НАРОДНАЯ УКРАИНСКАЯ АКАДЕМИЯ

СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ ОСНОВЫ ПРОГРАММИРОВАНИЯ В СРЕДЕ MS OFFICE

Методические рекомендации
для студентов, обучающихся по направлению подготовки
051–Экономика

Издательство НУА

НАРОДНАЯ УКРАИНСКАЯ АКАДЕМИЯ

СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ ОСНОВЫ ПРОГРАММИРОВАНИЯ В СРЕДЕ MS OFFICE

Методические рекомендации
для студентов, обучающихся по направлению подготовки
051–Экономика

Харьков
Издательство НУА
2018

УДК 004.4:33(072+075.8)
ББК 32.973.26-018.2р30
С56

Утверждено на заседании кафедры
информационных технологий и математики
Протокол № 10 от 7. 05. 2018

А в т о р-с о с т а в и т е л ь*: С. Б. Данилевич*
Рецензент*:* канд. техн. наук, доц. *К. С. Барашев*

Основная цель методических рекомендаций – оказать помощь студентам в рациональном распределении усилий в процессе самостоятельной работы, выделении главного в дисциплине и сосредоточении на нем своего внимания. Методические рекомендации помогут студентам в работе над программным материалом.

С56 **Современные информационные** технологии в экономике. Основы программирования в среде MS Office : метод. рекомендации для студентов, обучающихся по направлению подготовки 051–Экономика / Нар. укр. акад., [авт.-сост. С. Б. Данилевич]. – Харьков, 2018.– 32 с.

Основна мета методичних рекомендацій – надати допомогу студентам в раціональному розподілі зусиль у процесі самостійної роботи, виділення головного в дисципліні і зосередженні на ньому своєї уваги. Методичні рекомендації допоможуть студентам в роботі над програмним матеріалом.

УДК 004.4:33(072+075.8)
ББК 32.973.26-018.2р30

ВВЕДЕНИЕ

В современном мире необходимо обновлять свои знания и компетенции на протяжении всей жизни, чтобы обеспечить как профессиональную, так и социальную адаптацию в обществе. Одним из основных способов приобретения знаний является самообразование. Самостоятельная работа студентов над программным материалом становится ведущей формой организации учебного процесса. Ни одно учреждение высшего образования не может дать своим выпускникам такого запаса знаний, которого хватило бы им на весь последующий период трудовой деятельности. Поэтому овладение методикой самостоятельного добывания знаний позволит студентам и после выпуска из академии непрерывно повышать уровень своей подготовки, осваивать новые информационные технологии, систематически пополнять свои знания, совершенствоваться в профессиональной деятельности.

Главная проблема организации самостоятельной работы состоит в рациональном использовании ограниченного времени для глубокого усвоения программного материала. Как показывает опыт, для хорошей организации самостоятельной работы нужны определенные навыки: умение сосредоточенно и внимательно работать, развивая память; умение работать с информацией; способность вести наблюдения и записи; владение рациональными способами умственных действий и т. д. Процесс формирования таких навыков начинается в школе и продолжается в течение всего периода обучения. Для успешной организации самообразования следует разработать основные компоненты учебных стратегий: составить долговременные цели (планы, программы), определить критерии достижения учебных целей, овладеть технологиями (способами, приемами, методами), с помощью которых можно достичь учебных целей, получить доступ к ресурсам обеспечивающим достижение учебных целей и сознательно управлять своей учебной деятельностью.

1. РОЛЬ ДИСЦИПЛИН ИНФОРМАТИКИ В ПОДГОТОВКЕ СПЕЦИАЛИСТОВ ПО ЭКОНОМИКЕ

Появление электронных вычислительных машин и развитие новых информационных технологий являются основой современного этапа научно-технического прогресса.

Этот этап характеризуется интенсивным развитием новых структур ПК, информационных технологий, постоянным совершенствованием программных средств для всех областей знаний. Завершается проработка проектов создания ГК нового, пятого поколения, обладающих искусственным интеллектом которые, помимо высокой производительности и надежности, при более низкой стоимости, будут обладать следующими качественно новыми свойствами: возможностью взаимодействия с ПК при помощи человеческой речи и графических изображений; способностью системы постоянно обучаться, делать логические суждения, производить ассоциативную интеллектуальную обработку информации; способностью работать с комплексными базами знаний для решения прикладных задач.

Огромную роль в подготовке современного специалиста играет знание основ программирования.

Во-первых, знание программирования позволяет быстрее и более глубоко понять принципы построения и функционирования современных ПК и их программного обеспечения.

Во-вторых, современный специалист должен уметь использовать в своей практической деятельности знания современных информационных технологий и основ программирования.

2. РЕКОМЕНДАЦИИ ПО САМОСТОЯТЕЛЬНОМУ ИЗУЧЕНИЮ ИНФОРМАТИКИ

2.1. Планирование и организация самостоятельной работы

Обучение в высшем учебном заведении все больше основывается на творческой активности и высокой самостоятельности студентов. Особенно это характерно для изучения информационных технологий, когда сложность изучаемого материала постоянно возрастает, а время, отводимое учебным планом на его изучение, уменьшается. В этих условиях индивидуальный поиск знаний является самой характерной чертой работы студентов.

Под самостоятельной работой понимается деятельность студентов, как по заданию, так и по собственному желанию, направленная на закрепление, расширение и углубление получаемых знаний, умений и навыков, а также на усвоение нового материала без посторонней помощи.

Самостоятельная работа, как и любая другая, обязательно требует научной организации труда. Целью такой организации является достижение максимального эффекта от учебного труда студентов при условии минимальных затрат времени и человеческой энергии. Здесь студентам нужна квалифицированная помощь, направленная на то, чтобы они могли правильно организовать и спланировать свою работу.

Планирование – главная предпосылка правильной организации умственного труда. Чтобы работать ритмично, без перегрузок, необходим личный план. Следует всегда помнить: чем больше занят человек, тем тщательнее приходится планировать время и работу. Исходными материалами для составления личных планов служат расписание занятий и графики обязательных заданий (индивидуальных заданий, курсовых проектов, контрольных работ и т. п.).

Наряду с планированием, важное значение имеет техника личной работы. Как техника, так и организация личной работы студента должны отвечать основной идее научной организации труда – максимальный эффект при минимальной затрате времени.

К большим потерям времени приводит отсутствие порядка в работе и хранении необходимых материалов. Если под рукой не оказывается нужного конспекта, учебного пособия, работа откладывается, растут непроизводительные затраты времени. Создание нормальных условий труда определяется самими студентами.

Вначале необходимо осмыслить свой план самостоятельных занятий, т. е. продумать весь технологический процесс действий по усвоению намеченного к проработке учебного материала. Затем следует подготовить учебную литературу, конспекты, ручки, карандаши и т. п. Все эти предметы должны быть разложены в определенном порядке, чтобы не расходовать время на их поиск. Ведь производительность умственного труда в значительной степени зависит от порядка на рабочем месте.

2.2. Ведение и отработка конспекта лекций

Прорабатывая лекцию, необходимо прочитать свои записи, учебник и другую литературу. Делать это следует в такой последовательности:

1. прочитать конспект, сосредоточив основное внимание на понимании физической сущности рассматриваемого вопроса, установлении причинно-следственной связи с другими положениями, явлениями, событиями, не задерживаясь на математических и логических выкладках, пояснениях;

2. если не удалось достичь ясного понимания вопроса по конспекту, необходимо обратиться к учебнику и выяснить суть вопроса;

3. после того, как достигнуто понимание сути материала лекции, перейти к разбору математических выкладок и обоснований.

При отработке отдельных вопросов лекции необходимо по мере усвоения порции учебной информации вносить в конспект дополнения, уточнения, пояснения и т. п.

Необходимо разбить учебный материал лекции на смысловые отрезки, дать им легко запоминающиеся заглавия, выделить и подчеркнуть цветным карандашом главные мысли.

Такая отработка лекции как бы иллюстрирует материал. Это повышает умственную активность обучаемого, способствует лучшему усвоению материала и, конечно, впоследствии облегчит труд при подготовке к групповым, практическим и лабораторным занятиям.

2.3. Практические занятия

Практические занятия по информационным технологиям и программированию, как правило, проводятся в учебных группах и имеют целью изучение и практическое освоение новых информационных технологий и программирования, приобретение устойчивых навыков работы на ПК, умение готовить программы к вводу в ПК, читать и отлаживать программы, использовать полученные на ПК результаты. Одним из условий качественного проведения практических занятий является самостоятельная работа накануне, самостоятельность и инициатива на занятии. Достижение указанных целей будет более полным, если каждый студент (обучаемый) самостоятельно переработает учебный материал и придет на занятие хорошо подготовленным.

Приемов самостоятельного изучения техники программирования и новых информационных технологий существует много.

Однако как бы студенты ни занимались самостоятельно по соответствующим учебникам или учебным пособиям, постичь все тонкости современных информационных технологий и техники программирования без практики работы с ними невозможно.

В настоящее время в связи с усложнением информационных технологий и специальных программных продуктов все более возрастает количество отдельных приемов, действий и операций в целом, которые специалист должен знать в совершенстве и выполнять в кратчайшие сроки. Все это достигается только практикой. Вот почему в процессе самостоятельной работы следует особое внимание обращать на вопросы, связанные с эксплуатацией программных

продуктов на ПК, с программированием на языках высокого уровня и, в частности, на VBA .

3. ИЗУЧЕНИЕ АЛГОРИТМИЧЕСКОГО ЯЗЫКА И ПРОГРАММИРОВАНИЯ НА VBA

3.1. Последовательность изучения программирования

Главной целью изучения офисного программирования является получение совокупности знаний, позволяющих разрабатывать или модернизировать офисные программы для решения задач на ПК. Основу обучения офисному программированию составляет самостоятельная практическая работа по написанию, отладке и проверке программ решения задач различной сложности в основных приложениях Microsoft Office.

Так же, как нельзя научиться плавать, не заходя в воду, нельзя научиться программировать, не составив ни одной программы и не решив ни одной задачи на ПК. Это положение является фундаментом правильной организации самостоятельной работы по изучению программирования на VBA.

Опыт показывает, что изучение программирования целесообразно организовать в строго определенной последовательности.

В первую очередь необходимо уяснить этапы решения задачи на ПК. Основными этапами решения задачи на ПК являются следующие: постановка и формализация задачи; разработка алгоритма; составление программ; отладка программы.

Основной задачей первого этапа является уяснение содержания задачи, налагаемых ограничений и осуществление перехода от описательной постановки к математическому представлению.

Этап разработки алгоритмов является важнейшим этапом при подготовке задачи к решению на ПК. Начинать изучение данного этапа следует с глубокого уяснения понятия алгоритма, свойств, способов представления и видов алгоритмов. Здесь же надо изучить правила изображения схем алгоритмов.

В завершение этапа следует обязательно провести проверку правильности разработки алгоритма.

Таблица 1.

Последовательность изучения программирования

Что учить?	Как учить?
Что такое программирование? Составление программ для решения задач на ПК.	Разрабатывать алгоритмы и программы решений прикладных задач на ПК.
Как решать задачу на ПК? Этапы решения задачи на ПК.	Изучить содержание основных этапов решения задачи на ПК

Что такое алгоритм? Понятие алгоритма. Свойства, способы представления и виды алгоритмов.	Изучить ГОСТы, разрабатывать типовые алгоритмы.
На каких языках пишутся программы? Понятие языка программирования. Классификация языков программирования.	Ознакомиться с классификацией языков программирования, уяснить их общие черты и характерные особенности.
Как написать программу на объектно-ориентированном языке? Объектно-ориентированный язык.	Ознакомиться с алфавитом и операторами языка. Изучить структуру программы на объектно-ориентированном языке. Написать простейшую программу
Как лучше построить алгоритм и программу для решения сложной задачи? Нисходящее, модульное и структурное программирование.	Освоить приемы построения структурированных алгоритмов и программ.
Каким образом программы обрабатываются на ПК? Система программирования.	Изучить назначение, состав и принципы функционирования систем программирования.
Хорошо ли написана программа? Понятие качества программы. Оценка качества программы.	Уяснить основные показатели качества программ и их физический смысл. Уметь рассчитывать некоторые из них

Приступая к изучению этапа составления программы, следует выяснить два главных вопроса:

1) на каком языке должен быть записан алгоритм, чтобы его мог реализовать в среде Microsoft Office ПК?

2) какие средства языка наиболее удобны для представления конкретного алгоритма?

Необходимо помнить, что машина может выполнить только такую программу, которая представлена на языке ее команд. Однако для облегчения процесса составления программы применяются проблемно-ориентированные и объектно-ориентированные языки. Следует заметить, что наиболее трудным в программировании является освоение языка и приемов программирования на нем. Именно поэтому особенностям изучения языка программирования посвящены отдельные подразделы данного пособия.

Составленная на любом языке программа, как правило, содержит ошибки. Устранение допущенных в процессе алгоритмизации и программирования ошибок и получение правильных результатов счета является целью этапа отладки программ. По времени отладка программы примерно в 4 раза превышает длительность любого из рассмотренных этапов. Теоретически учиться отлаживать программы не имеет никакого смысла. Только практическая работа на ПК дает возможность усвоить методику отладки программ, записанных на том или ином языке программирования.

В заключение следует изучить вопросы, связанные с оптимальными методами разработки больших программ, с автоматизацией их создания и определения их качества.

Здесь основным является:

технология современного программирования;

функционирование систем программирования при обработке программы пользователя;

сложность программы;

трудоемкость разработки программы.

Изучение технологии программирования следует начинать с определения этого понятия, затем ознакомиться с основными методами программирования (нисходящее, модульное, структурное), уяснить их характерные особенности и диалектическое единство.

Основу *нисходящего программирования* составляет описание решения задачи по принципу «сверху вниз», т. е. сначала формулируется главная цель разработки (самый старший иерархический уровень), затем определяются второстепенные (подчиненные) цели, связанные иерархической зависимостью. В результате разработчик уже на начальном этапе определяет основные идеи и структуру сложной программы.

Модульное программирование предполагает разбиение программы на части-модули, которые можно независимо друг от друга программировать, компилировать (транслировать), отлаживать и корректировать. В основе модульного программирования лежит древний способ управления сложной ситуацией: «Разделяй и властвуй!».

Структурное программирование определяет принципы конструирования VBA-проектов, программ (модулей), позволяющих упростить их логику, улучшить понимаемость, облегчить отладку. В основе структурного программирования лежит склонность человека к последовательным действиям.

3.2. Методика изучения объектно-ориентированного языка

Приложения Microsoft Office предназначены, прежде всего, для создания сложных электронных документов, включающих базы данных Access, диаграммы и расчеты в таблицах Excel, которые можно выводить на печать, представлять в виде презентации Power Point т. п. Мощные средства вычисления, анализа и прогнозирования Microsoft Office позволяют решать практически любые задачи, возникающие при ведении бизнеса на любом предприятии. Высокий эффект взаимодействия компонентов Microsoft Office достигается организацией программ на макроязыке VBA, позволяющим настроить систему на решение как общих, так и специфических для каждой организации задач.

Современный специалист должен использовать в своей работе новейшие достижения информационных технологий, в том числе и иметь представление о языке VBA, так как приложения Microsoft Office сейчас применяются почти в любой области человеческой деятельности. С помощью этого языка реализуется визуальный стиль программирования, отличающийся тем, что программы проектируются в основном из уже созданных профессиональными программистами унифицированных средств. Овладеть такой средой программирования может каждый. Кроме того, специалисту в какой-то предметной области легче освоить программирование, чем заставить разработчика-программиста овладеть тонкостями прикладной темы.

VBA – это одно из встроенных инструментальных средств приложений Microsoft Office. Программы на VBA выполняются только в среде документов Microsoft Office. Использование макросов, пользовательских функций и форм, созданных с применением VBA, не заменяет, а дополняет набор средств Microsoft Office для создания документов.

Достоинство VBA заключается в использовании универсального языка VB-технологий для решения разных задач.

С точки зрения программирования язык VBA сходен с языками Visual Basic (VB) и VBScript. Владение VB-технологией позволяет заниматься офисным программированием на VBA, создавать приложения при помощи пакета Visual Basic и Web-сценарии на языке VBScript.

В основе методики изучения объектно-ориентированного языка Visual Basic for Applications лежат следующие принципы:

последовательность и направленность;

единство изучения конструкций языка и приемов алгоритмизации задач в соответствии с правилом: «Тип алгоритма – средство языка»;

тесное сочетание программирования на VBA с практикой решения задач на ПК.

Последовательность изучения объектно-ориентированного языка Visual Basic for Applications приведена в табл. 2. Рассмотрим основные этапы изучения языка.

В первую очередь необходимо уяснить общую структуру языка и иерархическую связь его основных элементов. Затем в соответствии с иерархией (снизу вверх) изучаются все средства, необходимые для составления простейших программ.

Начинается изучение языка с его алфавита. После этого необходимо изучить правила записи чисел, переменных и функций. Здесь же следует уяснить необходимость и правила объявления данных в программе. Особенно надо обратить внимание на объявление и представление массивов.

Далее следует уяснить простейшую структуру программы. При этом необходимо изучить следующие моменты:

правила оформления программы;

классификацию, назначение и порядок выполнения основных операторов;

особенности и порядок организации ввода и вывода.

После изучения перечисленных вопросов необходимо попрактиковаться в составлении простейших линейных, ветвящихся и циклических программ.

Наиболее сложным является программирование циклических вычислительных процессов. Здесь необходимо изучить операторы цикла. При этом основное внимание следует уделить структуре (формату) и порядку выполнения операторов цикла, в зависимости от вида заголовка цикла. Далее осваивается методика составления циклических программ по принципу «от простого к сложному». Сначала изучаются приемы программирования процессов, содержащих один цикл. Затем программируются циклы со сложной структурой: цикл в цикле (вложенные циклы), циклы с разветвлением, итерационныe циклы и т. д.

После изучения простейших по структуре программ переходят к программам, содержащим процедуры (блоки). Известно, что механизм процедур увеличивает достоинства языка. Появляется возможность более сжатой формы представления алгоритма на языке программирования.

Изучение процедур надо начинать с правил их оформления, принятых в языке. Необходимо конкретно разобраться, в чем состоят отличия процедуры-подпрограммы от процедуры-функции, уяснить способы обращения к процедурам.

Завершается изучение языка выходом на машину для решения задач. Этот практический этап должен быть направлен на освоение тех возможностей, которые представляют операционная система и система программирования программисту по организации решения задачи на ПК.

Основными вопросами при этом являются следующие:

особенности организацин диалога между программистом (пользователем) и ПК (команды, сообщения);

расшифровка сведений об ошибках, допущенных в программе, их локализация и устранение.

В заключение необходимо отметить, что процесс программирования является творческим процессом. Однако в нем встречаются повторяющиеся элементы, формальные приемы, стандартные подходы. Их нужно накапливать, что в конечном итоге поможет накопить определенный опыт, повысить квалификацию, сократить время на разработку новых программ и уменьшить количество ошибок.

Таблица 2.

Последовательность изучения объектно-ориентированного языка

Что изучать?	**Как изучать?**
Что такое VBA? Структура языка.	Ознакомиться.
Что такое алфавит языка? Набор букв, цифр и специальных символов.	Запомнить в первую очередь специальные символы
Как представляются данные? Константы, переменные. Объявление (описание) данных.	Изучить.
Что такое программа? Структура и порядок оформления программы.	Запомнить.
Как составить линейную программу? Выражения, функции. Оператор при сваивания. Операторы ввода и вывода. Методика программирования.	Структура линейного алгоритма.
Как составить разветвляющуюся программу? Логические выражения и выражения типа отношения (сравнения). Условный оператор. Особенности программирования ветвлений.	Программировать алгоритмы с разветвляющейся структурой.

Как составить циклическую программу? Оператор цикла. Особенности программирования циклов.	Программировать алгоритмы с циклической структурой.
Как организовать выполнение про граммы на ПК? Особенности ввода, редактирования, отладки и выполнения программ.	Решать конкретные задачи на ПК.

3.3. Особенности изучения алгоритмического языка VBA

Успех в современном бизнесе и менеджменте во многом опирается на оперативный анализ экономической ситуации и выбор оптимального решения из множества альтернатив. Для решения этих задач широко используются уже хорошо известные Вам компьютерные средства, такие как Micrsoft Word, Exce1, Access. Однако Вы на младших курсах изучали только порядок использования этих средств в своей работе, не выясняя, каким образом те или иные действия реализуются программой. На практике очень часто возникают ситуации, когда Вашу конкретную задачу Вы не сможете решить стандартными средствами указанных приложений. Возникает необходимость создать свое приложение. Для этого используются специальные средства тех же Microsoft Exce1, Access, Word. Одним из широко используемых в настоящее время для создания Windows приложений является язык высокого уровня Visual Basic for Applications. Этот язык позволяет достаточно легко решать многие задачи, о возможности решения которых стандартными средствами не может быть и речи. Язык программирования VBA с точки зрения изучения является наиболее простым из всех объектно-ориентированных языков программирования высокого уровня. В основном этот язык применяется для решения вычислительных задач и модернизации офиснъпс программ приложений Microsoft Office. Порядок изучения языка VBA представлен в табл. 2.

Приложения, созданные с помощью VBA, работают по событийному принципу.

Слово «Visual» означает, что с помощью этого языка реализуется визуальный стиль программирования. Это стиль, отличающийся от более ранних тем, что программы не пишутся, а проектируются из уже созданных профессиональными программистами унифицированных средств.

Слово «BASIC» означает тот факт, что VBA является дальнейшим развитием языка программирования, который

применяется

в Windows, разработанного еще в 1963 году.

3.3.1. Суть объектно-ориентированного программирования

Ориентирование на события – это стержень создания приложений Windows в VBA.

Windows – система, базирующаяся на сообщениях и событиях. Это значит, что каждое действие в Windows вызывает событие, которое в виде сообщения передается в приложение. Приложение анализирует сообщение и выполняет соответствующие этому сообщению действия.

Приложения, созданные с помощью VBA, также работают по событийному принципу. Сообщение передается объекту (например, кнопке), где затем вызывается необходимое событие (например, событие C1ick).

Одним из основных понятий VBA, таким образом, является понятие «*ОБЪЕКТ*» .

Объект – это программный элемент, который имеет свое отображение на экране, содержит некоторые переменные, определяющие его свойства, и некоторые методы для управления объектом. Объект является «кирпичиком» для построения программ.

В VBA имеется много встроенных объектов, например:

Range – диапазон ячеек; Cell – ячейка; Sheet – лист.

Каждый объект характеризуется параметрами, которые можно разделить на три категории:

события; методы; свойства.

Событие – это характеристика объекта, которая описывает внешнее воздействие, на которое реагирует объект во время выполнения приложения. События инициируются:

действиями пользователя;

сообщениями от системны средств;

сообщениями от приложения, которое выполняется.

Таким образом, события связаны с определенными действиями пользователя и иницинруют выполнение соответствующих процедур обработки события.

Процедуры описывают необходимые для обработки события действия и пишутся на языке VBA.

Методы – это рабочие операторы объекта. Они позволяют управлять объектом. Например, диапазон ячеек «Начальные_ данные» можно удалить командой Range («Начальньге_ данные») Clear.

Методы реализуются с помою встроенных процедур VBA.

Свойства отвечают за внешний вид и поведение объекта. Объекты объединяются в классы. К одному классу принадлежат объекты с одинаковым набором свойств, методов, событий.

Получить информацию о некотором объекте можно, воспользовавшись специальным каталогом объектов VBA, который содержит список всех объектов, сгруппированных по категориям. Эти категории называются библиотеками объектов.

В VBA имеется около двухсот встроенных операторов и функций. Каждый оператор и функция имеют четкую структуру (синтаксис), то есть правила грамматики, пуинтуацин и орфографии. для записи операторов используются алфавит, различные символы и ключевые слова.

Алфавит языка включает большие и малые буквы латинского алфавита и десятичные цифры от нуля до девяти. Арифметические операции задаются

символами:

+ – сложение;

- – вычитание;

* – умножение;

/ – деление;

\ – деление без остатка;

mod – деление по модулю.

Кроме символов для записи операторов в языке используются ключевые зарезервированные слова, такие как: ***Dim, As, New, If, Then, Else, While, End***. Пользователь не может использовать ключевые слова в других целях.

При изучении выражений целесообразно проводить аналогию между языковой и алгебраической формами их записи. При этом особое внимание следует уделить приоритетам операций и определению типа выражения. Определяя тип выражения, следует руководствоваться следующим правилом: если в выражении «замешаны» операции разных групп (арифметические, сравнения, логические), то тип выражения определяет операция, которая выполняется последней при вычислении выражения. При этом, если выражение содержит несколько операторов, то значения компонентов выражения рассчитываются в определенном порядке. Такой порядок называют порядком старшинства или приоритетом операторов.

Если выражение содержит операторы разных типов, то первыми выполняются арифметические операции, следом за ними операции сравнения, а последним – логические операции. Все операторы сравнения имеют равный приоритет, т. е. выполняются в порядке их расположения в выражении слева направо.

Синтаксис – это правило записи операторов и других конструкций программы. В каждой строке кода программы обычно записывается один оператор, если операторов в строке несколько, то они разделяются двоеточием, например:

У=A+B : X=C*D.

Разделители строк – это символ подчеркивания (_), который дает возможность сформировать длину строки так, чтобы она умещалась на экране. Строка программы в VBA может иметь максимум 1 023 символа и не больше 10 разделителей.

Комментарии используются в языке для пояснения отдельных фрагментов программы и игнорируются во время ее выполнения. Для выделения начала комментария можно использовать верхнюю запятую (`) или команду Rem, например:

A =2 Rem – это оператор присваивания

A =2 `– это оператор присваивания

Таким образом, текст комментария размещается справа от символа комментария.

Для того, чтобы составить программу на VBA необходимо правильно определить типы данных, используемых в программе.

3.3.2. Переменные и типы данных

Напомним, что ***переменная*** – *это именованная область памяти, предназначенная для хранения данных*. Для доступа к переменной достаточно знать имя переменной. Тип данных задает определенный формат или размер содержимого переменной и указывает, что хранит переменная: число, строку, дату и т. д.

В VBA используются явное и неявное объявление переменных. VBA не требует обязательного явного объявления переменных. При *неявном объявлении* переменные просто используются в программе (их тип определяется автоматически операционной системой), при *явном* они предварительно должны быть определены. Например, с помощью оператора Dim:

Dim Name As String явное объявление

A% = 7 неявное объявление.

Типы данных, используемых в VBA, приведены в табл. 3.

Если тип переменной не указан, то по умолчанию ей присваивается тип Variant. Переменные этого типа могут хранить все, что в них поместят, т. е. их тип изменяется в зависимости от последнего присвоения, поэтому он очень удобен.

Таблица 3

Типы данных, используемых в VBA

Тип данных	Размер в байтах	Что обозначает	Диапазон
Byte	1	Однобайтное целое	0-255
Boolean	2	Логическая перем.	0-1(True и False)
Integer	2	Целое число	от -32768 до 32767

Long	4	длинное целое	Все целые числа от -2147483648 до 2147483647
Currency	8	Денежная велич.	От -922337203685477,5808 до 922337203685477,5807
Single	4	Число с плав.точк.	Отрицательные числа от -3.4*1038 до $-1.4*10^{-45}$; Положительные числа от $1.4*10^{-45}$ до 3.4*1038
Double	8	Двойное с план, точк.	Отрицательные числа от -1.8*10308 до $-4.9*10^{-324}$; Положительные числа от $4.9*10^{-324}$ до 1.8*10308
Date	8	Дата \ время	До 9999 г.
String	10+строка	Строка перем. длины	От 0 до 2-х миллиардов символов
Variant	1-10	Любой тип данных	
usertype		Пользоват. тип данных	

В языке VBA имеется две формы представления чисел: с фиксированной точкой (естественная форма) и с плавающей точкой (нормальная форма). Поэтому вопрос записи числовых констант не представляет особой трудности.

При изучении переменных основополагающими понятиями являются понятия идентификатора, индекса, массива переменных. Особое внимание следует обратить на объявление массивов, при котором первый элемент всегда имеет значение индекса (номера), равное нулю.

3.3.3. Структуры данных

В VBA переменные могут быть простыми переменными и переменными с индексами (массивами).

Простая переменная занимает одну ячейку памяти ЭВМ.

Массив – несколько ячеек памяти в зависимости от размера массива. Массивы могут быть *одномерными* и *многомерными*. Например: Dim B(3,3) As Single ‘– двухмерный массив 3×3 (матрицу), состоящий из действительных чисел.

Dim A(12) As Integer ‘– одномерный массив (вектор), состоящий из двенадцати целых чисел, причем первый элемент массива A(0) (0 – базовый индекс массива.), а последний – A(11).

Можно изменить базовый индекс, использовав оператор Option Base 1. После этого индексы массивов будут начинаться с единицы.

Если в процессе вычислений требуется изменять размер массива, то массив объявляется динамическим. Например: Dim R( ) As Integer.

В операторе Dim размер массива не указывается, но в программе необходимо вычислить его размер, присвоить его некоторой переменной, например, n и указать размер динамического массива с помощью оператора ReDim: ReDim (n, n).

Константы. Основное отличие константы от переменной состоит в том, что ее значение нельзя изменять в процессе выполнения программы. Константы всегда сохраняют значения, присвоенные им при разработке. При объявлении констант используется ключевое слово «Const»:

Const Имя константы = Значение.

Константы можно объявлять и с указанием типа данных. Для указания типа данных используются те же ключевые слова, что и при объявлении переменных, например:

Const Pi As Single = 3.14159.

Область определения переменных (видимости переменных). Важной характеристикой переменных является область их определения. В VBA есть три вида областей определения, характеризующих доступность (видимость) переменной:

локальная переменная, доступна только в текущей процедуре;

переменная контейнера, доступна только в текущей форме;

глобальная переменная, доступна во всем проекте.

Стандартные функции VBA. Все функции, используемые в VBA, в общем случае имеют одинаковый формат:

Имя функции (аргументы),

где имя функции – это идентификатор из букв. Изучая функции, не надо стараться запомнить их все. Необходимо знать только основные, например: SIN (X), COS (X), ABS (X), SQR (X). Остальные функции познаются по мере появления необходимости в их использовании.

Операторы VBA. Приступая к изучению операторов, следует в первую очередь ознакомиться с их классификацией. Затем подробно изучить синтаксис и семантику операторов присваивания и ввода – вывода. Первоначально необходимо усвоить простейший оператор, которым является оператор присваивания, предназначенный для присвоения переменной одного значения. Структура оператора:

A=Выражение

Символ «=»означает операцию присваивания.

Порядок выполнения: вычисляется значение выражения, стоящего в правой части оператора и полученное значение присваивается переменной А. Переменные в выражении должны быть одного типа, за исключением, если переменная может быть вещественного типа, а выражение целого. Далее изучаются операторы

ввода INPUTBOX( ) и оператор вывода MSGBOX( ). Знание этих операторов позволяет перейти к изучению структуры программы на языке VBA, которая представляется последовательностью строк. Закрепление знаний по структуре программы необходимо провести путем составления простейших линейных программ.

Программирование на VBA заключается в создании кодов программ, которые прямо либо косвенно реализуют отклики на события.

Если пользователь производит какое-либо воздействие на систему, скажем, нажимает кнопку, тогда в качестве отклика выполняется код созданной им программы (процедуры обработки события). Если такой отклик не создан (не написана соответствующая программа), то система не реагирует на событие.

Таким образом, специальный вид процедур, прямо реагирующих на события, называется процедурами обработки событий, а все другие процедуры VBA являются вспомогательными и называются общими.

Приложение VBA (проект VBA) состоит из одного или нескольких модулей. ***Модуль*** *– это программа, включающая тексты процедур и функций, описывающих реакцию объекта на события*.

В VBA код программы состоит из одного либо нескольких операторов, процедур и функций, которые для выполнения программы преобразуются компилятором в машинный код.

Структуру VBA-программы рассмотрим на примере организации модуля:

```
Option Private Module ‘– заголовок модуля
Option Base 1
Option Explicit
Const pi As =3.14159 ‘– глобальная постоянная
Dim x As Double ‘x – глобальная переменная
Function Disc (R As Double) As Double
‘Функция Disc вычисляет площадь круга
x = 2 ‘ оператор присваивания
Disc = pi *R ^2
End Function
Function Rec (a As Double, b As Double, c As Double) As Double
` Rec – вычисляет площадь треугольника
Dim p As Double ‘p – локальная переменная функции
p = (a+b+c)/2
Rec=sgr(p * (p-a) * (p-b) * (p-c))
End Function
Sub Results ‘Процедура общая языка VBA
Dim R_1, R_2, a, b, c As Double
```

```
‘R_2, a, b, c – локальные переменные процедуры
R_1 = Disc(2.5)
x=x+2
MsgBox("площадь круга = " & CStr(R_1) & " , x = " & CStr(x)
 A=1: b=1: c=Sgr(2)
R_2 = Rec(a, b, c)
MsgBox ("площадь треугольника =" & CStr(R 2))
End Sub
```

Обычно текст программы начинается с опций, которые управляют описанием переменных. В нашем случае первая строка программы Option Вале 1 говорит о том, что в объявленных массивах индекс первого элемента равен единице, а не нулю. Вторая строка программы Option Explicit требует явного объявления переменных. Затем следует объявление глобальных переменных и констант для данного модуля, т. е. таких переменных, которые используются во всех процедурах модуля. Далее располагаются тексты функций пользователя и процедур, составляющих саму программу. В нашем случае объявлены глобальная переменная x и глобальная константа pi. Далее следуют функции пользователя Disc и Rec, вычисляющие соответственно площади круга и треугольника, и процедура Results, организующая вычисление этих площадей при заданных исходных данных. В процедуре в качестве рабочих объявлены локальные переменные R_1, R_2, a, b, c, значения которых сохраняются только внутри данной процедуры.

Процедура MsgBox обеспечивает вывод результатов решения задачи на экран в окно вывода. Функция CStr переводит числовой формат в строковый. Результат решения задачи будет выведен в следующем виде:

Площадь круга = 19.6349375, x=4

Площадь треугольника = 0.5

Составление программы – процесс творческий, но требующий твердых знаний правил записи операторов и порядка их выполнения.

Далее нужно изучить структуры процедур и функций VBA.

Процедура является самостоятельной частью программного кода, которая имеет имя и выполняет заданную последовательность инструкций, может содержать аргументы и изменять их значения.

Процедура имеет следующую структуру:

```
Public {[Private],[Static]} Sub Name (Список аргументов)
        P1
        P2
        …
        PN
Exit Sub – аварийный выход из процедуры
```

PN+1
PN+2
PN+M1
End sub

Рассмотрим элементы описания:

Public – глобальная процедура, доступна для всех других процедур во всех модулях проекта;

Private – процедура модуля, доступна для всех других процедур в данном модуле;

Static – служебное слово, которое говорит о том, что локальные переменные процедуры сохраняются в промежутках времени между вызовами этой процедуры;

Name – имя процедуры, удовлетворяющее стандартным правилам написания имен в VBA. Имя процедуры обработки события состоит из имени объекта и имени события;

Список аргументов – список формальных параметров (аргументов) процедуры Sub (имен переменных, которые должны быть переданы в процедуру при обращении к ней).

P1, P2, …, PN+M – любые инструкции (операторы) языка VBA.

Инструкция ***Exit Sub*** приводит к выходу из процедуры.

Синтаксис элемента структуры «***Список аргументов***»:

[Optional] [ByVal, ByRef] [ParamArray] Имя переменной _ As Тип (по умолчанию).

Optional – ключевое слово,` указывающее на то, что аргумент не является обязательным. Все аргументы, описанные как Optional, должны быть типа ***Variant***. ***ByVal*** – указывает на то, что этот аргумент передается по значению.

ByRef – указывает на то, что этот аргумент передается по ссылке На его адрес в памяти. Отшсание ***ByRef*** используется по умолчанию.

ParamArray – используется только в качестве последнего элемента в списке аргументов для указания о том, что конечным аргументом в списке параметров процедуры является описанный типом Variant массив значений, т. е. это слово позволяет задавать произвольное количество аргументов в процедуре.

Имя переменной – имя переменной, удовлетворяющее стандартным правилам написаъшя имен в VBA.

Тип – это тип данных, которые должны быть переданы в процедуру при обращешш к ней.

По умолчанию – любая константа или выражение, дающее константу.

Используется только вместе с **Optional**. Если указан тип Object, то единственным значением по умолчанию может быть Nothing (пусто).

В VBA различают процедуры обработки событий и общие или пользовательские процедуры. Пользовательские процедуры разрабатываются пользователем и, в отличие от первых, не имеют в заголовке имени события, на которое они реагируют (обрабатывают).

Приведенный ниже пример показывает все основные способы передачи параметров в процедуру.

Рассмотрим процедуру, находящую сумму двух чисел и выводящую результат в диалоговом окне:

```
Sub Assistant (ByVal a As Integer, ByVal b As Integer)
c = a+b
MsgBox CStr (c)
End Sub
```

а) вызов процедуры с конкретными числами как фактическими параметрами:

```
Assistant 1,3
```

6) присвоение переменным значений и вызов процедуры:

```
x = 1: y = 1
Assistant x, y+2
```

в) вызов процедуры с указанием фактических параметров по имени:

```
Assistant a:=l, b := 3
```

г) вызов процедуры с помощью инструкции Call с указанием фактических параметров по имени:

```
Call Assistant (a.=1, b := 3)
```

Функция – самостоятельная часть программного кода, имеющая имя, способная выполнять заданную последовательность инструкций и возвращать основную программу полученное значение:

```
Public {[Private],[Static]}Function Name (Список аргументов) As (Тип Name)
```

Функция построена точно также как процедура. Для определения функции используется ключевое слово «Function». В конце функции пишется End Function.

Служебное слово «Sub» заменено на Function.

Name – имя функции, построенное по правилам языка.

Тип Name – тип и идентификатор возвращаемого значения функции.

Наличие одного возвращаемого значения и возможность его использовать в качестве переменной в выражении u есть основное отличие функции от процедуры.

Примером может служить функция вычисления налога на добавленную стоимость. Функция в качестве аргументов получает сумму нетто и налоговую ставку и должна возвращать сумму налога. Структура функции имеет следующий вид:

```
Function NDS (Netto As Currency, Percent As Single) As Currency
NDS = Netto * Percent
End Function
```

Функция может использоваться в выражении как операнд, например:

```
Y=NDS(F,C)*K
```

Параметры могут передаваться в функцию теми же способами, что и в процедуру, т. е. по ссылке и по значению, а также путем присвоения требуемых значений с помощью оператора присваивания специального вида (:=).

3.3.4. Изучение составных операторов VBA

Вычислительный процесс называется *разветвляющимся*, если предусмотрено разветвление указанной последовательности действий на два и более направлений в зависимости от итога проверки заданных условий.

Такой, процесс описывают разветвляющиеся алгоритмы и программы. Основу алгоритмов составляют конструкции ***выбор*** и ***выбор варианта***, а основу программы – условные операторы и операторы варианта.

Условный оператор служит для организации процесса вычислений по различным путям вьшолнения программй в зависимости от итога проверки условия.

Существует два вида условного оператора:

If < выражение> Then < оператор - 1> – неполный оператор;

If < выражение> Then < оператор - 1> Else < оператор - 2> – полный оператор.

Здесь <выражение> – логическое выражение, принимающее значения True или False.

If (если), Then (то) и Else (иначе)– ключевые слова языка;

<оператор> – любой оператор, в том числе и условный.

Порядок выполнения: если значение выражения – True (условие соблюдается), выполняется оператор 1, если False (условие не соблюдается) – оператор 2, затем в обоих случаях управление передается дальше, к следующему оператору программы.

Если ветвь Else отсутствует (неполный оператор) и условие не соблюдается, то никакие действия не выполняются.

Примеры:

```
1) If X<0        2)If X<0        3)If X<0
   Then          Then            Then
```

```
Y= -X          Y= -1              Y = - X
Else           ElseIf X = 0 Then  End If
Y = X          Y = 0
End If         Else
               Y =1
               End If
```

Особое внимание при изучении условного оператора следует обратить на запись логических условий, порядок их проверки. Если по условию решаемой задачи в условном операторе после служебного слова «Then» необходимо поместить несколько операторов, то их следует записать на одной строке.

Операторы цикла. Основой решения на ЭВМ большинства прикладных задач являются повторения, в частности, многократные повторения расчетов по одним и тем же формулам, но при различных исходных данных. Такие вычислительные процессы (ВП) называются *циклическими*, а повторяющиеся участки – *циклами*.

Главными вопросами, которые необходимо уяснить при изучении программирования циклических ВП, являются:

1) как организовать повторение циклов при решении задачи;

2) сколько раз нужно повторять цикл.

В одних задачах число повторений можно определить заранее до начала решения задачи по исходным данным, в других этого сделать нельзя.

Исходя из этого, программы решения задач называют циклическими программами с заранее известным и заранее неизвестным числом выполнения циклов.

К ВП первого типа относятся задачи расчета суммы и произведения элементов массивов, табулирования функций (вычисления значений функции для ряда значений аргумента) и т. п.

К ВП второго типа относятся задачи определения корней уравнения приближенными методами, вычисление суммы бесконечного ряда, нахождение наибольшего или наименьшего значения функции и т.п.

Для программирования ВП с известным числом повторений используется оператор ***For ..Next***. Оператор повторяет выполнение группы операторов указанное число раз.

Структура оператора:

```
For I=A1 To A2 Step H
        [оператор P1]
        [оператор P2]
        …
        [оператор PN]
[Exit For]
```

[оператор PN +1]
[оператор PN+2]
Next I

Здесь: I – параметр цикла (переменная, с помощью которой организуется счетчик количества повторений цикла). Эта переменная не может принадлежать к типу Boolean или быть элементом массива;

Step (шаг – необязательный аргумент в структуре) – значение, на которое изменяется счетчик при каждом выполнении тела цикла. Если это значение не задано, по умолчанию шаг равен единице;

A1, A2 – начальное и конечное значения параметра цикла;

Операторы P1. ..PN+2 (операторы тела цикла) – любые операторы языка;

Exit For – оператор досрочного выхода из цикла.

Порядок выполнения оператора:

1. Присваивание параметру цикла начального значения I = A1;

2. Проверка условия I<= A2 перед выполнением цикла;

3. Выполнение оператором тела цикла, если условие 1<=A2 выполнено (True) и выход из цикла, если условие I<=A2 Не выполнено (False);

4. Увеличение параметра цикла на величину шага I = I + H;

5. Возврат к началу цикла.

Оператор цикла Do...Loop используется для организации вычислительных процессов с заранее неизвестным числом повторений. Набор инструкций (тело цикла) повторяется, пока условие имеет значение True, либо пока оно не примет значение True.

Имеется два способа проверки условия в операторе Do...Loop с помошью ключевых слов While и Until: `

1) условие проверяется до входа в шла;

2) условие проверяется после хотя бы однократного выполнения цикла.

Поэтому в зависимости от позиции условия выхода из цикла различают два варианта оператора цикла Do...Loop:

1) цикл с предусловием; 2) цикл с постусловием.

Цикл с предусловием Do...While...Loop; Do...Until...Loop

Структура и порядок выполнения:

Do...While «условие»	Do...Until «условие»
P1	P1
P2	P2
P3	P3
…	…
PN	PN
Loop	Loop

Здесь: Do (выполнять), While (пока), Until (до), Loop (петля) – служебные слова;

P1 ....PN – любые операторы языка;

«условие» – булево выражение либо выражение типа сравнение, принимающие значения True (False).

Порядок выполнения:

а) While – тело цикла (операторы P1....PN) выполняется до тех пор, пока условие имеет значение True.

Пример:

```
Sub PrimWhile()
Dim I As Integer
Dim Num As Integer
Dim Y As Single
Y= 0.45 : I=0
Num = 20
Do While Num > 10
Num = Num – 1
1=1 + 1
Y=Y*(I+1)
Loop
MsgBox "Выполнено " & I & " итераций цикла.",”Y=” & Y
End Sub
```

В процедуре PrimWhile условие проверяется до входа в цикл. Если Num задать равным 9 вместо 20, цикл выполняться не будет.

6) Until – тело цикла (операторы P1...PN) выполняется до тех пор, пока условие имеет значение False (пока не примет значения True).

Пример:

```
Sub PrimlUntil()
Dim I As Integer
Dim Num As Integer
Dim Y As Single
Y= 0.45 : I=0
Num = 20
Do Until Num = 10
Num = Num - 1
I: I + 1
Y=Y*(I+1 )
Loop
MsgBox( "Выполнено " & 1 & " итераций цикла.",”У=” & Y)
End Sub
```

Цикл с постусловием Do...Loop...While; Do...Loop...Until

Структура и порядок выполнеъшя:

```
Do                                  Do
    P1                                  P1
    P2                                  P2
    P3                                  P3
    …                                   …
    PN                                  PN
```

...Loop While «условие» … Loop Until «условие»

Здесь отличие от выше рассмотренной структуры заключается в том, что проверка условия выхода из цикла осуществляется не в начале, а в конце цикла.

В этом случае тело цикла. выполняется хотя бы один раз при любом значении условия выхода из цикла.

Примеры:

Условие While

В процедуре Prim2While инструкции внутри цикла выполняются только один раз до того, как условие примет значение False.

```
Sub Prim2WhileO
Dim I As Integer
Dim Num As Integer
Dim Y As Single
Y= 0.45 : I=0
Num = 9
Do
Num = Num - 1
I= I + 1
Y=Y*(I+1)
Loop While Num> 10
MsgBox "Выполнено " & I & " итераций."
End Sub
```

Условие Until

Повторение инструкцшй, пока условие не примет значение True.

```
Sub Prim2UntiI()
Dim I As Integer
Dim Num As Integer
Dim Y As Single
Y= 0.45 : I=0
Num = 1
Do Num = Num +1
I= I + 1
Y=Y*(I+1)
Loop Until Num = 10
MsgBox "Выполнено" & I & "итераций цикла", "Y="& Y
```

End Sub

Условие проверяется после хотя бы однократного выполнения цикла (как показано В процедуре PrimZUntil). Итерации продолжаются, пока условие имеет значение False.

При изучении оператора цикла необходимо помнить об организации вычитающих и суммирующих счетчиков. Роль счетчика будет выполнять параметр цикла. Знание условного оператора и оператора цикла позволяет составить разветвляющиеся И циклические программы, а также программировать более сложные вычислительные процессы («цикл в цикле», «ветвление в цикле» и т.д.). Таким образом, ясно, что VBA располагает мощными средствами для программирования циклических вычислительных процессов и представляет пользователю широкие возможности и свободу выбора этих средств. на заключительном этапе изучсния языкз VBA решается конкретная задача усложненного плана.

3.3.5. Формы и элементы управления VBA

Формы с элементами управления составляют основу визуального программирования. Снабдить незаполненную форму элементами управления можно с помощью панели инструментов Панель элементов. Для этого нужно щелкнуть левой кнопкой мыши на значке нужного элемента управления и установить его на поверхности формы. Некоторые свойства управляющих элементов (размеры, положение) можно изменять непосредственно мышью, другие – с помощью окна свойств. Элементы можно копировать, вырезать и вставлять.

Редактор VBA позволяет раскрыть в главном окне проекта до 7 окон, главными из которых являются:

окно экранной формы;

окно проводника проекта;

окно элементов управления;

окно свойств;

окно разработки формы.

В первом из окон разрабатывается экранная форма, которая в VBA проектируется; во втором – описывается структура проекта, который разрабатывается; в третьем – представлен стандартный набор пиктограмм ЭУ для создания на экранной форме объектов управления; в четвертом – представлен список свойств активного (выделенного) объекта.

Пользовательская форма (диалоговое окно, окно интерфейса программы) – это прямоугольная область экрана с элементами управления. С помощью форм можно: сообщать пользователю информацию; получать информацию от пользователя.

Окно разработки формы дает возможность управлять ее положением на экране во время выполнения программы. Для разработки пользовательской формы используются элементы управления – визуальные средства для создания объектов на экранной форме. Например, на форме можно создать метки, командные кнопки, списки, геометрические фигуры и т.д.

Для создания форм используются средства редактора. Чтобы создать пользовательскую форму (UserForm) нужно:

запустить редактор VBA;

вызвать окно UserForm в открывшемся окне редактора VBA.

Формат обладает свойствами, методами и событиями.

Некоторые свойства формы могут быть установлены программно, а некоторые выбраны в окне свойств.

4. РАБОТА С ФАЙЛАМИ И ОТЛАДКА ПРОГРАММ НА VBA

4.1. Работа с файлами

Составление программы немыслимо без организации долговременного хранения, печати результатов работы. Для решения этих задач на заключительном этапе изучается работа с файлами в языке VBA. Вся файловая система, используемая в языке, не рассматривается, а выбрано только минимально необходимое количество материала. Основополагающими вопросами здесь являются: понятие файла; порядок работы с файлами.

Порядок работы с файлами в языке VBA в основном аналогичен принятому и в других языках программирования высокого уровня. Он состоит в следующем:

открытие доступа к файлу;

запись (или считывание) в файл (из файла);

закрытие доступа в файл.

Все эти действия реализуются соответствующими операторами, которые необходимо изучить.

4.2. Отладка программ на VBA

В VBA заложены большие возможности для отладки программы, т. е. для поиска ошибок в последней. Ошибки можно поделить на три типа: ошибки компиляции, выполнения и логические ошибки.

Ошибки компиляции возникают, если редактор VBA не может интерпретировать введенный код. Строка, в которой содержится ошибка, выделяется красным цветом и на экране отображается диалоговое окно с сообщением о возможной причине, вызвавшей ошибку. В VBA предусмотрена также возможность откомпилировать программу без запуска на выполнение с помощью команды

«ОТЛАДКА», «КОМПИЛИРОВАТЬ» (в этом случае ошибка определяется синим цветом и появляется сообщение о возможной причине).

Ошибки выполнения возникают после успешной компиляции программы при ее выполнении. Причинами ошибок могут быть:

1) некорректная информация при считывании файла с диска;

2) некорректные данные, введенные пользователем, например, требуется число, а пользователь вводит строковую информацию;

3) некорректность вычислений, например, деление на ноль.

В этих случаях на экране отображается диалоговое окно с сообщением о номере ошибки и возможной причине ее появления.

Строка с ошибкой будет помечена стрелкой желтого цвета.

Логические ошибки труднее всего обнаружить И устранить. Эти ошибки не приводят к прерыванию выполнения программы, т. е. визуально все идет гладко и выглядит так, как будто программа выдает неверные результаты. Локализация логических ошибок связана с тщательным анализом алгоритма программы с привлечением средств отладки VBA.

Поскольку VBA – язык не компилируемый, а интерпретируемый, то после того как модуль написан и успешно отпажен, он готов к использованию.

Готовая программа представляет собой хранящийся в шаблоне или документе исходный текст, который переводится в машинный код при каждом ее запуске.

В режиме отладки используются следующие основные приемы:

1. Пошаговое исполнение программы.

Все команды выполняются последовательно, но после каждой из них нужно нажать кнопку <F8> (следующая команда будет подсвечиваться желтым цветом). Так можно увидеть, где ошибка.

2. Просмотр значений переменных.

Подводя курсор в режиме отладки к имени любой переменной, можно увидеть ее значение. А значения их всех имеются в окне «Локальные переменные».

3. Точки останова.

Если программа большого объема, то перемещаться по всем ее строчкам с помощью кнопки <F8> довольно обременительно. Лучше использовать точки останова, т. е. те отметки в тексте программы, где ее исполнение остановится, а она сама перейдет в режим отладки.

В заключение отметим, что в настоящее время очень широкий круг профессиональных задач экономиста может быть решен лишь с использованием современных информационных технологий, одним из элементов которых являются языки высокого уровня. В том числе и изучаемый Вами язык офисного программирования VBA.

СПИСОК ЛИТЕРАТУРЫ

1. Бобыр Е. И. Современные информационные технологии в экономике. Основы программирования в среде MS Office : учеб. пособие для студентов фак. «Бизнес-упр.» / Е. И. Бобыр, С. Б. Данилевич ; Нар. укр. акад. – Харьков : Изд-во НУА, 2017.– 84 с

2. Малачівський П. С. Програмування на VISUAL BASIC : навч. посіб. для студентів ВНЗ / П. С. Малачівський, Я. В. – Львів : Растр-7, 2014. – 407 с.

3. Юрченко І. В. Інформатика та програмування / Юрченко І. В., Сікора В. С. – Чернівці : Вид. Яворський С.Н., 2015. – Ч. 2. – 210 с.

4. Методичні вказівки до лабораторних, практичних, самостійних та контрольних робіт з дисципліни «Інформатика та комп'ютерна техніка» / Харків. нац. ун-т міськ. г-ва ім. О. М. Бекетова ; уклад. : С. В. Дядюн, М. П. Пан, Г. В. Білогурова. – Харків : Вид-во ХНУМГ ім. О. М. Бекетова, 2015. – 113 с.

СОДЕРЖАНИЕ

Навчальне видання

СУЧАСНІ ІНФОРМАЦІЙНФ ТЕХНОЛОГІЇ В ЕКОНОМІЦІ. ОСНОВИ ПРОГРАМУВАННЯ В СЕРЕДОВИЩІ MS OFFICE»

Методичні рекомендації
для студентів, які навчаються за напрямом підготовки
051–Економіка

(російською мовою)

Автор-упорядник Данилевич Сергій Борисович

В авторській редакції
Комп'ютерна верстка *С. Б. Данилевич*

Підписано до друку 12.05.2018. Формат 60×84/16.
Папір офсетний. Гарнітура «Таймс».
Ум. друк. арк. 2. Обл.-вид. арк. 1,4.
Тираж 50 пр. Зам. №

План 2017/2018 навч. р., поз. 4 в переліку робіт кафедри

Видавництво
Народної української академії
Свідоцтво № 1153 від 16.12.2002.

Надруковано у видавництві
Народної української академії

Україна, 61000, Харків, МСП, вул. Лермонтовська, 27.