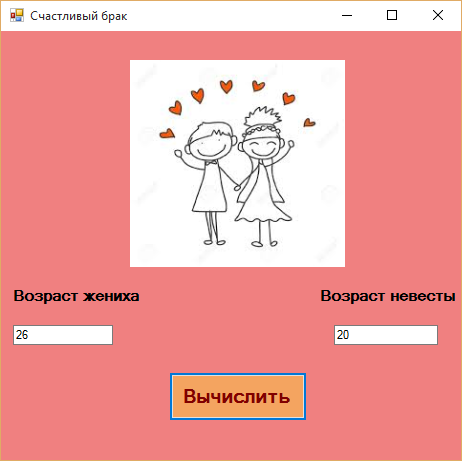


Рис. 9 Интерфейс проекта «Счастливый брак»

*Программный код:*

Option Explicit

*'логические переменные, локальные для формы*

Dim f1, f2 As Boolean

Private Sub Button1\_Click()

*'переменные, локальные для процедуры*

Dim vg, vn As Byte

vg = Val(TextBox1.Text)

vn = Val(TextBox2.Text)

*'вычисляем возраст жениха или невесты*

If f1 = True Then

vn = vg / 2 + 7

TextBox2.Text = vn

Else

vg = vn \* 2 - 14

Text1Box.Text = vg

End If

TextBox1.Enabled = True

TextBox2.Enabled = True

End Sub

Private Sub TextBox1\_Change()

*'запоминаем, в какое текстовое поле*

*'вводили информацию*

f1 = True

f2 = False

*'второе текстовое поле не доступно*

*'для ввода информации с клавиатуры*

TextBox2.Enabled = False

End Sub

Private Sub TextBox2\_Change()

f2 = True

f1 = False

TextBox1.Enabled = False

End Sub

1. *Встроенные константы Visual Basic*

В Visual Basic предусмотрено большое количество встроенных констант: коды цветов (vbRed, vbBlue и др.), коды клавиш (vbTab, vbCrLf), тип пиктограмм, набор кнопок в диалоговых окнах (vbInformation, vbCritical, vbOK, vbCancel, vbYes) и другие. Константы подробно описаны в справочной системе Visual Basic. Кроме того, значение и описание большинства внутренних констант можно посмотреть, воспользовавшись браузером объектов. Основная часть окна Обозревателя разделена на две половины: Classes (Классы) и Members (Члены). Обозреватель позволяет выбрать библиотеку, просмотреть в левой части окна названия классов объектов в алфавитном порядке. В правой половине представлены названия свойств, методов, функций, событий, принадлежащих выбранному объекту, а также встроенные константы. В нижней части окна отображается название используемой библиотеки, синтаксис написания выбранного элемента и комментарий. Обозреватель можно использовать, например, как справочник для правильного написания названий функций, свойств и методов в коде программы.

1. *Диалоговые окна*

Это специальные элементы интерфейса интерактивной программы, предназначенные для ввода и отображения информации.

*Окно ввода*: для ввода текстовой информации пользователем во время работы приложения. Вид команды в программном коде:

Символьная переменная = *InputBox*( «текст сообщения», «заголовок окна», «значение по умолчанию»)

Например:   
nm = InputBox("Как тебя зовут?", "Знакомство")

В результате на экран выводится диалоговое окно. Введенная с его помощью информация помещается в символьную переменную и может быть использована при работе приложения. Если пользователь не ввел информацию в текстовое поле окна, нажал Cancel или кнопку закрытия окна, то в символьной переменной будет храниться пустая строка.

1. *Окно сообщений*: выдается на экран во время работы приложения.

Оператор *MsgBox:*

*MsgBox «текст сообщения», встроенные константы, «заголовок окна»*

Константы определяют, сколько и каких кнопок будет в окне, а также тип отображаемой в окне пиктограммы (критическая ошибка, запрос, предупреждение, информация).

Например: MsgBox "Наберите имя и нажмите Ok", vbInformation, "Неверный ввод"

Функция *MsgBox***,** в отличие от оператора, возвращает значение:

Числовая переменная = *MsgBox* («текст сообщения», встроенные константы, «заголовок окна»)

Например: ex = MsgBox("Вы действительно хотите закончить работу с приложением?", vbYesNo, "Завершение работы")

1. *Массивы*

*Массивы переменных:* Это наборы переменных одного типа, объединенные общим именем. В школьном курсе рассматриваются одномерные массивы (последовательности) и двумерные (таблицы). Каждый элемент массива имеет имя, тип, порядковый номер (индекс), значение.

Объявление массива: Dim A(10) As Single (в массиве 11 элементов вещественного типа (нумерация индексов с 0)); Dim C(1 to 20) As Integer (в массиве 20 элементов целого типа).

Способы заполнения массива:

С помощью оператора присваивания:

For i = 1 To n

x(i) = Int(Rnd \* 100)+1 ' заполнение случайными числами из отрезка [1 ; 100]

Next i

Ввод массива с клавиатуры:

For i = 1 To n

x(i) = InputBox("x(" & Str(i) & ")=", "Ввод массива с клавиатуры")

Next i

Массивы элементов управления: использование массивов элементов управления экономит системные ресурсы во время работы приложения и значительно сокращает программный код. Чтобы *создать массив элементов*, надо поместить на форму первый из них, скопировать в буфер обмена, вставить на форму и утвердительно ответить в появившемся диалоговом окне на запрос   
о создании массива.

Пример: на форме создать массив из 3 меток.

Замечание: элементы массива будут иметь одинаковое имя Label1 и отличаться значением свойства Index: 0, 1, 2.

1. *Флажки, переключатели, фреймы*

*Переключатели (Option Button):* обычно образуют группу и помещаются внутри фрейма, обеспечивая для пользователя возможность выбора 1 из предложенных вариантов.

*Флажки (Check Box):* используются для размещения на форме данных, которые могут иметь одно из 2 значений. Если флажки образуют группу, то обеспечивают для пользователя возможность выбора нескольких вариантов из предложенных.

Создание группы элементов:

1) на форме разместить фрейм установить для него необходимые свойства;

2) поместить на форму элемент управления (переключатель, флажок и др.), установить его свойства, вырезать его в буфер обмена;

3) выделить фрейм и вставить элемент из буфера обмена, затем разместить его в нужном месте во фрейме;

4) вставить следующие элементы группы, включив их в массив.

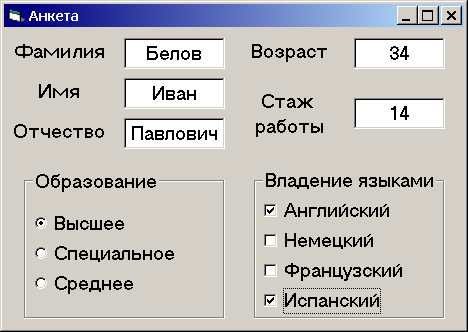


Рис. 10 Пример использования флажков, переключателей, фреймов

1. *Списки*

*Обычный список (List Box)*

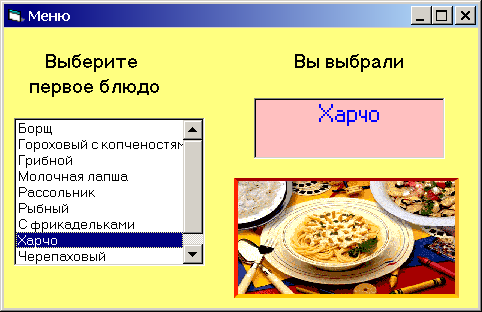


Рис. 11 Пример использования элементов списка

Заполнение списка:

На этапе проектирования значения элементов вводятся в свойстве List (после каждого значения нажимается сочетание клавиш Ctrl +Inter)

Во время выполнения приложения используется метод AddItem:

*For i=1 to 100*

*List1.AddItem i ‘список заполняем числами от 1 до 100*

*Next i*

Удаление элементов из списка:

Для очистки всего списка используется метод Clear (команда List1.Clear)

Для удаления элемента с указанным номером используется метод RemoveItem, например: List1. RemoveItem 5 (удаляется элемент списка с № 5)

*Комбинированный список (Combo Box)*

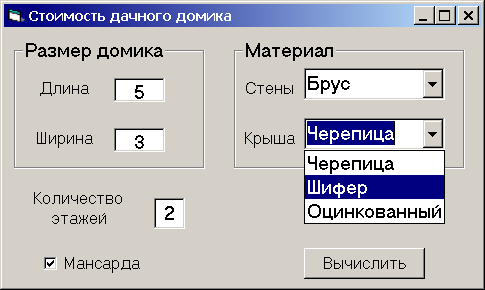


Рис. 12 Пример использования комбинированного списка

Этот элемент позволяет выбирать данные из списка и добавлять новые элементы в текстовое поле списка (сочетает в себе список и текстовое поле).

1. *Полосы прокрутки (VScrollBar, HScrollBar)*

Используются для ввода параметра, значения которого изменяются в некотором диапазоне.

События: *Scroll* (возникает при перемещении бегунка на полосе прокрутки),*Change*(возникает после перемещения бегунка в момент отпускания кнопки мыши, после щелчка мыши по стрелке на краях полосы прокрутки, после щелчка мыши по полосе прокрутки).

*Пример:* проект «Скорость» демонстрирует удобный способ ввода данных во время работы приложения с помощью полосы прокрутки (при перемещении бегунка любым из указанных выше способов соответственно изменяется значение скорости).

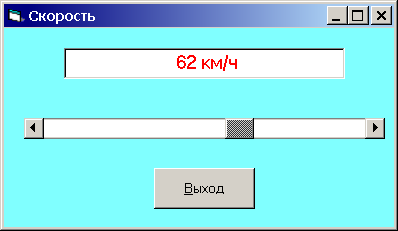


Рис. 13 Пример использования полосы прокрутки

*Теоретический материал для учащихся*

*II семестр*

## *Графические методы и элементы управления Visual Basic*

Без использования графики интерфейс созданного приложения будет скучным, а часто и недостаточно понятным. Графика широко применяется для выделения определенной информации на экране, для разнообразия вариантов отображения информации на экране (например, построение графиков, диаграмм), для поддержки интуитивно-понятного интерфейса приложения.

В Visual Basic графику можно применять двумя основными способами: с помощью графических методов или элементов управления.

### *Понятие координатной системы*

При использовании графических методов или элементов управления нужно описывать, где именно на форме (в графическом поле) должен располагаться нужный элемент или нарисована геометрическая фигура, то есть необходимо задать координаты. В отличие от математики, в Visual Basic значение вертикальной координаты Y возрастает сверху вниз. Таким образом, получаем систему координат указанного вида:

## X

## Y

(0, 0)

Стандартной единицей измерения в системе Visual Basic является *твип* (1/1440 дюйма). Это более точная единица измерения, чем привычные для пользователей пиксели, размер которых зависит от установленного в данный момент разрешения экрана.

### *Графические методы*

Графические методы – это функции, которые содержатся в языке Visual Basic и вызываются во время работы приложения. Их можно применять для рисования изображений в форме, в графическом поле PictureBox, а также для объекта Printer. В среде Visual Basic с помощью графических методов можно решать уже ставшие традиционными графические задачи: «Неоновая реклама», «Мыльные пузыри», «Конфетти», «Падающий снег», «Змейка», строить орнаменты, имитировать движение геометрических фигур по разным траекториям и многое другое (как, например, в языке Basic). В то же время Visual Basic предоставляет новые возможности при работе с графикой (например, богатая цветовая палитра, разнообразные способы закраски, выбор единиц измерения, возможность установить новый масштаб, более простой способ создания анимации и др.).

*Основные графические методы:*

1. *Сls: объект.Cls*

Удаляет графические изображения, созданные в результате работы графических методов.

1. *Pset: объект.Pset(X,Y) [, цвет]*

Построение точки, например, команда Pset (300,100), RGB(0, 0, 255**)** строит точку на текущей форме с координатами x = 300, y = 100 ярко-синего цвета.

1. *Point: объект.Point(X,Y) Возвращает RGB-цвет точки.*
2. *Line: объект.Line(X1,Y1)-(X2,Y2) [, цвет, B/BF]*

Построение линии, прямоугольника (если указан параметр B), закрашенного прямоугольника (если указан параметр BF), например:

Команда *Line(200, 300) – (600, 800), vbRed* строит на форме линию красным цветом, а команда *Line(200, 300) – (600, 800), vbBlue*, В строит на форме прямоугольник (прямоугольник задается своей диагональю).

Если значения (X1,Y1) в команде не указано, то построение начнется в текущей точке, например, *Line - (2000, 300).*

1. *Circle:*Построение окружности, эллипса, дуги, сектора.

*объект.Circle(X,Y), радиус [ цвет, начальн. угол, конечн. угол, коэффициент сжатия]*

Здесь (X,Y) – координаты центра, начальный и конечный углы задаются в радианах для построения дуг (от 0 до 2π или от 0 до -2π) и секторов (углы должны принимать отрицательные значения), коэффициент сжатия задается для построения эллипса.

Пример: Команда *Circle(1000, 1000), 300, , 3\*Pi/4, Pi/3* строит дугу окружности, а команда *Circle(1000, 1000), 300, , , , 2* строит вытянутый по вертикальной оси эллипс.

1. *Print: объект. Print числовое или строковое выражение*

Вывод текста, начиная с текущей позиции, например, команда *Print “Значение функции равно”; y*выводит на форму, начиная с текущей позиции, ответ задачи.

Если в методе Print несколько значений, они могут разделяться запятой (тогда каждое значение печатается в своей колонке, ширина которой 14 символов) или точкой с запятой (тогда значения печатаются в одной строке рядом друг с другом).

Пример:*Print “y= “; y, “z= “; z*

Каждый метод Print выводит значения в новую строку. Когда, вызывая новый метод Print, необходимо продолжить печать в той же строке, надо в конце предыдущей команды Print поставить запятую или точку с запятой: *Print “Расстояние “; : Print s*

В методе Print можно использовать функции *Spc(n)* (задает количество пробелов перед печатаемым значением) и *Tab(n)* (выполняет печать, начиная с n-ой позиции строки), например: команда *Print Tab(20) “задача”* печатает слово “задача”, начиная с 20-ой позиции строки.

*Способы задания цвета:*

1. В Visual Basic имеются встроенные константы для обозначения некоторых цветов: vbBlack, vbRed, vbGreen, vbYellow, vbBlue, vbMagenta, vbCyan, vbWhite.
2. Значение цвета можно определить также с помощью функции *RGB(r, g, b),*где параметры r, g, b принимают целые значения от 0 до 255 и определяют интенсивность красной, зеленой и синей составляющей цвета. Этим способом можно задать более 16 миллионов цветов. Например, следующая команда строит линию серого цвета (поровну красного, зеленого и синего компонента): *Line(100, 100) – (300, 400), RGB(200, 200, 200)*
3. Еще одна полезная функция для работы с цветом *QBColor(c),* где параметр с принимает целые значения от 0 до 15, а функция возвращает соответствующее шестнадцатеричное значение цвета. Например, команда *Pset(200, 500), QBColor(4)* строит на форме точку красным цветом.

Замечание: Если в графическом методе цвет явно не определен, то для построения соответствующего элемента используется цвет, указанный в свойстве ForeColor текущего объекта.

При работе с графическими методами используются свойства объектов (формы, графического поля, принтера), которые воспроизводят это изображение:

*ForeColor:* значение цвета точки в методе Pset и цвета границ фигур в методах Line, Circle (если цвет явно не указан в методе), а также цвет текста в методе Print.

*DrawWidth:* размер точки в методе Pset и толщина линий в методах Line, Circle.

*DrawStyle:* стиль линии в методах Line, Circle.

*FillColor:* цвет заполнения контура в методах Line, Circle.

*FillStyle:* стиль заполнения в методах Line, Circle.

*Font:* шрифт и размер текста в методе Print.

*CurrentX, CurrentY:* координаты текущей точки в методе Line и первого символа в методе Print (эти свойства задаются только в программном коде).

Например, построение треугольника, начиная с указанной текущей точки:

CurrentX=1500 : CurrentY=500

Line - (3000, 2000) : Line - (1500, 2000) : Line - (1500, 500)

*AutoRedraw:* перерисовка изображения. Определяет, будет ли результат работы графических методов автоматически обновляться в окне, если окно было скрыто или свернуто.

*Пример1:* определить, находится ли точка, на которой пользователь щелкнул мышью, внутри красного прямоугольника, нарисованного во время загрузки формы (проект «Прямоугольник»).

*Private Sub Form\_Load()*

*Form1.AutoRedraw = True*

*Line (100, 200)-(1500, 2500), vbRed, BF*

*End Sub*

*Private Sub Form\_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)*

*If Point(X, Y) = vbRed Then*

*MsgBox "Точка (" & X & "," & Y & ") находится в прямоугольнике"*

*Else*

*MsgBox "Точка (" & X & "," & Y & ") находится вне прямоугольника"*

*End If*

*End Sub*

*Пример2:* после щелчка по кнопке Построить на форме строится линия произвольного размера и цвета. В проекте предусмотрена возможность удалить изображение.

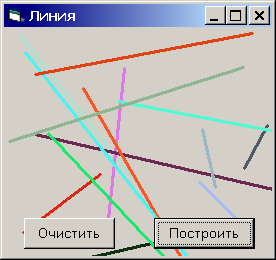
**

Рис. 13 Пример построения линий в среде VB

*Private Sub Command1\_Click()*

*Randomize ‘дает возможность при каждом запуске программы генерировать с помощью ‘Rnd разные числовые последовательности*

*Line (Rnd \* Form1.Width, Rnd \* Form1.Height)-(Rnd \* Form1.Width, Rnd \* Form1.Height), \_RGB(Rnd \* 255, Rnd \* 255, Rnd \* 255)*

*End Sub*

*Private Sub Command2\_Click()*

*Cls*

*End Sub*

*Пример3:* при нажатии левой кнопки мыши на форме строятся разноцветные окружности с центром в точке положения курсора (проект «Рисунок»).

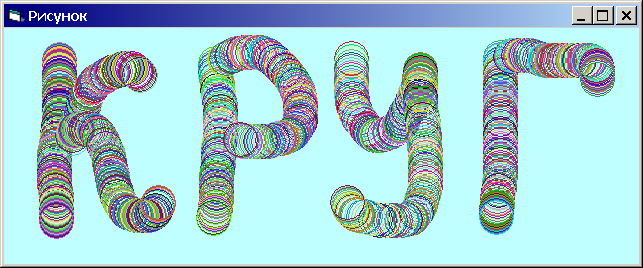


Рис. 13 Пример построения окружностей в среде VB

В процедуре перемещения указателя мыши MouseMove параметр Button дает возможность определить, какая кнопка мыши нажата (1 – левая, 2 – правая),   
X, Y – текущие координаты курсора мыши.

Программный код:

*Private Sub Form\_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)*

*If Button = 1 Then Circle (X, Y), 200, RGB(Rnd \* 255, Rnd \* 255, Rnd \* 255)*

*End Sub*

1. *Элементы управления, применяемые для работы с графикой*

Простейшими элементами управления для работы с графикой являются Line и Shape.

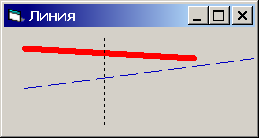
**

Рис. 13 Пример работы с элементами графики в среде VB

1. *Линия (Line):* используются для размещения на форме линий различной толщины и стилей. Некоторые свойства:

*Name:* по умолчанию Line1

*X1, Y1, X2, Y2:*координаты начальной и конечной точек

*BorderColor:* цвет линии, *BorderWidth:* толщина линии

*BorderStyle:* стиль линии (если свойство BorderWidth=1, то может принимать целые значения от 0 до 6: сплошная, пунктирная, штрих-пунктирная и другие стили).

1. *Фигура (Shape):* используются для размещения на форме геометрических фигур (прямоугольник, квадрат, овал, окружность, скругленный прямоугольник, скругленный квадрат).

Некоторые свойства:

*Name:* по умолчанию Shape1

*Shape:* задает тип фигуры (целые значения от 0 до 5)

*BorderColor:* цвет линии границы, *BorderWidth:* толщина линии границы

*BorderStyle:* стиль линии границы (если BorderWidth=1, то целые значения от 0 до 6)

*FillColor:* цвет заполнения фигуры, *FillStyle*: стиль заполнения (целые значения от 0 до 7: прозрачный, сплошная заливка, вертикальные линии и другие стили).

*Отличие линий и фигур* от других элементов управления заключается в том, что для них не обрабатываются никакие события.

*Пример*: в зависимости от заданного значения свойства Shape строятся разные типы фигур (проект «Фигуры»).

*Private Sub HScroll1\_Change()*

*Shape1.Shape = HScroll1.Value*

*Label2.Caption = HScroll1.Value*

*End Sub*

*Private Sub HScroll1\_Scroll()*

*HScroll1\_Change*

*End Sub*

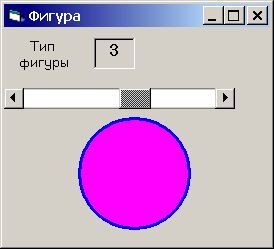
**

Рис. 14 Интерфейс графическогопроекта «Фигуры»

В Visual Basic существуют специальные элементы управления для размещения на форме готовых изображений, хранящихся в графических файлах формата JPG, GIF, BMP, ICO.

1. *Поле образа (Image)* – элемент управления, который помещает изображение в нужное место формы, может изменять его размеры и пропорции.

Некоторые свойства:

*Name:* по умолчанию Image1

*Stretch:* способ подгонки изображения

а) Если Stretch=True, то размер загружаемого изображения подгоняется под размер Image

б) Если Stretch=False, то размер Image подгоняется под размер загружаемого изображения

*Picture:* путь доступа к файлу, хранящему загружаемое в Image изображение

*Пример:* представлено одно и то же изображение, помещенное в элементы Image с разным значением свойства Stretch: в первом изображение исходного размера (Stretch = False), а в других показаны возможные эффекты при изменении размера Image (Stretch = True).

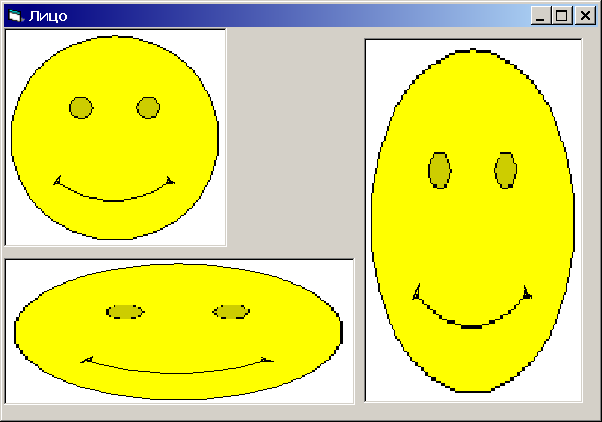
**

Рис. 15 Пример работы с элементами Image в среде VB

1. *Графическое поле (PictureBox)* – элемент управления, который помещает изображение в нужное место формы, а также имеет собственную систему координат и позволяет использовать графические методы и анимацию. В отличие от элемента Image, элемент Picture Box не может изменять исходный размер изображения.

Некоторые свойства:

*Name:* по умолчанию PictureBox1

*AutoSize:*

а) Если AutoSize =True, то размер Picture подгоняется под размер загружаемого изображения

б) Если AutoSize =False, то размер Picture не подгоняется под размер загружаемого изображения

*Picture:* путь доступа к файлу, хранящему загружаемое в Picture изображение

*AutoRedraw:* перерисовка изображения

*Пример 1*: представлено одно и то же изображение, помещенное в элементы Picture Box, у которых разные размеры и значения свойства AutoSize. В последнем случае графическое поле подгоняется по размеру изображения.

**

Рис. 16 Пример работы с элементами PictureBox в среде VB

*Пример 2:* На форму помещаем элемент Picture и командную кнопку Нарисовать, после щелчка по которой создается «картина» из 50 разноцветных прямоугольников («Авангард»).

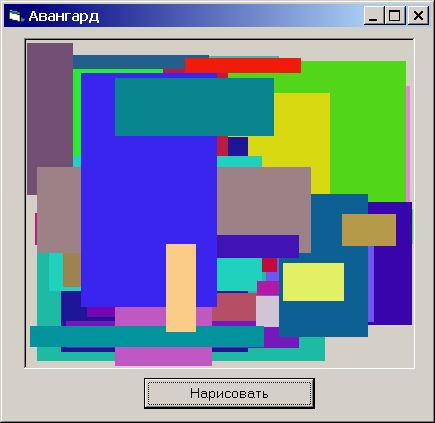


Рис. 17 Пример работы с элементами PictureBox в среде VB

*Программный код:*

*Private Sub Command1\_Click()*

*Dim i As Integer 'номер фигуры*

*Dim x1, x2, y1, y2 As Integer 'координаты фигуры*

*Randomize Timer*

*Picture1.Cls 'очищаем графическое поле*

*'строится 50 закрашенных прямоугольников*

*'координаты и цвет задаются случайным образом*

*For i = 1 To 50*

*x1 = Picture1.Width \* Rnd*

*y1 = Picture1.Height \* Rnd*

*x2 = Picture1.Width \* Rnd*

*y2 = Picture1.Height \* Rnd*

*Picture1.Line (x1, y1)-(x2, y2), RGB(Rnd \* 255, Rnd \* 255, Rnd \* 255), BF*

*Next i*

*End Sub*

1. *Использование переменных для хранения информации*
2. *Переменные: имя, тип, способы объявления*
3. Имя: латинские и русские буквы, цифры, символ подчеркивания. Начинается с буквы. Максимально 255 символов.
4. Тип:

1) Числовые:

*Byte:*целые числа от 0 до 255

*Integer:* целые числа от -32768 до 32767

*Long:* длинные целые

*Single:* вещественные одинарной точности

*Double:* вещественные двойной точности

*Currency:* денежный формат (числа с фиксированной точкой)

2) Символьные:

*String:* переменной длины

*String\*n:* фиксированной длины

3) Особые типы:

*Boolean:* логический тип; *Date:* тип дата/время

*Object:* объектный тип; *Variant:* стандартный тип (по умолчанию)

1. Объявление переменных:
2. Явное объявление (с помощью специального оператора *Dim, Private, Static, Public): Dim имя переменной [As тип переменной]*Например: Dim f As Integer
3. Неявное объявление (с помощью специального символа *%, &, !, #, @,**$*после имени переменной): S% (тип Integer), Y# (тип Double)
4. Время жизни переменных:

По времени жизни переменные делятся на:

1. Динамические: существуют, пока выполняется породившая их процедура.
2. Статические: сохраняют свои значения после окончания работы процедуры. Их необходимо объявлять: *Static имя переменной [As тип переменной].* Такие переменные часто используют для подсчета количества.
3. Зона видимости переменных:
4. Локальные:

А) локальные для процедуры: описываются внутри конкретной процедуры оператором *Dim* или *Private*. Такие переменные нельзя использовать из других процедур.

Б) локальные для формы: описываются в разделе общих объявлений General Declaration этой формы оператором *Dim*. Такие переменные можно использовать из всех процедур кода для данной формы.

1. Глобальные: описываются в разделе общих объявлений General Declaration программного модуля проекта (файл с расширением bas) оператором *Public.* Такие переменные можно использовать из всех процедур проекта.
2. Числовые переменные. Арифметические операции. Математические функции.

*Арифметические операции* (в порядке приоритета):

1) ^ – степень 2) - смена знака 3) \* / умножение, деление

4) \ – целочисленное деление 5) MOD – остаток от целочисленного деления 6) + –

* Все арифметические операции записываются в одну строку.
* В десятичных дробях целая часть отделяется от дробной точкой.
* Знаки арифметических операций опускать нельзя.



Пример:

записывается: (8.7\*x+3\*x^2)/(5\*x-9\*x^(1/5))

*Математические функции:*

ABS(x) - |x| SQR (x) - √x FIX(X), CINT(x) – округление до целого

INT(x) – наибольшее целое число, не превосходящее x

SIN(x), COS(x), TAN(x) – тригонометрические функции

RND – генерирует произвольное число, большее 0 и меньшее 1.

* Аргументы всех функций пишутся в скобках.
* Аргументы тригонометрических функций задаются в радианах:



* Функции не являются самостоятельными командами языка.

Примеры: Функция INT(9.7) возвращает 9, INT(-8.5) возвращает –9;

Функция SIN(17\*3.141592/180) возвращает значение sin(17°);

Функция INT(RND\*100)+1 возвращает произвольные целые числа в диапазоне от 1 до 100.



записывается: cos(5\*x)^3\*(tan(3\*x^2)-4.6)/(sgr(x+1)^5+7.23)

1. Символьные переменные, строковые функции. Функции преобразования типов данных.

В Visual Basic предусмотрен специальный набор средств, чтобы можно было выполнять разнообразные операции со строками.

*Объединение строк (конкатенация):* выполняется с помощью оператора & (можно также использовать +). При объединении двух строк вторая помещается в конец первой, например, в результате команды «пример » & «№1» получим строку «пример №1», а по команде «25» & «693» получим строку «25693».

Кроме указанного оператора, в Visual Basic используется большое количество встроенных строковых функций.

1. Функция Len() используется для определения длины строки. Общий вид команды: результат = Len(строка), например, функция Len(«файл») возвращает число 4.
2. Функции UCase() и LCase() используются, чтобы изменить регистр символов заданной строки на верхний или нижний соответственно.
3. Функции Left(), Right() и Mid() используются для выделения части строки. Общий вид команды: Новая строка = Left(Исходная строка, число символов), например, в результате команды Left(“самолет”, 3) получим строку «сам», а по команде Right («пароход», 3) получим строку «ход». Функция Mid() является более универсальной и часто заменяет собой Left() и Right(). Общий вид команды: Новая строка = Mid(Исходная строка, начальная позиция, число символов), например, в результате команды Mid(“информатика”, 3, 5) получим строку «форма», а по команде Mid(“подберезовик”, 4) (параметр “число символов” опущен) получим строку «березовик».
4. Функции LTrim(), RTrim() и Trim() отсекают пробелы с начала, конца или с обеих сторон строки соответственно.

В Visual Basic используются также встроенные функции преобразования типов данных.

1. Функция Str() преобразует число в строку текста. Если число положительное, то первым символом в полученной строке будет пробел. Общий вид команды: Строка, содержащая цифры = Str(число), например, в результате команды Str(895) получим строку « 895», а по команде Str (-5678) получим строку «-5678».
2. Функция Val() преобразует строку, содержащую цифры, в число, последовательно считывая символы строки, пока не встретится нецифровой знак. Общий вид команды: Число = Str(Строка), например, в результате команды Val(«45Т89») получим число 45, а по команде Val («К593») получим число 0.
3. Функция Chr() преобразует ASCII-код символа в соответствующий символ. Например, в результате команды Chr(72) получим символ H, а по команде Chr (65) получим символ А.
4. Функция Asc() выполняет обратную задачу: преобразует символ в соответствующий ему ASCII-код. Например, в результате команды Asc(«0») получим код 48, а по команде Asc («а») получим код 97.

*Пример1:* записать символы данного слова в обратном порядке («Порядок символов»).

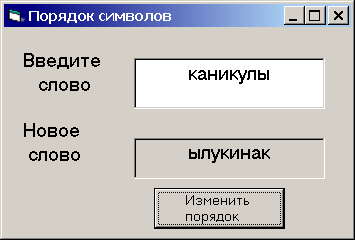
**

Рис. 18 Интерфейс проекта «Порядок символов»

*Пример2:* приложение для каждого введенного символа выводит соответствующий ему ASCII-код.

**

Рис. 19 Интерфейс проекта «Код символа»

1. *Массивы переменных*

Это наборы переменных одного типа, объединенные общим именем. В школьном курсе рассматриваются одномерные массивы (последовательности) и двумерные (таблицы). Каждый элемент массива имеет имя, тип, порядковый номер (индекс), значение.

1. Объявление массива: Dim A(10) As Single (в массиве 11 элементов вещественного типа (нумерация индексов с 0)); Dim C(1 to 20) As Integer (в массиве 20 элементов целого типа).
2. Способы заполнения массива:

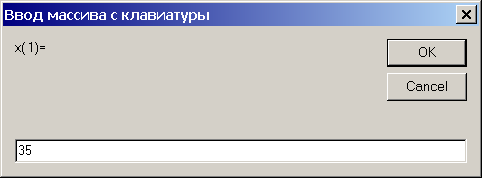


Рис. 20 Интерфейс проекта «Ввод массива с клавиатуры»

1. С помощью оператора присваивания:

*For i = 1 To n*

*x(i) = Int(Rnd \* 100)+1 ' заполнение случайными числами из отрезка [1; 100]*

Next i

1. Ввод массива с клавиатуры:

*For i = 1 To n*

*x(i) = InputBox("x(" & Str(i) & ")=", "Ввод массива с клавиатуры")*

*Next i*

1. *Основные алгоритмические конструкции*
2. *Линейные алгоритмы*

Существует большое количество алгоритмов, в которых все команды должны быть выполнены последовательно одна за другой. Такие алгоритмы называются *линейными.*

Например, рассмотренные ранее проекты «Свойства формы», «Метка», «Фигуры» и ряд других кодируются с применением линейных алгоритмов.

1. *Алгоритмы ветвления*

Часто требуется изменить естественный ход последовательного выполнения операторов программы. Тогда применяют *конструкции принятия решения (управляющие конструкции).*

1. Основой для принятия решения в управляющих конструкциях являются *условные выражения (условия).*Они возвращают одно из 2 значений: истина или ложь и используют следующие операторы сравнения: **=, >, <, > =, < =, < >** (не равно). Например, условное выражение 5\*х +2 > 0 при х = 3 возвращает значение истина, а при х = -1 возвращает значение ложь.

Существуют *сложные условия* – это последовательности простых условий, объединенные между собой знаками логических операций AND (и), OR (или), NOT (не), например: математическое неравенство –5 ≤ y ≤ 8 в программировании запишется в виде сложного условия *–5 <= y and y <= 8*.

1. *Управляющая конструкция IF … Then***:** применяется, когда необходимо выполнить один или группу операторов при соблюдении определенного условия. Вид команды:

*А) IF условие THEN действие (однострочный оператор)*

*Б) IF условие THEN*

*Действия (многострочный оператор)*

*END IF*

При истинности условия выполняются действия, указанные после слова Then, а если условие ложное, то выполняются команды, следующие за оператором IF … THEN.

Например: команда *If x > 0 then y = SQR(x)*дает возможность избежать ошибки при выполнении программы, так как корень будет извлечен только из положительной величины х.

1. *Управляющая конструкция IF … Then … Else:*применяется, когда необходимо выполнить разные действия в зависимости от соблюдения или несоблюдения условия.

Вид команды:

*А) IF условие THEN действие 1 ELSE действие 2*(однострочный оператор)

*Б) IF условие THEN*

*Действия 1*

*ELSE (многострочный оператор)*

*Действия 2*

*END IF*

Действия 1 выполняются в случае истинного условия, а действия 2 – в случае ложного.

х**2** +7, при x > 0

5 - 2x**3**, при x ≤ 0

Пример: вычислить Y=

Соответствующая команда имеет вид:

*If x > 0 Then y = x^2 + 7 Else y = 5 – 2\*x^3*

1. *Оператор ELSEIF:*применяется, когда необходимо проверить несколько условий. Условие, указанное после ElseIf, проверяется только в том случае, когда предыдущее условие ложно. Вид команды:

*IF условие 1 THEN*

*действия 1*

*ELSEIF условие 2 THEN*

Например: IF x > 0 THEN

Label1.Caption = “Значение положительное”

ELSEIF x = 0 THEN

Label1.Caption = “Значение равно 0”

ELSE

Label1.Caption = “Значение отрицательное”

END IF

*действия 2*

*. . .*

*ELSE*

*действия n*

*END IF*

1. *Оператор выбора SELECT CASE:*применяется для реализации ветвления с несколькими условиями и вариантами выбираемых действий. Вид команды:

После Case, кроме конкретного значения переменной, можно использовать:

1. Список Case 1, 5, 9, 14
2. Диапазон Case 10 TO 20
3. Условие Case Is < 0
4. Остальные случаи Case Else

*Select Case переменная*

*Case значение 1*

*команды*

*Case значение 2*

*команды*

*…*

*Case значение n*

*команды*

*End Select*

Например:

Select Case vozrast

Case 0 to 6

Print “Дошкольник”

Case 7 to 17

Print “Школьник”

…

Case Else

Print “Недопустимое значение”

End Select

С применением алгоритмов ветвления кодируются рассмотренные ранее проекты «Текст», «Прямоугольник», «Рисунок», «Счастливый брак» и другие.

1. *Циклические алгоритмы*

Такие алгоритмические конструкции используются для выполнения в программе повторяющихся действий (эти повторяющиеся действия называют *телом цикла*). Использование циклов позволяет сократить число команд в программном коде проекта, облегчает чтение и отладку программ. Рассмотрим два основных типа циклов.

1. *Цикл со счетчиком (FOR … NEXT)*

Такой цикл применяют, когда заранее известно, сколько раз должно выполняться тело цикла. В данном цикле вводится переменная-счетчик (обозначим ее I), которая определяет число повторений цикла. Для счетчика задаются значения параметров: начальное значение (N) – присваивается переменной-счетчику перед первым выполнением цикла; конечное значение (K); величина шага изменения счетчика (H) - на эту величину увеличивается значение счетчика после каждого прохода цикла. Перед каждым выполнением цикла значение счетчика сравнивается с конечным значением: если I ≤ K при H>0(I≥K при H < 0), то выполняются команды тела цикла. В противном случае цикл завершается, и управление передаются следующему после NEXT оператору.

**FOR** счетчик = нач. значение **TO** кон. значение **STEP** шаг

Тело цикла

**Next** счетчик

Ключевое слово Step можно опустить. Тогда значение шага будет равно единице.

Особенно часто циклы со счетчиком используются при работе с массивами.

Счетчик I: N, K, H

Тело цикла

Блок–схема 1 Цикл со счетчиком

Пример: вычислить сумму первых ста натуральных чисел.

Программный код:

*Sub Command1\_Click()*

*For I = 1 TO 100*

*S = S + I*

*Next I*

*Label1.Caption = S*

*End Sub*

С применением цикла со счетчиком кодируются рассмотренные ранее проекты «Авангард», «Порядок символов», «Меньше 10».

1. *Условный цикл (DO WHILE … LOOP)*

Используется, когда число повторений цикла зависит от некоторого условия.

Словесная форма записи:

Пока условие выполняется, повторять:

Действия (тело цикла)

Конец цикла.

1. Условие в начале цикла (цикл с предусловием):

**Do while** условие

операторы тела цикла

# Loop

**условие**

тело цикла

действия 1

**да**

**нет**

Блок–схема 2 Условный цикл

Условие в конце цикла (цикл с постусловием):

Do

операторы тела цикла

# Loop while условие

**условие**

тело цикла

действия 1

**да**

**нет**

Блок­–схема 4 Цикл с постусловием

Пример: на форме строятся разноцветные окружности, произвольно расположенные, разного радиуса (от 50 до 150) до тех пор, пока значение радиуса не станет равно 100. Количество таких окружностей подсчитывается и выводится на экран.

Программный код:

*Private Sub Command1\_Click()*

*Dim r As Byte*

*Dim i As Integer ‘счетчик количества построенных окружностей*

*Randomize*

*Do*

*r = Rnd \* 100 + 50*

*Circle (Rnd \* Form1.Width, Rnd \* Form1.Height), r, RGB(Rnd \* 255, Rnd \* 255, Rnd \* 255)*

*i = i + 1*

*Loop While r <> 100 ‘цикл повторяется, пока радиус отличен от 100*

*Label1.Visible = True*

*Label1.Caption = i*

*End Sub*

Иногда бывает необходимо *прервать выполнение цикла*. Тогда внутри тела цикла используют оператор *EXIT FOR* для цикла со счетчиком *(EXIT DO* для условного цикла).

Пример: с помощью цикла со счетчиком найдем порядковый номер первого положительного элемента массива.

*Sub Command1\_Click()*

*Dim Z, I as Integer*

*For I = 1 TO 100*

*IF A(I) > 0 THEN Z = I : EXIT FOR*

*Next I*

*Label1.Caption = Z*

*End Sub*

*Задачи и тестовые задания для самоконтроля по курсу   
«Объектно-ориентированное программирование.   
Язык программирования Visual Basic»*

1. **Введение в Visual Basic**
2. **Контрольные вопросы**
3. Укажите назначение среды Visual Basic.
4. Назовите фирму - разработчик языка Visual Basic. Какие еще программные продукты, созданные этой фирмой, Вы знаете?
5. Перечислите основные возможности языка Visual Basic.
6. Какой тип программирования реализован в среде Visual Basic?
7. Что такое объект? Приведите примеры объектов в среде Visual Basic и других изученных Вами программных продуктах.
8. Назовите основные характеристики объекта. Приведите конкретные примеры свойств и событий для объектов Visual Basic. Перечислите свойства, общие для всех объектов Visual Basic.
9. Запишите команду для запуска среды Visual Basic.
10. Перечислите основные элементы среды Visual Basic, объясните их назначение.
11. Что такое проект? Какие файлы могут входить в состав проекта? Как можно узнать состав и структуру данного проекта?
12. Назовите основные операции, которые можно выполнять с проектом. Запишите соответствующие команды.
13. Перечислите основные этапы разработки приложения в среде Visual Basic и охарактеризуйте каждый этап.
14. Что такое процедура? Запишите общий вид событийной процедуры.
15. **Упражнения**
16. Запустите среду Visual Basic из меню Пуск системы Windows или   
    с помощью соответствующего ярлыка на рабочем столе.
17. Создайте стандартный проект.
18. Найдите на экране все изученные элементы среды Visual Basic. Отобразите   
    на экране окно Редактора кода.
19. С помощью окна Проводник проекта изучите структуру проекта. Как называются файлы для хранения формы и самого проекта?
20. Сохраните Ваш проект (не забудьте создать отдельную папку для файлов проекта).
21. Добавьте в проект еще одну форму. Изучите структуру проекта в окне Проводник проекта. Обратите внимание на имя новой формы и файла,   
    в котором она хранится.
22. Закройте все внутренние окна, а затем снова откройте окно Конструктора форм для первой, затем для второй формы. Вызовите на экран окно свойств для второй формы.
23. Удалите из проекта вторую форму. Изучив структуру проекта, убедитесь, что операция удаления выполнена верно.
24. Уберите с экрана, а затем снова вызовите на экран Панель элементов управления.
25. Завершите работу со средой Visual Basic.
26. **Тест**
27. Основным в объектно-ориентированном программировании является понятие

а) программа в) проект

б) объект г) элемент управления

1. Укажите свойство, которое есть у всех объектов среды Visual Basic.

а) text в) name

б) picture г) caption

1. Что такое метод объекта?

а) действие, выполненное объектом

б) способ воздействия на объект

в) стандартная процедура, связанная с объектом

г) действие пользователя по отношению к объекту

1. Чтобы объект реагировал на некоторое событие, должна быть создана ...

а) ситуация, чтобы событие могло произойти

б) форма и нужный объект

в) среда существования объекта

г) процедура обработки этого события

1. С каким понятием программирования можно ассоциировать понятие «свойство объекта»?

а) подпрограмма в) переменная

б) процедура г) оператор

1. Что можно изменить у программного объекта?

а) значения свойств в) набор методов

б) набор свойств г) набор событий

1. Что такое проект?

а) программа в среде Windows

б) программа в среде Visual Basic

в) готовое приложение

г) приложение на стадии разработки

1. Команды для отображения окон и панелей инструментов сосредоточены в меню …

а) Widows в) Project

б) View г) Tools

1. Какое окно не видно на экране сразу после запуска среды Visual Basic?

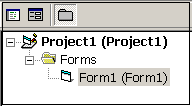
а) окно Свойства объекта в) окно Конструктор форм

б) окно Редактор кода г) окно Проводник проекта

1. Назовите внутреннее окно среды Visual Basic, фрагмент которого представлен на изображении.

а) окно Свойства объекта

б) окно Конструктор форм



в) окно Редактор кода

г) окно Проводник проекта

1. **Формы и основные элементы управления**
2. **Контрольные вопросы**
3. Для чего необходим графический интерфейс приложения? На каком этапе и как он создается?
4. Укажите назначение формы. Перечислите основные свойства, методы и события для формы.
5. Как вызвать на экран окно Редактора кода? Опишите алгоритм создания событийной процедуры в окне Редактор кода. Какие приемы автоматизации ввода команд Вы знаете?
6. Каково назначение элементов управления в Windows-приложениях? Перечислите элементы управления, наиболее часто встречающиеся   
   в Windows-приложениях.
7. Расскажите о приемах размещения на форме элементов управления, способах их выделения, перемещения, изменения размеров, удаления, выравнивания.
8. На каких этапах можно изменить свойства элементов управления? Расскажите о технологии изменения свойств этих объектов с помощью окна Свойства объекта и в окне Редактор кода.
9. Укажите назначение метки как элемента управления. Какие основные свойства, методы и события метки вы знаете?
10. Назовите основные свойства, методы и события текстового поля. Сравните объекты метка и текстовое поле с точки зрения их назначения и основных свойств.
11. Расскажите о командной кнопке как наиболее часто используемом элементе управления, о ее свойствах, событиях, методах. Какие графические свойства формы и командной кнопки можно использовать в проектах?
12. Для чего в приложениях используются флажки, переключатели, фреймы? Расскажите об алгоритме создания группы элементов управления.
13. Какие типы списков можно использовать в Windows-приложениях? Расскажите о технологии заполнения списков, добавления и удаления отдельных элементов, об очистке списка.
14. Укажите назначение полос прокрутки в диалоговых окнах в Windows-приложениях. Перечислите основные свойства для этих элементов управления, укажите главные события и когда они происходят.
15. Какие имена по умолчанию получают создаваемые на этапе проектирования интерфейса элементы управления? Какие соглашения об именах, присваиваемых разработчиком элементам управления, Вы знаете?
16. **Упражнения**

Разработайте проекты:

1. Интерфейс: форма с заголовком «Мой проект», на ней метка, текстовое поле и кнопка. На этапе выполнения после щелчка по форме ее заголовок становится «Форма», размеры 3000х2000 твипов, цвет фона – синий; в метке появляется надпись «Метка», в текстовом поле – текст «Текстовое поле», на кнопке – слово «Выход». После щелчка по кнопке «Выход» приложение завершает работу.
2. При щелчке по форме она «отскакивает» и цвет ее фона становится красным.
3. После запуска проекта при изменении с помощью мыши размеров формы выдается информация о ее новых размерах.
4. Компьютер просит пользователя ввести свое имя и здоровается с ним, обращаясь к пользователю по его имени.
5. Пользователю предлагается с помощью кнопок «Да» и «Нет» ответить   
   на некоторый вопрос. При вводе верного ответа на форму загружается изображение улыбающегося лица, при неверном – грустного.
6. Пользователь вводит пароль. После ввода текстовое поле становится недоступным, а в метку выводится сообщение, верен ли пароль.
7. Создайте электронный бланк анкеты, заполняя который пользователь вводит свои фамилию, имя, отчество, с помощью группы переключателей указывает, какое образование имеет, с помощью списка вводит свою профессию,   
   с помощью фрейма с флажками отмечает, какими владеет иностранными языками.
8. Пользователь может отформатировать текст, выбрав способ выравнивания текста, размер символов, их цвет.
9. Из списка школьных предметов пользователь выбирает наиболее интересные. Выбранные предметы помещаются в новый список в алфавитном порядке.
10. На форму помещено изображение термометра. В качестве шкалы использована вертикальная полоса прокрутки. Пользователь вводит числовое значение температуры, и оно отображается на шкале термометра.
11. **Тест**
12. Назовите основной элемент для создания графического интерфейса разрабатываемого приложения

а) метка в) форма

б) командная кнопка г) текстовое поле

2. Запишите команду, задающую синий цвет формы

а) Form1.Color=vbBlue в) Form1:Color=vbBlue

б) Form1.GrundColor=vbBlue г) Form1.BackColor=vbBlue

1. Укажите назначение команды

IF form1.caption= «Вторая» then form1.height= form1.height - 200

а) у формы с надписью «Вторая» высота уменьшается на 200

б) у формы с именем «Вторая» высота уменьшается на 200

в) у формы с надписью «Вторая» ширина уменьшается на 200

г) заголовок формы становится «Вторая» и изменяется высота

1. Для чего используют метки?

а) для создания поясняющих надписей в) для вывода информации

б) для ввода и вывода информации г) для ввода информации

1. В каком свойстве текстового поля сохраняется его содержимое?

а) Caption в) Alignment

б) Text г) PasswordChar

1. Чтобы скрыть вводимые в текстовое поле символы, используют свойство

а) Text в) Alignment

б) Font г) PasswordChar

1. Чтобы поместить на кнопку графическое изображение, надо

а) у свойства Style задать значение Standart

б) у свойства Caption задать значение Graphical

в) у свойства Style задать значение Graphical

г) у свойства Caption задать значение Picture

1. Каково назначение группы флажков?

а) обеспечивают возможность выбора 1 из предложенных вариантов

б) обеспечивают возможность выбора нескольких из предложенных вариантов

в) дают ответ на вопрос типа «Да/нет»

г) используются для решения логических задач

1. Чтобы обеспечить возможность выбора 1 из предложенных вариантов, надо:

а) поместить во фрейм флажки

б) поместить во фрейм переключатели

в) создать группу флажков

г) создать группу переключателей

1. Какое свойство хранит значение выбранного элемента списка?

а) Text в) ListCount

б) List г) ListIndex

1. В каком порядке будут размещены числа 8, 12, 5, 11, 7, 6, 9, 10 в списке с установленным свойством Sorted?

а) 5, 6, 7, 8, 9, 10, 11, 12 в) 12, 11, 10, 9, 8, 7, 6, 5

б) 10, 11, 12, 5, 6, 7, 8, 9 г) 8, 12, 5, 11, 7, 6, 9, 10

1. Как очистить список от элементов?

а) применить метод Clear в) нажать кнопку Очистка

б) применить метод RemoveItem г) нажать Ctrl+Delete

1. Какое свойство устанавливает величину перемещения бегунка при щелчке по стрелкам на краях полосы прокрутки?

а) Value в) LargeChange

б) Min г) SmallChange

1. Когда возникает событие Scroll?

а) после щелчка мыши по полосе прокрутки

б) при перемещении бегунка на полосе прокрутки

в) после перемещения бегунка в момент отпускания кнопки мыши

г) после щелчка мыши по стрелке на краях полосы прокрутки

1. Дан список элементов: метка, флажок, текстовое поле, переключатель. Запишите последовательность номеров соответствующих пиктограмм.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 1 | 2 | 3 | 4 |

а) 1, 2, 3, 4 в) 4, 2, 1, 3

б) 1, 2, 4, 3 г) 4, 3, 1, 2

1. **Основные элементы программирования**
2. **Контрольные вопросы**
3. Укажите назначение переменных в программировании.
4. Каковы правила образования имен переменных в Visual Basic? Есть ли отличия по сравнению с правилами образования имен переменных   
   в традиционном Basic?
5. Перечислите основные типы переменных Visual Basic. Приведите примеры переменных разных типов и укажите несколько их допустимых значений. Почему нельзя всем переменным присваивать тип Variant?
6. Назовите способы объявления переменных. Как подразделяются переменные   
   в зависимости от того, как были объявлены?
7. Укажите типы числовых переменных. Какие операции с ними можно выполнять? Какие математические функции вы знаете? Запишите произвольное арифметическое выражение на языке Visual Basic.
8. Укажите типы символьных переменных. Какие операции с ними можно выполнять? Какие строковые функции и функции преобразования типов данных вы знаете? Приведите конкретные примеры, запишите результат работы функции и проверьте себя, создав и выполнив проект «Строковые функции».
9. Назовите типы встроенных констант Visual Basic. Какими способами можно познакомиться с их назначением и описанием?
10. Для чего предназначены диалоговые окна? Какие типы диалоговых окон вам известны? Какими способами можно создать нужное диалоговое окно?
11. Какие объекты Visual Basic могут объединяться в массивы? Для чего применяются массивы?
12. Приведите примеры объявления массивов переменных, укажите способы их заполнения. Как создать массив элементов управления и проверить, что данные элементы управления образуют массив?
13. **Упражнения**

Разработайте проекты:

1. Пользователь вводит два числа и указывает, какую арифметическую операцию с ними выполнить.
2. Компьютер вычисляет значение дроби . Если результат вычислить невозможно,   
   приложением выдается соответствующее предупреждение.



1. Зная количество полученных в течение четверти пятерок, четверок, троек и двоек, вычислите итоговую оценку ученика, если учитель вычисляет среднее арифметическое всех оценок и округляет результат в меньшую сторону.
2. Пользователь вводит текущее время, а приложение называет соответствующее время суток.
3. Запрограммировать реакцию родителей на полученную оценку.
4. После щелчка кнопка отпрыгивает в произвольное место формы, после десятого щелчка она перестает перемещаться, и появляется надпись  
   «Я устала!».
5. Определить, сколько цифр во введенном натуральном числе.
6. Пользователь вводит свое имя, а приложение записывает его правильно: первый символ – прописная буква, остальные – строчные.
7. Пользователь вводит некоторый текст, выбирает, какую строковую функцию   
   к нему применить, задает необходимые для данной функции параметры и видит полученный результат.
8. Пользователь задает цвет фона формы, нажимая на определенную клавишу:  
   к – красный, б – белый, с – синий и т.д. Более сложный вариант: воспользоваться ASCII-кодами символов.
9. Приложение вычисляет корень введенного числа. Если число отрицательное, выводится диалоговое окно с сообщением, а содержимое текстового поля очищается.
10. Приложение запрашивает анкетные данные пользователя и выводит их   
    на форму в соответствующие метки.
11. Создайте массив их пяти командных кнопок. Изучите, какие имена получила каждая командная кнопка, и какое значение указано в ее свойстве Index.
12. Пользователь вводит 10 компьютерных терминов, а приложение располагает их в алфавитном порядке. Можно ввести ограничение на длину терминов  
    (не более 15 символов).
13. **Тест**
14. Каково значение переменной N после выполнения команд:   
    N=2: N=N^2: N=ABS(N-5\*N)?

а) 8 в) -8

б) 16 г) -16

1. Запишите команду, которая увеличивает значение переменной R в 5 раз.

а) Z=R\*5 в) R=R\*5

б) Print R\*5 г) Print R=R\*5

1. Укажите тип переменной, хранящей значение 5.7Е-6.

а) Integer в) Double

б) Long г) Single

1. Найдите переменную, имя которой задано неверно:

а) 8BX% в) Рa2W!

б) М5S& г) Usi7#

1. Где должна быть описана глобальная переменная?

а) в любом месте в) в разделе общих объявлений формы

б) внутри процедуры г) в разделе общих объявлений модуля

1. Укажите верную запись выражения на языке Visual Basic:



а) (6,79\*z-5\*z^4)/(2\*x^1/3-1) в) (6.79\*z-5\*z^4)/(2\*x^(1/3)-1)

б) (6.79z-5z^4)/(2x^(1/3)-1) г) (6.79\*z-5\*z^4)/(2\*x^1/3-1)

1. Укажите результат работы функции Mid$(“программирование”, 3, 8):

а) мир в) риммарго

б) ограммир г) рим

1. Укажите результат работы функции Str$(Val(«36H52Т»)/2):

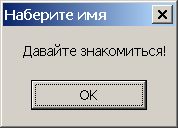
а) « 18»мир в) +18

б) «18» г) 0

1. Укажите тип информации, введенной с помощью диалогового окна ввода:

а) Integer в) String

б) Single г) String\*n

1. Для создания указанного диалогового окна надо использовать команду:

а) MsgBox "Наберите имя", , "Давайте знакомиться!"

б) MsgBox "Давайте знакомиться!", , "Наберите имя"

в) InputBox "Наберите имя", , "Давайте знакомиться!"

г) InputBox "Давайте знакомиться!", , "Наберите имя"

11. Назовите верное условие, определяющее лежит точка с координатой x отрезку [а; b]

а) x≥a and x≤b в) x>a and x<b

б) x≥a or x≤b г) x<a or x>b

1. Сформулируйте условие задачи:

INPUT A, B, C

IF A > B THEN Y=B ELSE Y=A

IF Y > C THEN Z=C ELSE Z=Y

PRINT Z

а) упорядочить числа по возрастанию в) найти максимальное из 3 чисел

б) упорядочить числа по убыванию г) найти минимальное из 3 чисел

1. Назовите значение переменной prim после выполнения фрагмента программы, если ее исходное значение 8 :

Select Case prim

Case 0 to 6

prim = prim +2

Case 7, 9, 12

prim = prim -2

Case Is<0

prim = prim \*3

Case Else

prim = prim +10

End Select

а) 10 в) 24

б) 6 г) 18

1. Назовите значение переменной K после выполнения фрагмента программы:

For X = 1 TO 5

K =K\*X

Next X

а) 0 в) 100

б) 5 г) 120

1. Что будет на экране после выполнения программы:

X=1

DO

X=X+3 : Print X+1

LOOP WHILE X<=15

а) 4, 7, 10, 13, 16 в) 5, 8, 11, 14, 17

б) 4, 7, 10, 13 г) 5, 8, 11, 14

1. Цикл – это…

а) разветвление в программе в) повторное выполнение программы

б) повторение программных строк г) последовательность строк программы

1. Какой оператор используют, если необходимо прервать выполнение условного цикла?

а) End DO в) Exit DO

б) End FOR г) Exit FOR

18. Запишите команду для объявления массива Z, хранящего 10 фамилий:

а) DIM Z (10) As String в) DIM Z (1 TO 10) As Single

б) DIM Z (1 TO 10) As String г) DIM Z# (1 TO 10)

1. Укажите назначение приведенного фрагмента программы:

For i = 1 To 100

A(i) = Int(Rnd \* 50)+1

Next i

а) просмотр элементов массива

б) каждому элементу массива присваивается значение Int(Rnd \* 50)+1

в) заполнение массива одинаковыми числами

г) заполнение массива случайными целыми числами из отрезка [1; 50]

1. Если элементы управления образуют массив, то у них:

а) одинаковое значение Name и разные значения свойства Index

б) разные имена и одинаковое значение свойства Caption

в) все свойства одинаковые

г) все свойства разные

1. **Графические возможности языка Visual Basic**
2. **Контрольные вопросы**
3. Каково назначение графики при создании современных приложений?
4. Назовите, какими способами можно применять графику в Visual Basic.
5. Расскажите, каковы особенности координатной системы в Visual Basic.
6. Какова стандартная единица измерения в системе Visual Basic и почему?
7. Что такое графические методы, для каких объектов Visual Basic они применяются?
8. Перечислите основные графические методы и приведите конкретные примеры их применения. Проверьте правильность приведенных примеров, разработав приложение, реализующее эти примеры.
9. Перечислите свойства объектов, которые могут использоваться при работе   
   с графическими методами.
10. Назовите простейшие элементы управления для работы с графикой. В чем состоит их принципиальное отличие от других элементов управления?
11. Какие элементы управления применяют в Visual Basic для размещения   
    на форме готовых изображений?
12. Расскажите о назначении элемента Image и его свойстве Stretch. Как сделать, чтобы загружаемое в Image изображение имело ширину, в два раза большую, чем в исходном файле?
13. Расскажите о назначении элемента Picture Box. Каковы его отличия   
    от элемента Image?
14. Как загрузить готовое изображение в элементы Image и Picture Box на этапе проектирования? Запишите команду для загрузки изображения во время выполнения приложения.
15. **Упражнения**

Разработайте проекты:

1. По щелчку по командной кнопке на форме строятся:

* Вертикальная пунктирная линия красного цвета.
* Прямоугольник с синей границей толщиной 5, закрашенный сплошным зеленым цветом.
* Окружность с красной штрих-пунктирной границей, закрашенная вертикальными синими линиями.
* Эллипс с границей черного цвета, закрашенный диагональными линиями фиолетового цвета.
* Сектор с зеленой границей толщиной 3, закрашенный сплошным красным цветом.
* Выводится имя разработчика приложения большими синими буквами.

1. Пользователь может выбрать, какой графический элемент будет построен: точка, линия, окружность, эллипс, прямоугольник, сектор. Местоположение, размеры и цвет построенного элемента генерируются случайным образом.
2. Построить на форме одноэтажный дом в указанном пользователем месте. Окно дома должно быть закрашено выбранным цветом.
3. Вычислить и вывести на форму таблицу 10 значений квадратичной функции.
4. Компьютер генерирует произвольным образом значение радиуса   
   (от 50 до 200), координат центра и цвета линии окружностей и строит их   
   на форме до тех пор, пока значение радиуса не станет равным 100. Число построенных окружностей вычисляется и выводится на форму.
5. Пользователь может задать тип фигуры, создаваемой элементом Shape, и стиль ее закраски. Приложение демонстрирует полученный графический образ.
6. После щелчка по кнопке «Нарисовать» в элементе Picture Box создается «картина» из 30 разноцветных прямоугольников произвольных размеров.
7. Пользователь может просматривать готовое графическое изображение,   
   изменяя его пропорции.
8. **Тест**
9. Где располагается центр координатной системы в Visual Basic?

а) в левом верхнем углу формы в) в левом нижнем углу формы

б) в правом нижнем углу формы г) в центре формы

1. Стандартной единицей измерения в системе Visual Basic является

а) миллиметр в) твип

б) пиксель г) пункт

1. Для какого из указанных объектов Visual Basic нельзя применять графические методы?

а) форма в) принтер

б) экран г) графическое поле

1. Каково назначение метода Point?

а) задает координаты точки в) указывает координаты текущей точки

б) задает цвет точки г) возвращает значение цвета точки

1. Команда Line(200, 300) – (200, 800), vbRed строит

а) линию красного цвета

б) горизонтальную линию красного цвета

в) вертикальную линию красного цвета

г) прямоугольник красного цвета

1. Если в графическом методе цвет явно не определен, то для построения используется цвет, указанный в свойстве … текущего объекта.

а) ForeColor в) FillColor

б) BackColor г) Color

1. СвойствоDrawWidth задает

а) тип линий в) ширину графического элемента

б) стиль заполнения контура г) размер точки и толщину линий

1. В чем основное отличие элементов Line и Shape от других элементов управления?

а) имеют мало свойств в) редко используются

б) не реагируют на события г) очень простые элементы

1. Какой элемент позволяет изменять размеры и пропорции готового графического изображения?

а) Form в) Picture Box

б) Shape г) Image

1. Какая функция позволяет загружать изображения во время выполнения приложения?

а) Picture в) LoadPicture

б) Load г) LoadImage

1. **Создание проектов в среде Visual Basic**

Учащимся предлагается самостоятельно разработать проекты в среде Visual Basic. При этом каждая задача имеет два уровня сложности.

1. Светофор
2. При щелчке по форме циклически изменяется ее цвет: красный, желтый, зеленый, и процесс повторяется.
3. Сделать работу светофора более реалистичной: после зеленого цвета должен вновь загореться желтый.
4. Приветствие
5. Компьютер предлагает пользователю ввести свое имя и приветствует человека, обращаясь к нему по имени.
6. Дополнительно предусмотреть в проекте проверку, ввел ли пользователь имя, и вывод повторной просьбы назвать себя, если пользователь не захотел представиться или ошибся при вводе данных
7. Форматирование
8. Пользователь может выбрать способ выравнивания текста: по левому краю, по правому краю, по центру, а также установить крупный или мелкий шрифт. Результат форматирования можно наблюдать на предложенном образце текста.
9. Дополнительно можно выбрать цвет шрифта и его начертание: обычный, полужирный, курсив, подчеркнутый и их сочетания.
10. Калькулятор
11. Имитировать работу обычного калькулятора, умеющего выполнять четыре арифметические операции: сложение, вычитание, умножение, деление.
12. Имитировать работу инженерного калькулятора, который, кроме указанных арифметических операций, может вычислять значения арифметического корня, модуля, квадрата и обратной величины введенного числа, а также значения тригонометрических функций.
13. Мороженое
14. Имитировать процесс покупки мороженого одного сорта. Покупатель может купить несколько порций, а роли электронного продавца – работающее приложение.
15. Дополнительно покупателю предлагается мороженое нескольких сортов, и он может купить несколько порций выбранного сорта.
16. Таблица умножения
17. Учащемуся последовательно предлагаются произвольные примеры таблицы умножения, и после выполнения десятого задания выставляется оценка по пятибалльной шкале.
18. Дополнительно перед началом теста запрашивается фамилия учащегося, а по окончании тестирования выдается итоговая информация: фамилия, число верных ответов, оценка. Примеры, в которых были допущены ошибки, выводятся на экран с указанием правильных ответов.
19. Число
20. Компьютер определяет количество цифр во введенном целом числе.
21. Дополнительно: подсчитывается сумма цифр в этом числе.
22. Просмотр рисунка
23. Загруженный рисунок можно просматривать в разных масштабах, например, в натуральную величину, в уменьшенном и увеличенном виде.
24. Дополнительно: пользователь может из предложенного списка выбирать рисунки для просмотра.
25. Логическая схема
26. Имитировать работу логической схемы, состоящей из 4 - 6 логических элементов И, ИЛИ, НЕ с несколькими входными сигналами и одним выходным сигналом. Пользователь вводит значения входных сигналов схемы и получает значение выходного сигнала.
27. Дополнительно: пользователь сам вычисляет значение выходного сигнала схемы, вводит его (0 или 1) и получает сообщение, правильно ли решена задача.
28. Алфавит
29. Пользователь вводит несколько фамилий, а приложение располагает их в алфавитном порядке.
30. Дополнительно: упорядочить фамилии по алфавиту (достаточно по первой букве), не используя в проекте элемент управления Список.
31. Графический редактор
32. Пользователь выбирает тип фигуры: точка, линия, прямоугольник, окружность, дуга, сектор. И в графическом поле строится соответствующая фигура, у которой размер, расположение и цвет задаются случайным образом.
33. Дополнительно: пользователь может задать толщину линии, ее цвет.
34. Электронное расписание уроков
35. В списке выбирается день недели, и если это не выходной, на экран выдается соответствующее расписание уроков.

Дополнительно: пользователь, выбрав в расписании конкретный предмет, получит информацию о фамилии преподавателя и номере кабинета.

*Методические рекомендации для учителей*

1. Наиболее целесообразно набирать в группу по освоению данного курса ребят, имеющих некоторый опорный уровень знаний языка программирования Basic (назначение и типы переменных, основные операторы, базовые алгоритмические конструкции и их реализация в Basic), и практические навыки работы с Windows-приложениями. Тогда учащиеся смогут в большей степени самостоятельно продвигаться в процессе освоения курса.
2. В тематическом планировании курса преподавателем не обязательно строго придерживаться последовательности изложения теоретического материала по модульному принципу, как они указаны в программе курса. Например, если учащиеся в группе достаточно хорошо знают основные алгоритмические конструкции языка Basic, то уже в начале изучения тем второго модуля, актуализовав знания учащихся, целесообразно решать задачи с применением алгоритмов ветвления и циклов. Изучение флажков, переключателей, списков лучше проводить параллельно или после изучения массивов элементов управления. Изучая материал второго и третьего модуля, целесообразно решать задачи с применением графических методов, что, безусловно, повысит интерес учащихся к программированию. В среде Visual Basic можно предлагать учащимся уже ставшие традиционными графические задачи: «Неоновая реклама», «Мыльные пузыри», «Конфетти», «Падающий снег», «Змейка», построение орнаментов, движение геометрических фигур по разным траекториям и другие, закрепляя с их помощью знания основных алгоритмических конструкций (ветвлений, циклов) и операторов, умение использовать массивы и базовые функции языка Basic. В то же время Visual Basic предоставляет новые возможности при работе с графикой, и учащиеся станут с удовольствием их исследовать и применять.
3. Для проведения теоретических занятий учитель может использовать любой из многочисленных учебников по курсу Visual Basic (см. список рекомендуемой литературы в программе курса).
4. Целесообразно в начале каждого семестра (полугодия) предложить учащимся теоретический материал изучаемого курса (материал для учащихся, прилагается к работе), который они могут изучать как самостоятельно, так и под руководством учителя, в зависимости от их начального уровня знаний технологии программирования и конструкций языка.
5. Для организации эффективной самостоятельной работы желательно перед изучением новой темы (модуля) предлагать учащимся материал для самоконтроля: контрольные вопросы, которые помогут выделить главное   
   во вновь изучаемом материале, упражнения для самостоятельного выполнения.
6. Роль учителя во время занятия состоит в объяснении нового материала и постановке задачи, а затем консультировании учащихся в процессе выполнения практического задания. Преподаватель также осуществляет различные формы контроля качества освоения программы, проверяет и оценивает созданные учащимися проекты. Итоговый тематический контроль осуществляется учителем по результатам выполнения учащимися практических заданий и/или тестирования.
7. На первых занятиях надо познакомить учащихся с требованиями, предъявляемыми к интерфейсу и программному коду создаваемых проектов, добиваться их соблюдения в период всего этапа изучения курса, воспитывая аккуратность, внимательность, трудолюбие, ответственность за созданный программный продукт.
8. Учителю необходимо иметь «коллекцию» разноуровневых задач для изучения каждой темы курса. Основой проектной деятельности данного курса является методическое пособие «Visual Basic в задачах» Бурденко Е.Э. Кроме того, коллекцию задач можно пополнить из специальной литературы по VISUAL BASIC (см. библиографический список), а также изучая примеры справочной системы, встроенной в среду программирования. Так как основной материал III и IV модулей унаследован из традиционного Бейсика, для проведения занятий соответствующих тем можно привлекать практический материал курса «Программирование на языке Basic». При этом традиционные задачи классического программирования, благодаря использованию визуальных средств системы, получают «второе дыхание», становятся более интересными, а часто и более простыми с точки зрения программирования.
9. Необходим дифференцированный подход, выражающийся в разном уровне предлагаемых практических заданий, степени помощи отдельным ребятам. Важно учитывать разный уровень способностей, базовых исходных знаний и разную мотивацию каждого ученика, и большой запас задач необходим, чтобы «слабому» предложить посильное задание, заинтересовать «лентяя», «сильному» дать возможность проявить себя, проверить свои способности и уровень знаний, посоревноваться с другими. Даже последовательность и глубина изучения материала во многом может зависеть от потребности самих учащихся.
10. Развивая в учащихся стремление к творчеству, необходимо рассматривать и обсуждать разные варианты решения одной и той же задачи, отмечать наиболее красивые способы решения. Надо поощрять желание ребят усложнять сценарий рассматриваемой задачи и самостоятельно искать теоретический материал для ее решения.

*Требования к работе учащихся*

1. При создании интерфейса проекта желательно:

* использовать гармонично сочетающееся цветовое оформление формы и элементов управления (поощряется использование готовых или самостоятельно созданных графических образов);
* при размещении текстовой информации устанавливать соответствующее выравнивание и выбирать подходящий размер шрифта, использовать поясняющие надписи, не допускать в тексте орфографических ошибок;
* выбирать элементы управления, наиболее удобные в данном проекте для взаимодействия с пользователем, способствующие применению более рационального алгоритма решения задачи и сокращению числа команд в программном коде; не перегружать форму элементами управления.

1. В программном коде проекта:

* необходимо включать комментарий с информацией о назначении переменных, процедур, с указанием того, что происходит в циклах и др.;
* желательно использовать наиболее рациональные алгоритмические конструкции языка;
* предусматривать реакцию приложения на случай ввода пользователем недопустимых исходных данных, чтобы работа приложения не завершалась аварийно;
* поощряется разработка оригинальных и компактных алгоритмов решения задачи, диалоговый характер работы приложения, создание дружественного для пользователя интерфейса.

Ответы к тестам

Ответы к тесту модуля I

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| НОМЕР | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| КЛЮЧ | б | в | в | г | в | а | г | а | б | г |

Ответы к тесту модуля II

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| НОМЕР | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| КЛЮЧ | в | г | а | в | б | г | в | б | г | а |
| НОМЕР | 11 | 12 | 13 | 14 | 15 |  |  |  |  |  |
| КЛЮЧ | б | а | г | б | в |  |  |  |  |  |

Ответы к тесту модуля III

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| НОМЕР | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| КЛЮЧ | б | в | г | а | г | в | б | а | в | б |
| НОМЕР | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| КЛЮЧ | а | г | г | а | в | б | в | б | г | а |

Ответы к тесту модуля IV

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| НОМЕР | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| КЛЮЧ | а | в | б | г | в | а | г | б | г | в |

*Рекомендации по выполнению заданий модуля V*

С помощью практических заданий модуля V осуществляется итоговый контроль качества освоения программы курса. В ходе решения этих задач происходит комплексное творческое применение учащимися полученных знаний, навыков, умений, их углубление и закрепление.

Задания должны быть предоставлены учащимся в начале каждого семестра. Предполагается их самостоятельное выполнение, а также возможность обратиться к преподавателю за консультацией. В конце семестра созданные проекты предоставляются для проверки. Они оцениваются, отмечаются положительные и отрицательные моменты и даются необходимые рекомендации.