**ПРЕДИСЛОВИЕ**

**Информационные и компьютерные технологии** — широкий класс дисциплин и областей деятельности, относящихся к [технологиям](http://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F) создания, управления и [обработки](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) [данных](http://ru.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5_(%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0)), в том числе с применением вычислительной техники. В последнее время под информационными технологиями чаще всего понимают [компьютерные технологии](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B5_%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B8). В частности, ИКТ имеют дело с использованием компьютеров и [программного обеспечения](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) для хранения, преобразования, защиты, обработки, передачи и получения информации.

Целью курса является формирование знаний и практических навыков для самостоятельного решения на компьютере инженерных задач с использованием типового математического обеспечения, что позволит применять вычислительную технику в специальных дисциплинах и в инженерной практике. Курс «Информационные и компьютерные технологии» изучаются на втором курсе и относится к циклу естественно-научных дисциплин.

В соответствии с государственным образовательным стандартом высшего образования обучение студентов всех форм подготовки, включая и заочную, осуществляется на основании государственных требований к минимуму содержания и уровню подготовки.

**Язык и среда VBA**

VBA (Visual Basic for Application) – это язык программирования, поддерживаемый всеми приложениями интегрированного пакета Microsoft Office, в состав которого входят такие популярные приложения, как Microsoft Access, Microsoft Excel, Microsoft PowerPoint, Microsoft Word и др.

VBA – относительно несложный язык программирования, который очень прост и удобен в освоении и позволяет быстро получить ощутимые результаты – конструировать профессиональные приложения для решения практически любых задач в среде Microsoft Windows. Можно сказать, что язык VBA является прямым наследником языка программирования Basic, но, тем не менее, в сравнении с ним он шагнул далеко вперед. Поэтому и возможностей, как внутренних (например, в части вывода на экран всевозможных форм, кнопок и других элементов управления), так и в отношении взаимодействия с другими приложениями, в VBA гораздо больше. VBA в отличие от других сред программирования напрямую интегрирован с офисными пакетами. Поэтому, изучив язык Visual Basic for Application, станет понятно, что создавать многие офисные приложения с его помощью гораздо проще и быстрее, чем с помощью других языков программирования.

Следует заметить, что VBA будучи развитым *языком* программирования также включает в себя полноценную интегрированную *среду разработки* программ с полным набором специализированных меню и окон, упрощающих проектирование, правку, отладку и тестирование программ. Интегрированная среда разработки VBA представлена приложением, называемым «Редактором Visual Basic». Этот редактор имеет типичное для приложений Windows экранный интерфейс со строкой меню и целым набором панелей инструментов, которые позволяют получить доступ к целому ряду окон, предоставляющих дополнительные инструментальные средства, необходимые для разработки программ и быстрого создания пользовательского интерфейса, что превращает VBA в полноценную визуальную среду разработки приложений.

**Редактор VBA**

Редактор VBA представляет собой совокупность инструментальных средств, предназначенных для создания *модулей* программ, просмотра существующих модулей, создания и редактирования исходного кода программного модуля, создания пользовательских диалоговых окон и выполнения других задач, относящихся к написанию и обслуживанию программ на VBA. Редактор представляет собой отдельное приложение, которое можно запустить из любого приложения, входящего в состав пакета Microsoft Office.

Чтобы открыть Редактор Visual Basic for Application из среды Excel, нужно либо нажать комбинацию клавиш {**Alt**}+{**F11**}, либо выбрать команду **Сервис** → **Макрос** → **Редактор Visual Basic (**Microsoft Office 2003) или в ленточной вкладке **Разработчик** в группе **Код** выбрать команду **Visual Basic** **(**Microsoft Office 2007, 2010, 2013). После выполнения этих операций на экране появится окно редактора модулей, в котором и происходит вся дальнейшая работа по созданию проекта разработки программы (рис. 1).

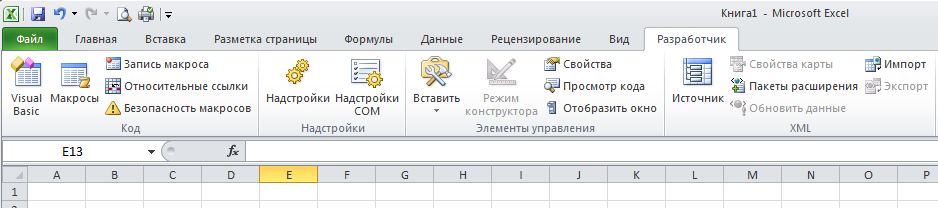


Рис. Окно Excel 2010

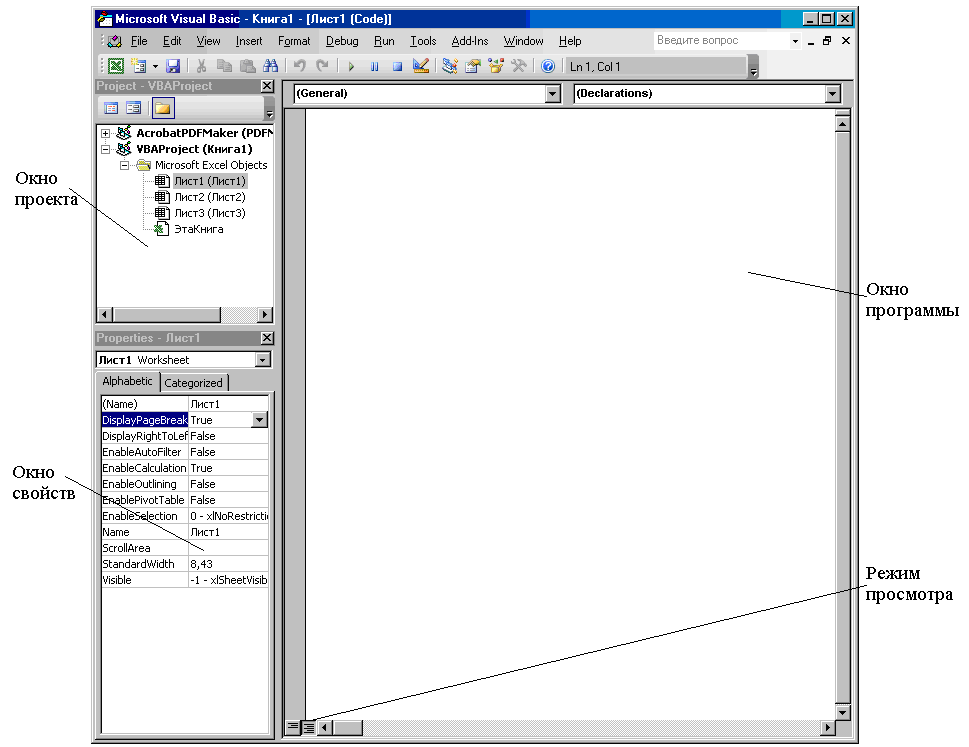


Рис. 1. Окно редактора Visual Basic в Microsoft Excel

Редактор VBA состоит из следующих окон:

– **Окно проекта (Project Explorer)** содержит дерево-диаграмму открытых в данный момент файлов (документы, шаблоны или рабочие книги) и объектов, содержащихся в этих файлах (объекты host-приложения, модули, ссылки, формы и так далее). **Project Explorer** можно использовать для перехода к различным модулям и другим объектам в проекте VB при помощи кнопок (панели инструментов этого окна) **View Code **(Программа), **View Object **(Объект) и **Toggle Folders **(Папки). Для удобства они сгруппированы по нескольким категориям. VBAProject, Лист1, Лист2 и др.

– **Окно свойств (Properties Window)** содержит все свойства объекта текущего выбора. Вкладка **Alphabetic** (По алфавиту) этого окна предоставляет список свойств выделенного объекта, составленный из имен свойств в алфавитном порядке. Вкладка **Categorized** (по категориям) отображает свойства объекта, отсортированные по категориям. Каждый элемент управления, вставленный в форму, имеет набор свойств, с помощью которых можно определить его внешний вид. Все эти свойства собраны в отдельном окне. Окно состоит из двух столбцов. В первом столбце выводятся названия свойств, во втором – их значения. При редактировании формы, модулей и модулей классов часто приходится прибегать к этому окну, меняя свойства элементов управления, такие, как надпись, цвет, размеры и некоторые другие. Для того чтобы изменить значение свойства, следует установить курсор в соответствующую строку и набрать новое значение. Впрочем, значения некоторых свойств можно выбирать из списка или настраивать с помощью диалогового окна.

– **Окно для редактирования программ (Code Window)** – это окно, в котором создается, просматривается и редактируется текст программного модуля (в VBA текст программы называется *кодом*, а отдельный функционально завершенный фрагмент кода называется *процедурой* или *макросом*). В режиме **Full Module View** весь исходный код макроса (процедуры) в модуле отображается сразу в прокручивающемся текстовом окне, а макрос отделяется от другого макроса серой линией. Редактор VB позволяет также просматривать содержимое модуля в режиме **Procedure View** (представление процедуры). Чтобы выбрать режим просмотра, надо щелкнуть по кнопке в нижнем левом углу **Code Window**. *Окно проекта может быть скрыто*, и для его открытия нужно щелкнуть по кнопке **View Code** , расположенной в Окне проекта.

**Панели инструментов редактора VBA**

Для многих пользователей выбрать командную кнопку с помощью мыши легче, чем выбрать команду в меню. Поэтому Редактор VBA, также как и все другие приложения Windows, предоставляет основные и наиболее часто используемые команды в виде кнопок на инструментальной панели. По умолчанию Редактор VBA отображает только панель инструментов **Standard**, показанную на рис. 2.



Рис. 2. Панель инструментов **Standard**

Вид инструментальной панели можно самостоятельно настраивать В конце панели, справа, находится кнопка **More Buttons** . Если, например, необходимо удалить с панели или, наоборот, добавить некоторые кнопки на панель, надо нажать на эту кнопку, а затем указать соответствующую кнопку в появляющемся меню.

Кроме панели **Standard** Редактор VBA предлагает еще три инструментальные панели: **Edit** (правка), **Debug** (отладка) и **UseForm**. Панель **Edit** содержит несколько командных кнопок, полезных при редактировании текста программы в окне **Code Window** (окне редактирования программ). Для управления отображением панелей инструментов на экране используется команда **View → Toolbars** (вид → панели инструментов). Так как Редактор VBA по умолчанию не отображает панель **Edit**, ее нужно выводить на экран вручную.

Панель инструментов **Standard** содержит 19 кнопок. Каждая кнопка является быстрым способом выбора соответствующей команды меню.

Таблица 1. Кнопки панели инструментов **Standard**

|  |  |
| --- | --- |
| Кнопка | Действие |
| 1 | 2 |
| View Microsoft Excel | Переключает на host-приложение VBA, из которого был запущен Редактор VBA. Значок этой кнопки изменяется в зависимости от конкретного приложения, из которого запускался редактор VBA |
| Insert UserForm | При щелчке на кнопке со стрелкой  отображается список объектов, которые можете вставить в текущий проект: UserForm, Module, Class Module или Procedure. Это можно выполнить и при использовании одноименных команд меню Insert (вставка) |
| Save | Сохраняет текущий проект (команда File → Save) |
| Cut | Вырезает выделенный текст или объект и помещает его в буфер обмена; (как команда Edit → Cut) |

Продолжение табл. 1

|  |  |
| --- | --- |
| 1 | 2 |
| Сору | Копирует выделенный текст или объект в буфер обмена;(команда Edit → Сору) |
| Paste | Вставляет текст или объект из буфера обмена в Code Window (окно редактирования программ) или форму пользователя в позицию курсора (команда Edit → Paste) |
| Find | Открывает диалоговое окно **Find** для нахождения определенного слова или фразы в модуле; (команда **Edit** → **Find**) |
| Can't Undo | Отменяет самую последнюю команду, если это возможно (не все действия могут быть отменены); (команда **Edit** → **Undo**) |
| Can't Redo | Возобновляет самую последнюю отмененную команду; так же, (команда **Edit** → **Redo**) |
| Run Macro | Запускает текущую процедуру или форму (команда **Run** → **Sub/UserForm**) |
| Break | Прерывает выполнение программы VBA (команда **Run** → **Break**) |
| Reset | Перезапускает программу VBA (команда **Run** → **Reset**) |
| Design Mode | При нажатии этой кнопки Редактор VBA переходит в режим **Design** (конструктора) (командой **Run** → **Design Mode**) |
| Project Explorer | Отображает Окно проекта (команда **View** → **Project Explorer**) |
| Properties Window | Отображает Окно свойств (команда **View** → **Properties Window**) |
| Object Browser | Отображает диалоговое окно Просмотр объектов (команда **View** → **Object Browser**) |
| Toolbox | Отображает Панель элементов (команда **View** → **Toolbox**) |
| Справка: Microsoft Visual Basic | Отображает окно Справочной системы Microsoft Office для контекстно-зависимой подсказки по текущей задаче |
| Cursor position | Отображает положение курсора. Эта область панели инструментов **Standard**, фактически, не является командной кнопкой и появляется только тогда, когда курсор находится в Code Window. Эта область показывает, в какой строке модуля и в каком столбце строки находится курсор вставки |
| More Buttons | Позволяет добавлять или удалять кнопки панели |

**Типы данных языка Visual Basic. Переменные и константы**

Переменные и константы служат для хранения в программе определенных числовых или текстовых значений. *Переменная* – это величина, значение которой может изменяться в программе. Соответственно, *константа* – это величина, значение которой в программе изменяться не будет (константы можно изменять только редактированием исходного кода VBA).

Каждая переменная в программе имеет не только название (имя), но и тип. *Тип* переменной – это параметр, который определяет диапазон хранимого в ячейки памяти значения (чисел или символов), методы его обработки и объем выделяемой памяти. В приведенной ниже таблице представлены печень основных типов переменных, объем выделяемой памяти на одну переменную и диапазон хранимых значений (как задать в программе нужный тип переменной будет рассмотрено в следующем разделе)

Таблица 2. Типы данных VBA

|  |  |  |  |
| --- | --- | --- | --- |
| Название типа  переменной | Символ суффикса в имени переменной | Размер  в байтах  переменной | Описание и диапазон изменения значений  переменной данного типа (минимальное и максимальное значения) |
| 1 | 2 | 3 | 4 |
| Integer | % | 2 | Используется для хранения *целого* числа  в диапазоне от –32768 до +32767 |
| Long | & | 4 | Используется для хранения *целого* числа  от –2147483648 до 2147483647 |
| Byte |  | 1 | Используется для хранения *целого положительного* числа в от 0 до 255 |

Продолжение табл. 2

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Boolean |  | 2 | Используется для хранения *логических* значений (переменная имеет только два возможных значения – **True** (Истина) или **False** (Ложь) |
| Single | ! | 4 | Используется для хранения *дробного* числа (8 десятичных знаков после запятой).  Отрицательные числа:  от –3.402823Е38 до –1.401298Е45  Положительные числа:  от 1.401298Е-45до3.402823Е38 |
| Double | # | 8 | Используется для хранения *дробных* чисел с повышенной точностью (16 десятичных знаков после запятой)  Отрицательные числа: от  –179.769313486232D+308 до  –4. 94065645841247D–324.  Положительные числа: от  4.94065645841247D–324 до  1.797693 13486232D+308 |
| String  (переменной длины) | $ | 10 байт + длина строки | Используется для хранения *текстовой* переменной. Может содержать от 0 символов до приблизительно 2 миллиардов символов |
| String  (фиксированной длины) |  | Длина строки (1 байт на символ) | Используется для хранения *текстовой* переменной. Может содержать от одного до приблизительно 600 тыс. символов |
| Variant |  | 16 байт + 1 байт/символ | Тип Variant может хранить значения любого другого типа данных. Диапазон для данных типа Variant зависит от фактически сохраняемых данных.  Этот тип данных является типом данных *По умолчанию* (если в программе отсутствует указание типа какой-либо переменной, то VBA приписывает ей тип Variant и выделяет 16 байт) |

Окончание табл. 2

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Date |  | 8 | Используется для хранения информации *о дате и времени*. Диапазон дат может быть от 1 января 100 года до 31 декабря 9999 года Диапазон времени от 00:00:00 до 23:59:59 |

При выборе *имени переменной* в VBA необходимо соблюдать следующие обычные в программировании правила:

– имя переменной должно начинаться с буквы алфавита (допускается только латиница);

– после первой буквы имени переменной может стоять любая комбинация цифр, букв или символов подчеркивания, или нижнее тире (\_);

– имена переменных не должны содержать пробелы, знаки препинания или любые другие *специальные символы*, например, символы математических операций (=, +, –, / ) и т.п.;

– имена переменных могут быть любой длины , но не должны превышать 255 символов;

– имя переменной *не должно дублировать* определенные ключевые слова языка VBA;

– имя переменной должно быть уникальным в рамках его *области действия*, то есть в пределах процедуры или модуля, в котором данная переменная объявляется.

Имена переменных не «чувствительны» *к регистру* символов, то есть написание имени переменной прописными (заглавными) или строчными (маленькими) буквами не имеет значения. Например, имена FirstMyVar и firstmyvar представляют для VBA одно и то же имя. При выборе имен переменных надо делать их по возможности более информативными (например, имя AllSum для переменной, хранящий некоторую сумму данных, будет более информативным, чем простой символ S).

Имена могут иметь не только переменные, но и константы (*именованные константы*). Использование именованных констант повышает читабельность процедур. Например, процедура, выполняющая геометрические вычисления, легче читается и более понятна, если использовать именованную константу Pi вместо константы, записанной непосредственно в числовой форме – 3,14.

Переменная типа String имеет некоторые особенности. П умолчанию переменные данного типа принимаются средой VBA строками *переменной длины*. Строковые переменные переменной длины изменяют длину в зависимости от конкретной длины строки, сохраняемой в данной переменной. Иногда может понадобиться использовать строку *фиксированной длины* (строки фиксированной длины всегда имеют одну и ту же длину, указываемую при объявлении переменной). Это может быть полезно в случае, если необходимо обеспечить, чтобы текст, сохраняемый в строковой переменной, всегда содержал одно и то же число символов.

**Задание типа данных переменной**

Хотя объявлять тип переменным не обязательно ( VBA автоматически присваивает им тип **Variant**), объявление типа переменных имеет несколько преимуществ:

– объявленные переменные ускоряют выполнение кода программ (при необъявленной переменной среда VBA тратит некоторое время на анализ переменной, которой по умолчанию присвоен тип **Variant,** чтобы определить ее конкретный тип);

– объявленные переменные повышают эффективность кода. Переменные типа **Variant** могут занимать гораздо больше памяти, чем переменныеопределенных типов. Объявленная переменная занимает только тот размер памяти, который необходим для хранения данных этого определенного типа, поэтому это может значительно сократить объем памяти, требуемый для программы в целом;

– программный код с переменными объявленного типа легче читать и понимать.

Объявить тип переменной можно явно и неявно.

Для *явного* объявления переменной и ее типа используется оператор **Dim**, который в общем случае имеет следующий формат записи:

Dim name\_l As type\_l, name\_2 As type\_2, ...

Примеры правильного синтаксиса для объявлений типа переменных:

Dim a, b, sum, z12 As Integer

Dim DataPlata As Date

Dim Message1 As String

Dim Counter2 As Integer

Dim Profit As Single, Message As String

При *неявном* объявлении тип переменной задается добавлением суффикса *символа определения типа* в конец имени переменной (см. табл. 2), например:

MyVar% = 25 это соответствует типу Integer

CelsiusTemp! = 15,35 это соответствует типу Single

Title$ = "Excel" это соответствует типу String

Хотя неявное объявление переменных может показаться более удобным, с ним связаны некоторые проблемы. Когда переменные объявляются неявно, существует риск нечаянно создать новую переменную, когда на самом деле, необходимо использовать уже существующую переменную, или использовать существующую переменную, когда надо создать новую. Обе эти ситуации приводят к ошибкам в коде, которые очень трудно потом отслеживать. Чтобы заранее не допустить проблем, связанных с непроизвольным неявным объявлением переменных, VBA предоставляют команду **Option Explicit**. При использовании команды **Option Explicit** VBA требует *обязательного* объявления всех переменных перед их использованием.

После объявления переменной независимо от того, объявляется ли эта переменная явно или неявно, эта переменная сохраняет тот же самый тип столько времени, сколько она существует. Нельзя переобъявить переменную или переопределить ее тип.

**Встроенные математические функции**

Программы с численными расчетами как правило предполагают использование различных математических функций, таких как вычисление корня, тригонометрические функции и т.д. Для наиболее часто встречающихся математических функций в алгоритмический язык как правило включаются *встроенные подпрограммы* для вычисления этих функций. В табл. 3 приведен перечень математических функций, реализованных в VBA. В этой таблице X означает любое численное выражение; все аргументы функций являются обязательными, если только не указано иначе.

Таблица 3. Встроенные математические функции VBA

|  |  |
| --- | --- |
| Функции (аргументы) | Возвращает / действие |
| 1 | 2 |
| Abs(X) | Возвращает абсолютное значение X |
| Atn(X) | Возвращает арктангенс X как угол в радианах |
| Сos(X) | Косинус угла X, где X это угол, измеряемый в радианах |
| Sin(X) | Возвращает синус угла; X – это угол, измеряемый в радианах |
| Sqr(X) | Возвращает корень квадратный из X. VBA отображает ошибку, исполнения, если X – отрицательное |
| Tan(X) | Возвращает тангенс угла; X – угол в радианах |
| Exp(X) | Возвращает экспоненту – константу *е*, возведенную в степень *х*. (е – это основание натурального логарифма) |
| Fix(X) | Возвращает целую часть X. Fix не округляет число, а отбрасывает любую дробную часть. Если X является отрицательным, Fix возвращает ближайшее отрицательное целое, большее, чем или равное X |
| Int(X) | Возвращает целую часть X. Int не округляет число, а отбрасываетлюбую дробную часть. Если X является отрицательным, Int возвращает ближайшее отрицательное целое меньшее, чем или равное X |
| Log(X) | Возвращает натуральный логарифм X |
| Rnd(X) | Возвращает случайное число; аргумент является необязательным. Функцию Rnd используется только после инициализации VBA-генератора случайных чисел оператором Randomize |
| Sgn(X) | Возвращает знак числа:  –1, если X отрицательное;  1, если X положительное;  0, если X равно 0 |

**Порядок вычисления математических выражений**

Математические выражения в широком смысле – это выражения или формулы, в которых могут присутствовать операции вычисления арифметических, алгебраических, тригонометрических и других математических выражений, а также логические операции сравнения (больше, меньше и т.п.). Все математические выражения состоят из одной или более следующих частей:

– константы;

– переменные (в том числе элементы массива);

– знаки математических операций;

– функции.

Выражения используются для выполнения непосредственно математических вычислений, но и для выполнения логических операций сравнения. Вычисленные выражения, так же как и переменные, имеют один из типов данных Visual Basic.

При вычислении сложных выражений VBA следует таким правилам:

– все операции выполняются последовательно;

– части выражения, заключенные в круглые скобки, всегда вычисляются в первую очередь. Если выражение, заключенное в круглые скобки, является частью другого сложного выражения, также заключенного в круглые скобки, VBA применяет эти же правила последовательно от внутренних выражений в круглых скобках к внешним;

– конкретные операции выполняются в зависимости от иерархии операторов (см. табл. 4); если операторы имеют равный уровень приоритета, они вычисляются в порядке слева направо;

– если в выражении присутствуют арифметические и логические операции, то сначала выполняются арифметические, и только потом логические.

В табл. 4 приведена иерархия последовательности выполнения математических операций VBA. Все операции перечислены в порядке убывания приоритетов от самого высокого до самого низкого. Операции, помещенные в одной и той же строке таблицы, имеют равный приоритет.

Таблица 4. Иерархия операторов/операций

|  |  |
| --- | --- |
| Оператор/  операция | Комментарии |
| 1 | 2 |
| ^ | Возведение в степень, наивысший приоритет |
| - | Унарный минус (присвоение знака числу) |
| \*, / | Умножение и деление имеют равные приоритеты; они вычисляются по мере появления в выражении слева направо |
| \ | Вычисляет результат целочисленного деления первого математического выражения (X) на второе (Y). Перед вычислением значение каждого выражения округляется до целых по правилам математики. |
| +, - | Сложение и вычитание имеют равный приоритет; они вычисляются по мере появления в выражении слева направо |
| & | Сложение строк выполняется после любых арифметических операций в выражении и перед любыми операциями сравнения или логическими операциями |
| <, <=, >, >=, =, <> | Все операторы сравнения имеют равные приоритеты и вычисляются по мере появления в выражении слева направо. Для группирования операторов сравнения в выражениях надо использовать круглые скобки |
| Not | Логическое отрицание |
| And | Функция И (логическое умножение) |
| Or | Функция ИЛИ (Логическое сложение) |
| Xor | Исключающее ИЛИ |
| Eqv | Функция эквивалентности |
| Imp | Функция импликации |

**Организация диалога между пользователем и программой.**

**Ввод данных в программу и вывод результата**

Между программой VBA и пользователем, как правило, происходит постоянный обмен данными – получение программой VBA данных от пользователя с сохранением их в заданных переменных и отображение результатов выполненных действий. Все эти операции ввода/вывода выполняются при помощи специальных операторов VBA. При этом обмен данными между программой VBA и пользователем можно осуществить двумя способами.

Первый способ – использование операторов (функций) ввода/вывода **InputBox** и **MsgBox.**

Данные, вводимые пользователем, называются входными данными*.* Чтобы передать входные данные от пользователя программе VBA, т.е. *ввести* данные в программу, используется функция **InputBox**. При вызове функция **InputBox** на экране монитора отображается *диалоговое окно*, в котором программа VBA запрашивает (*приглашает*) пользователя ввести некоторое значение в *текстовое окно*. Кроме текстового приглашения и окна ввода данных диалоговое окно содержит также командные кнопки **ОК** и **Cancel.** По нажатию кнопки **ОК** содержимое текстового окна будет записано на переменную, указанную в *левой части выражения* вызова функции **InputBox.** Функция **InputBox** имеет следующий синтаксис:

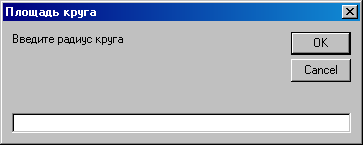
stringvar = InputBox (Promt [, Title])

Здесь stringvar – это переменная пользователя. Аргумент Prompt представляет любое строковое значение (символ, символьную константу или переменную). В диалоговом окне этот аргумент будет отображаться как текст *приглашения* к вводу данных (см. пример и рисунок ниже). Указание этого аргумента является обязательным. В случае, если нет необходимости указания какого-либо приглашения, можно просто привести пустые кавычки (пустая символьная константа).

Аргумент Title является необязательным вторым аргументом для функции **InputBox**. Параметр Title представляет собой также любое строковое значение (символ, символьную константу или переменную). Функция **InputBox** отображает символьное содержимое этого аргумента в *строке заголовка* диалогового окна. Если при вызове функции **InputBox** аргумент Title опущен, VBA отображает в строке заголовка диалогового окна название открытого приложения (Microsoft Excel). Пример вызова функции **InputBox** в программе VBA:

R1 = InputBox("Введите радиус круга", "Площадь круга")

В результате выполнения этого действия VBA выводит на экран монитора следующий запрос:



*Вывод данных* производится командой (оператором) **MsgBox** (вывод ответа в диалоговом окне). При выполнении оператора **MsgBox** программа VBA выводит на экран монитора *диалоговое окно* (см. пример ниже). Оператор **MsgBox** имеет следующую форму записи:

MsgBox "Promt", [Buttons As VbMsgBoxStyle = vbOKOnly], [Title]

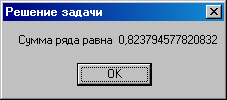
Следует подчеркнуть, что **MsgBox** является не функцией, а оператором (командой), т.е. он не содержит в левой части переменную. Кроме того, обратите внимание, что список аргументов оператора **MsgBox** не заключен в скобки.

Первый аргумент Promt является *текстовым комментарием*, который будет отображен в выводимом диалоговом окне. Второй аргумент Buttons As VbMsgBoxStyle = vbOKOnly является необязательным. Он определяет, сколько и какого типа командные *кнопки* появляются в диалоговом окне, отображаемом посредством **MsgBox**. Когда этот необязательный аргумент опущен, диалоговое окно содержит только одну кнопку – **OK** (если какой-либо необязательный аргумент опускается, то для продолжения указания списка аргументов после предыдущего аргумента обязательно ставится *две* запятые).

Третий аргумент Title определяет *заголовок* диалогового окна. Подобно первому аргументу текст для строки заголовка диалогового окна заключается в кавычки. Пример:

MsgBox "Сумма ряда равна " & z, , "Решение задачи"

После выполнения этой команды на экране отображается диалоговое окно, сообщающее результат – вместо переменной z выводится ее численное значение.



Функция **InputBox** и оператор **MsgBox** используется как правило для ввода и вывода единичных данных и различного рода сообщений на экран монитора. Для более объемных данных более удобен ввода/вывод не в диалоговое окно, а на *Рабочий лист* Excel. Это производится в программе VBA с помощью ключевого имени (функции) **cells**. Функция cells имеет следующий формат:

Имя\_переменной = Cells(i, j)

i – номер строки; j – номер столбца. Например, строка программы

xEnd = Cells(3, 2)

присваивает переменной xEnd числовое значение, записанное в ячейке Рабочего листа Excel, а именно, – в 3-й строке, 2-м столбце.

Для вывода результата в рабочий лист Excel используется запись с обратным расположением переменной и ячейки Excel:

Cells(i, j) = Имя\_переменной

Номер строки и столбца можно указывать как в виде конкретных целых чисел, так и в виде переменных (их числовое значение должно быть целым и определено заранее). Например, выполнения действий:

i = 3

Cells(i, 7) = x

программа VBA выводит численное значение переменной *x* в ячейку рабочего листа, находящейся на пересечении 3-й строки, 7-го столбца.

**Программы с простым линейным следованием операторов**

**(алгоритмическая структура следование)**

Программы с алгоритмом следование, т.е. без ветвлений и циклов, используют для решения простых задач, не требующих передачи управления или циклических вычислений. Такими задачами являются различные единичные расчеты математических функций и выражений, ввод и вывод результатов и т.п. Алгоритмы следование означает, что выполнения действий в них проходят последовательно, шаг за шагом, от одного оператора к другому, oт начала к концу (сверху вниз по блок-схеме).

Простейшая программа решения математической задачи имеет три основных блока или действия: *задание типа переменных и ввод данных, вычисление выражения, вывод результата*.

**Пример 1.** Вычислить значение функции *y* при значении *х* = 45°.

.

Sub primer\_1()

Dim x As Integer ‘ объявление типа переменных

Dim y As Single

Dim pi, x\_rad As Double

'задаем числовое значение х

x = 45

'определяем точное числовое значение числа Пи

pi = 4 \* Atn(1)

' переводим х в радианы, так как в тригонометрических функциях аргумент должен быть в радианах

x\_rad = x \* pi / 180

' производим расчет заданного математического выражения

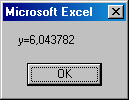
y = (2 \* Cos(x\_rad - x\_rad / 6) + x) / (1 / 2 + Sin(x\_rad) ^ 2 + Sqr(x))

'выводим результат в диалоговое окно

MsgBox "y=" & y

End Sub

Результат программы:



**Оператор разветвляющейся структуры If … Then**

Базовая структура ветвление (называемая также ЕСЛИ–ТО–ИНАЧЕ) обеспечивает в зависимости от результата проверки условия (истина или ложь) выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к общему выходу (продолжению алгоритма). Работа алгоритма продолжается независимо от того, какой путь будет выбран. Возможные пути выполнения алгоритма помечаются на схемах алгоритмов соответствующими метками: "да"/"нет" (или "1"/"0"). Алгоритм, в состав которого входит базовая структура ветвление, называется разветвляющимся алгоритмом, а реализуемый им вычислительный процесс – разветвляющимся вычислительным процессом.

**If … Then … Else** – управляющий оператор VBA, осуществляющий условное ветвление операций, основанное на оценке логического выражения. Выражение может быть истинным или ложным. Оператор имеет две формы записи –линейную и блочную.

**Линейный синтаксис оператора If … Then**

При линейном синтаксисе весь оператор записывается *в одну строчку* (перенос на новую строку не допускается).

**If** логическое\_выражение **Then** операторы 1 [**Else** операторы 2]

– логическое\_выражение – выражение, возвращающее ненулевое значение (истина) или ноль (ложь) (если логическое выражение состоит из нескольких составных частей, то они соединяются друг с другом посредством логических функций);

– операторы 1 – операторы, выполняющиеся при значении логического выражения "истина" (если операторов несколько штук, то один от другого отделяется двоеточием);

– операторы 2 – операторы, выполняющиеся при значении логического выражения "ложь" (если операторов несколько штук, то один от другого отделяется двоеточием).

Выражение, стоящее в квадратных скобках является *не обязательным параметром*. Таким образом, можно выделить два вида записи линейной формы – краткую и полную.

*Краткая форма* записи (Если … То …. ) не содержит часть **Else** операторы 2.

**If** логическое\_выражение **Then** оператор1

– логическое\_выражение – любое логическое выражение, допустимое в Бейсике;

– оператор1 – любой оператор (или группа операторов в одну строку через разделитель – двоеточие) Бейсика, который исполняется при выполнении условия, заданного логическим\_выражением. Действие оператора **If** поясняется блок-схемой, приведенной на рис. 3.

условие

выполняется?

оператор 1

да

нет

выход

Рис. 3. Краткая форма оператора **If … Then**

*Полная форма* записи (Если … То …. Иначе) содержит часть **Else** операторы 2.

**If** логическое\_выражение **Then** операторы 1 **Else** операторы 2

– операторы 2 выполняется только тогда, когда логическое\_выражение ложно. Действие оператора **If** поясняется блок-схемой, приведенной на рис. 4.

условие

выполняется?

оператор 1

да

нет

выход

оператор 2

Рис. 4. Полная линейная форма оператора **If…Then**

**Пример 2.** Определение количества знаков в числе от 0 до 1000

Sub lineynaya\_forma\_If()

Dim x As Single

Dim y As Integer

m1: x = InputBox("Введите целое положительное число в интервале от 0 до 1000", "Запрос задачи")

'повтор ввода, если ввели не отвечающее требованиям число

If x < 0 Or x > 1000 Or x <> Int(x) Then GoTo m1

If x < 10 Then y = 1

If x < 100 Then y = 2

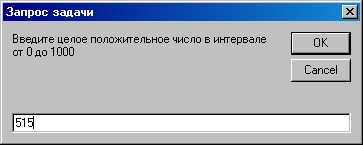
If x < 1000 Then y = 3

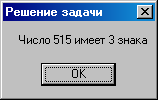
If x = 1000 Then y = 4

MsgBox "Число " & x & " имеет " & y & " знака", , "Решение задачи"

End Sub

Результат программы:





**Блочный синтаксис оператора If … Then**

Эта форма применяется, когда необходимо выполнить несколько операторов или выполнить более сложный выбор в процедурах, выбирая между двумя, тремя или более ветвями. При этом исполняемые операторы записываются в несколько строк – блок. Блок должен заканчиваться оператором **End If** (конец блока **If**).

**If** логическое\_выражение\_1 **Then**

[блок\_операторов\_1]

**ElseIf** логическое\_выражение\_2 **Then**

[блок\_операторов\_2]

……………………………………………………………………………………………………

**ElseIf** логическое\_выражение\_N–1 **Then**

[блок\_операторов\_N–1]

**Else**

[блок\_операторов\_N]

**End If**

– логическое\_выражение – выражение, возвращающее не нулевое значение (истина) или ноль (ложь);

– блок\_операторов – любое количество операторов.

При блочной форме проверяется сначала первое логическое выражение. Если оно ненулевое ("истина"), выполняются операторы блока, следующего за словом **Then** (блок\_операторов\_1). Если логическое выражение равно нулю ("ложь"), происходит переход к оценке условия, заданного в последующей строке **ElseIf**. При выполнении этого условия выполняются операторы данного блока (блок\_операторов\_2) и т.д. Если ни одно из условий **ElseIf** не выполняется (все "ложь"), то выполняются операторы блока **Else** (блок\_операторов\_N). Последний блок операторов должен заканчиваться строкой **End If**.

Любую программу можно составить с использованием только самой простой формы оператора **If** – краткой линейной формы. Все остальные формы используются только для удобства записи программы. Как было уже сказано, если проверяются одно или два условия, то имеет смысл использовать более простую линейную форму **If…Then** или **If...Then…Else**.

Блочная форма применяется, когда необходимо анализировать не одно или два условия, а целый ряд взаимосвязанных условий:

**If** логическое\_выражение\_1 **Then**

[блок\_операторов\_1]

**ElseIf** логическое\_выражение\_2 **Then**

[блок\_операторов\_2]

……………………………………………………………………………………………………

**ElseIf** логическое\_выражение\_N–1 **Then**

[блок\_операторов\_N–1]

**Else**

[блок\_операторов\_N]

**End If**

Действие оператора **If** поясняется блок-схемой, приведенной на рис. 5.

условие 1

выполняется?

операторы 1

да

нет

условие 2

выполняется?

операторы 2

да

нет

условие N-1

выполняется?

опер. N-1

да

нет

выход

опер. N

Рис. 5. Блочная форма оператора **If…Then**

**Пример 3.** Определение количества знаков в числе от 0 до 1000.

Sub blochnaya\_forma\_If()

Dim x As Single

Dim y As Integer

m1: x = InputBox("Введите целое положительное число в интервале от 0 до 1000", "Запрос задачи")

If x < 0 Or x > 100 Or x <> Int(x) Then

'повтор ввода, если ввели не отвечающее требованиям число

MsgBox "Вы ввели не верное число", , "Повторите ввод"

GoTo m1:

ElseIf x < 10 Then

y = 1

ElseIf x < 100 Then

y = 2

ElseIf x < 1000 Then

y = 3

Else

y = 4

End If

MsgBox "Число " & x & " имеет " & y & " знак(а)", , "Решение задачи"

End Sub

**Логические выражения, логические операции и операции отношения**

Логическое выражение, стоящее после оператора **If** или **ElseIf** может быть истинным или ложным. В отличии от арифметических выражений логические выражения принимают не числовые значения, а логические значения – "TRUE" (истина) и "FALSE" (ложь).

В логических выражениях можно использовать следующие *операции отношений*: =, <, >, < =, > =, < > (равно, меньше, больше, меньше либо равно, больше либо равно, не равно).

В логическом выражении допустимы два или несколько условий (операций отношения), которые связываются *логическими операциями* **And**, **Or**, **Not** (И, ИЛИ, НЕ). Логические операции осуществляют *манипуляции над битами*. Они возвращают значение "Истина" (не нулевое) и "Ложь" (нулевое) значения. Описание логических операций приведено в таблице 5.

Таблица 5. Логические функции языка VBA

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | Название | Логическое выражение | Объяснение |
| **And** | Логическое  умножение (И) | **А And B** | Логическое выражение истинно, когда выполняются оба условия одновременно (истинно А *и* истинно В). |
| **Or** | Логическое сложение (ИЛИ) | **А Or В** | Логическое выражение истинно, когда хотя бы одно из условий выполняется (истинно *или* А *или* В *или* оба условия одновременно). |
| **Not** | Логическое отрицание (НЕ) | **Not A** | Логическое выражение истинно, когда условие A *не* выполняется (А ложно). |
| **Xor** | Исключающее ИЛИ | **А Xor В** | Логическое выражение истинно, когда условия А и В не совпадают. |
| **Eqv** | Эквивалент-ность | **А Eqv В** | Логическое выражение истинно, когда А и В одновременно истинны или одновременно ложны. |
| **Imp** | Импликация | **A Imp B** | Логическое выражениепринимает значение "ложь", если A истинно, а В ложно, и значение "истина" в других случаях. |

**Пример 4.** Определить, попадает ли точка с координатами под кривую функции , заданную в интервале .

Sub Primer()

Dim x0 As Single, x\_radian As Double

Dim y0 As Double, y As Double

x0 = InputBox("Введите х координату точки от 0 до 360 градусов", "Запрос задачи на попадание")

If x0 < 0 Or x0 > 180 Then

x0 = InputBox("Вы ввели неверное число. Повторите ввод", "Повторный запрос")

End If

y0 = InputBox("Введите у координату точки", "Повторный запрос")

' перевод х из градусов в радианы

x = x0 \* 4 \* Atn(1) / 180

' вычисляем значение функции

y = Sin(x\_radian)

If y > y0 Then

MsgBox "Точка лежит под кривой", , "Ответ"

ElseIf y = y0 Then

MsgBox "Точка лежит на кривой", , "Ответ"

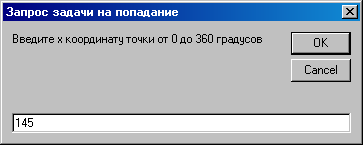
ElseIf y < y0 Then

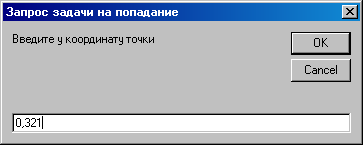
MsgBox "Точка лежит над кривой", , "Ответ"

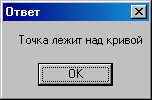
End If

End Sub

Результат программы:







**Циклические программы. Оператор For…Next**

В программировании часто возникает необходимость многократного выполнения однотипных операций, выполняемых по определенному алгоритму. Такие задачи называют циклическими, а группу операторов, которые используются в программе многократно – циклом. Различают циклы арифметические и итерационные. Примером циклического алгоритма является задача ввода в программу массива данных, их обработка и вывод.

Реализация арифметического цикла осуществляется операторами **For…Next.**

Цикл **For…Next** организует счетный цикл и используется, когда необ­ходимо повторить действие или ряд действий заданное количество раз, из­вестное до начала выполнения цикла.

Цикл **For...Next** имеет следующий синтаксис:

**For** счетчик = Начало **То** Конец [**Step** Шаг]

[Операторы цикла]

**Next** счетчик

Счетчик — любая переменная, по которой организован цикл.

В квадратных скобках указан необязательный параметр.

По умолчанию VBA увеличивает счетчик на 1 каждый раз при выпол­нении операторов в цикле (шаг по умолчанию равен 1). Если нужно другое значение шага, надо включать в оператор **For** необязательное ключевое сло­во **Step**.

При выполнении цикла **For…Next** VBA поступает следующим образом:

– присваивает значение Начало переменной счетчик;

* выполняет все операторы, представленные с помощью Операторы  
  цикла, пока не достигнет ключевого слова **Next**, которое указывает VBA на  
  то, что достигнут конец тела цикла;
* изменяет счетчик на величину Шаг (если в оператор **For** включается необязательное ключевое слово **Step**; если **Step** не определено, то VBA по умолчанию увеличивает пере­менную на 1);
* возвращается к началу цикла и сравнивает текущее значение счетчика  
  со значением Конец. Если значение счетчика меньше или равно Конец, VBA  
  выполняет цикл снова. Если значение Счетчика больше значения Конец,  
  VBA продолжает выполнение кода с первого оператора после ключевого  
  слова **Next**.

Если Шаг >0, то такой цикл называется *циклом с возрастающим счет­чиком;* если Шаг <0 – *цикл с убывающим счетчиком.* В последнем случае Начало должно быть больше, чем Конец.

Цикл **For…Next** выполняется только в том случае, если начальное зна­чение счетчика цикла плюс шаг изменения значения счетчика меньше или равно конечному значению счетчика. Если конечное значение счетчика меньше начального значения, то шаг должен быть отрицательным. Цикл вы­полняется до тех пор, пока текущее значение счетчика не выйдет за рамки конечного значения. При завершении текущего цикла к значению счетчика прибавляется значение шага. Если начальное и конечное значения счетчика совпадают, то цикл выполняется один раз, вне зависимости от значения шага. Если шаг равен нулю, цикл продолжается неопределенное количество раз.

**Пример 5.** Построить заполнить таблицу значениями функции

 на интервале 16° < *х* < 40° с шагом 4°.

Sub Demo\_ForNext()

' Объявляем начало, конец цикла, шаг

Dim xStart, xEnd, xStep As Integer

Dim x As Integer

Dim i As Integer

Dim xradian, у As Single

‘ Чтение числовых значений из рабочего листа Excel

xStart = Cells(2, 2)

xEnd = Cells(3, 2)

xStep = Cells(4, 2)

' Номер строки заголовка таблицы значений функции

i = 1

For x = xStart To xEnd Step xStep

' Вычисляем значение х в радианах

xradian = 3.14\* х/180

' Вычисляем значение функции

у = (2.51 \* Sin (xradian ) / *(2 + 3\** Cos (xradian))^ (1 / 3))

i = i + 1

' Передаем полученные значения в рабочий лист

Cells(i, 4) = х

Cells(i, 5) = y

Next x

End Sub

**Итерационные циклы**

Рассмотрим новый тип организации циклических программ – итерационные циклы. Данную конструкцию применяют тогда, когда неизвестно сколько раз должен повториться цикл. А чтобы цикл свою работу завершил, необходимо внутри цикла изменять значение переменной, которая используется в проверке до достижения требуемой точности или выполнения заданного условия. Для организации итерационных циклов используются операторы: **While...Wend**, **Do...Loop**, которые записываются в следующих форматах:

1) **While**  условие

[тело цикла]

**Wend**

2) **Do While** условие

[тело цикла]

**(Exit Do)**

**Loop**

3) **Do**

[тело цикла]

**(Exit Do)**

**Loop While** условие

Условие – логическое выражение, имеющее значение «истина» (когда условие выполняется) и значение «ложь» ( когда условие не выполняется).

Тело цикла – любое количество операторов. Пока условие истинно операторы будут циклически выполняться.

Как только результат проверки условия изменится на «ложь» (его значение равно нулю), VBA обходит цикл и передает управление строке следующей за **Wend** или **Loop**. Оператор **While** ...**Wend** работает аналогично оператору **For...Next**. Разница заключается в том, что в конструкции **For...Next** нужно определить начальное и конечное значения до начала цикла. После чего счетчик цикла просчитает все значения. Цикл **While...Wend** управляется условиями, которые могут иметь значения «истина» или «ложь», в соответствии с поставленной задачей.

Оператор **Do...Loop** очень похож на оператор **While...Wend**, но он более гибкий и в этом его преимущество. Управляющее условие в данном случае может быть размещено как в начале цикла (2), так и в конце (3). Из цикла возможен альтернативный выход. Обе формы записи начинаются с ключевого слова **Do** и заканчиваются словом **Loop**. Каждое логическое выражение начинается со слова **Wend**. При проверке условия в конце операторы тела цикла в отличие от формы проверки сверху, выполняются хотя бы один раз. В этом состоит основное различие двух форм записи условий (2, 3).

**Массивы**

В программировании часто возникает необходимость обработки большого количества однородных данных – массивов (последовательностей чисел, временных рядов, матриц и т.п.). *Массив* – это множество однотипных элементов, объединенных общим именем и занимающих в компьютере определенную область памяти. Элементы массива располагаются в последовательных ячейках памяти, обозначаются именем массива и индексом. Каждое из значений, составляющих массив, называется его *компонентой* (или *элементом* массива).

Массив данных а программе рассматривается как переменная структурированного типа. Массиву присваивается имя, посредством которого можно ссылаться как на массив данных в целом, так и на любую из его компонент. Индекс (порядковый номер элемента) записывается в скобках после имени массива.

**Например:** числовая последовательность четных натуральных чисел 2, 4, 6, …, *N* представляет собой линейный массив, элементы которого можно обозначить А(1) = 2, А(2) = 4. А(3) = 6, где 1, 2, 3 – номер элемента, а числа 2, 4, 6 – значения элементов массива.

Таким образом, для работы с массивами данных в программировании используются индексированные переменные. Для задания индексированных переменных в VBA применяется оператор **Dim (**от англ. Dimension – размерность).

Для ввода элементов массива можно использовать:

1) функцию **InputBox** (это особый тип процедуры VBA, возвращающая значение). Функция **InputBox** отображает диалоговое окно, содержащее текст, который запрашивает пользователя ввести некоторое значение и текстовое окно для ввода этого значения. **Например**:

Sub primer()

Dim A(5) As Integer

For I=1 To 5

A(I)=InputBox("Введите элемент массива","Пример")

Next I

. . . . . . . . . . . . . . ..

[продолжение программы]

2) функцию **Сells.** В этом случае пользователь перед запуском программы вводить данные элементов массива в ячейки рабочего листа Excel. **Например**:

Sub primer()

Dim A(5) As Integer

For I=1 To 5

A(I)=Cells(I, J) ‘ считывание значения из I – ой строки, J –го столбца

Next I

. . . . . . . . . . . . . . ..

[продолжение программы]

Этот способ более предпочтителен, поскольку он не требует ввода данных с клавиатуры каждый раз при новом появлении диалогового окна.

1. Можно воспользоваться стандартной функцией RND (если значение массива не задано):

Sub primer()

Randomize Iimer

Dim A(5) As Integer

For I=1 To 5

A(I)=Rnd

Next I

. . . . . . . . . . . . . . ..

[продолжение программы]

Во всех случаях, поскольку нам необходимо выполнять последовательность однотипных действий (присвоить значение переменной *A*(1), затем переменной *A*(2), затем *A*(3) и т.д.), заполнение массива проводится в цикле.

Ввод и вывод элементов одномерного массива осуществляется с помощью оператора цикла **For…Next**.

**Пример 6.** Ввести массив А из 10 элементов. Найти сумму элементов массива.

В этом примере используется характерный прием программирования – метод накопления: к переменной *S* последовательно прибавляются отдельные элементы массива; в результате при выходе из цикла переменная *S* будет содержать сумму всех перебранных элементов. Аналогично можно найти произведение всех элементов (в этом случае начальное значение *S* должно равняться не нулю, а единице (*S*=1).

Sub primer\_2()

Dim A(10) As Integer

Dim S As Integer

For i = 1 To 10

A(i) = Cells(1, i)

Next i

S = 0

For i = 1 To 10

S = S + A(i)

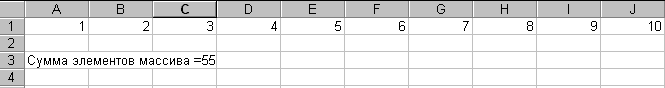
Next i

Cells(3, 1) = "Сумма элементов массива =" & S

'вывод суммы в 3 строку 1 столбец активного листа Excel

**End Sub**

Результат программы:



**Пример 7.** Ввести массив А из 10 элементов. Найти произведение элементов массива.

Sub primer\_3()

Dim A(10) As Integer

Dim Р As Integer

For i = 1 To 10

A(i) = Cells(1, i)

Next i

Р = 1

For i = 1 To 10

Р = Р \* A(i)

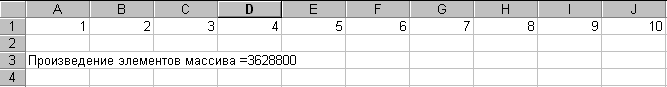
Next i

Cells(4, 1) = "Произведение элементов массива =" & Р

'вывод произведения в 4 строку 1 столбец активного листа Excel

End Sub

Результат программы:



**Пример 8**. Ввести массив А из 10 элементов. Найти максимальный элемент массива.

В этом примере переменная MAX используется для запоминания текущего максимального элемента с обновлением – в случае необходимости – по мере перебора всех элементов.

Sub primer\_4()

Dim A(8) As Integer

For i = 1 To 8

A(i) = Cells(1, i)

Next i

MAX = A(1)

For i = 2 To 8

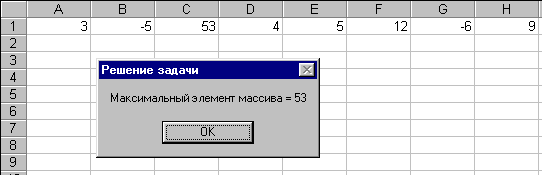
If A(i) > MAX Then MAX = A(i)

Next i

MsgBox "Максимальный элемент массива = " & MAX, , "Решение задачи"

End Sub

Результат программы:



**Создание новых массивов при решении задач**

Часто при решении практических задач возникает необходимость создания нового массива из выборочных элементов исходного массива. Особенностью этого типа задач является то, что индексы элементов этих массивов не совпадают. Следовательно, в цикле должны изменяться два параметра и необходимо использовать прием организации цикла с несколькими одновременно изменяющимися параметрами. Обычно в блоке решения перед циклом начальные значения индексов новых массивов принимаются равными нулю. Далее в цикле по мере отбора элементов в новый массив вычисляется текущее значение этих индексов, которые и присваиваются соответствующим элементам.

**Пример 9**. Дан вещественный массив *А(N).* Вывести отдельно массив целых чисел (если они есть в исходном массиве) и массив вещественных чисел.

Решение. Для фиксации текущих значений индексов в новых массивах введем счетчик *к* для массива *B* и счетчик *q*  для массива *D*. При нахождении элементов массива соответствующих первому условию задачи увеличим значение *к* на единицу и присвоим этот индекс первому элементу нового массива *B* и т.п.

Аналогичная процедура повторяется и при выполнении второго условия задачи, где формируется еще один массив *D.*

Sub primer\_9()

Dim A(10), B(10), D(10) As Single

For i = 1 To 10 ' ввод массива

A(i) = Cells(1, i)

Next i

For i = 1 To 10 ' вывод начального массива

Cells(2, i) = A(i)

Next i

к = 0: q = 0 'начальные значения индексов новых массивов

For i = 1 To 10

If Abs(A(i)) - Int(A(i)) = 0 Then

к = к + 1: B(к) = A(i) ' вычисление текущего индекса массива В

Else

q = q + 1: D(q) = A(i) 'запись элемента А(i) в новый массив D

End If

Next i

If к = 0 Then

Cells(3, 1) = "В массиве целых чисел нет"

Else

Cells(3, 1) = "Массив целых чисел B(к))"

For i = 1 To к

Cells(4, i) = B(i)

Next i

End If

If q = 0 Then

Cells(5, 1) = "В массиве дробных чисел нет"

Else

Cells(5, 1) = "Массив вещественных чисел D(q)"

For i = 1 To q

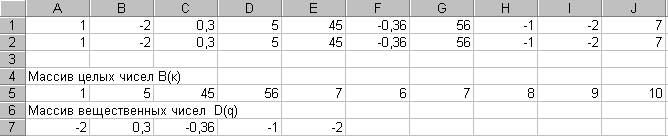
Cells(6, i) = D(i)

Next i

End If

End Sub

Результата программы:



## Динамические массивы

Иногда бывает трудно заранее определить, сколько элементов нужно будет поместить в список. Для этого используют динамические массивы.

*Динамические массивы* используются в том случае, когда количество элементов массива заранее неизвестно и будет определяться в процессе выполнения программы. Как следствие, после того, как работа с динамическими массивом в программе будет выполнена, можно освободить память, которую он занимает.

Динамический массив объявляется так же, как и любой другой тип массива – оператором **Dim**. Единственное отличие состоит в том, что вы не указываете (в скобках) размер массива. Например:

Dim A() As Byte ‘ объявление динамического массива с указанием типа массива

Этой строкой мы создаем массив, но откладываем решение о том, сколько он будет иметь элементов. Перед тем, как использовать этот массив, нам необходимо будет указать его размер, который во время работы программы мы можем изменять сколько угодно раз. Это делается с помощью оператора **ReDim**. Например:

ReDim A(5) ‘ указание размерности динамического массива

Следует отметить, что при указании значений для размерностей динамического массива оператором **ReDim** его тип не может быть изменен. В противном случае будет выдано сообщение об ошибке.

При помощи оператора **ReDim** можно установить устанавливать любые значения границ и количество размерностей, а также менять любой из установленных параметров как в сторону уменьшения, так и увеличения. Например:

Dim A() As Byte

ReDim A() (Диапазон от 0 до 5)

ReDim A(1 to 4) (Диапазон от 1 до 4)

ReDim A(3,3, 1 to 3)(Диапазон от 0 до 3, от 0 до 3, от 1 до 3)

В качестве как верхней, так и нижней границы могут быть использованы не только числа, но и переменный целого типа, например:

Dim n As Integer

Dim A() As Integer

N=10

ReDim A(n)

Необходимо иметь ввиду, что при каждом выполнении оператора **ReDim** все значения элементов массива, которые до этого в нем хранились, будут потеряны, так как данный оператор обнуляет все элементы в соответствии с их типом, например, числовые массивы принимают значение 0, а символьные “” (пусто).

Для того, чтобы имеющиеся в массиве значения не пропали при его переопределении, следует использовать служебное слово Preserve, например:

Sub primer()

Dim n As Integer

Dim I As Integer

Dim A() As Integer

N=10

ReDim A() As Integer

For i=1 to n

A(i)=i^2

Next i

ReDim Preserve A(1 To 15)

For i=11 to 15

A(i)=i^3

Next i

End Sub

В данной программе требуется заполнить массив из десяти элементов квадратами первых десяти натуральных чисел. Затем следует переопределение размера массива до 15 и заполнение новых элементов (с 11 по 15) кубами следующих натуральных чисел.

**Пример 10.** Заполнить массив *А(N)* нечетными членами натурального ряда (1, 3, 5...) до тех пор, пока их произведение не станет больше произвольного числа *М*. Найти сумму элементов массива и их число.

Sub primer\_10()

Dim A() As Integer

Dim M As Integer, N As Integer

Dim S As Integer, Pr As Integer

M = InputBox("Введите число М", "Пример 1")

N = 100

ReDim A(N)

A(1) = 1: i = 2

Cells(1, 1) = A(1)

S = 0: Pr = 1 ' начальные значения суммы и произведения

k = 1 ' счетчик количества элементов

Do While Pr <= M ' начало цикла, проверка условия

k = k + 1 ' текущее количество элементов

A(i) = A(i - 1) + 2

Pr = Pr \* A(i) ' вычисление произведения

i = i + 1

Loop ' конец цикла

For i = 1 To k - 1 ' вывод массива

Cells(1, i) = A(i) ' заполнение массива элементами

S = S + A(i) ' накопление суммы элементов

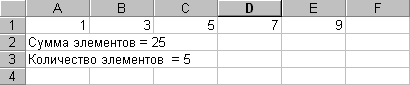
Next i

Cells(2, 1) = "Сумма элементов = " & S

Cells(3, 1) = "Количество элементов = " & k - 1

End Sub

Результат программы:



**Операции с матрицами**

Нередко приходится иметь дело с различными данными, которые должны обрабатываться одинаковым образом. Если для каждого значения отвести свою переменную, то и обрабатывать их придется индивидуально. Если, например, необходимо обработать таблицу, состоящую из нескольких сотен ячеек, то создание такой программы практически невозможно. Решить эту проблему позволяют *массивы*. *Массив* представляет собой набор однотипных переменных, объединенных одним именем и различающихся по числовому индексу. Этот индекс записывается после имени переменной в скобках. Обращение к элементам массива осуществляется путем указания индексов элемента. Количество индексов определяет размерность массива. Любая таблица чисел или символов по определению является *двумерным массивом*. Двумерный массив *А*(*N, M*) соответствует понятию прямоугольной матрицы (таблицы), состоящей из *m* строк и *n* столбцов.



Другим примером двумерных массивов могут быть любые данные (числа или символы), представленные в табличной форме, т.е. любая таблица чисел или символов по определению является двумерным массивом.

В редакторе Visual Basic for Application двумерный массив задается с помощью оператора **Dim**:

Dim A(N,M)

Здесь оператором **Dim** переменная *А* объявляется двумерным масси­вом размерностью *N*x*M*, где переменная *N* определяет количество строк (первый индекс массива), а переменная *М* – количество столбцов (второй ин­декс). Значения переменных *N* и *М*, определяющие размер массива, должны быть заданы до оператора **Dim**. Сам оператор **Dim** должен находится до пер­вого обращения к элементам массива, т.е. как правило – в начале программы. Размер массива можно задать также явно, если вместо переменных *N* и *М* по­ставить конкретные числа, например

DIM A(10, 5)

Так будет объявлен массив, состоящий из 10 строк и 5 столбцов.

В качестве имени массива можно использовать любое допустимое на языке VBA имя переменной. В одном операторе **Dim** можно объявить несколько массивов. Можно также использовать несколько операторов **Dim**.

Элементами массива могут быть не только числа (целые или вещественные), но и символьные данные. В этом случае в команду описания массива следует вставить определение типа, например

Dim A( 10, 5) As String,

где часть As String, говорит о том, что элементы массива представляют собой символьные (строковые) переменные.

Массив может быть не только двумерным, но трехмерным, четырехмерным и большей размерности (до 8 индексов). Длина массива (количество элементов по каждому индексу) может быть разной (в несколько тысяч элементов) и ограничивается только размером памяти компьютера, отводимых под данные.

Поскольку элементы двумерно­го массива задаются двумя индексами – номером строки и номером столбца – для перебора всех строк и столбцов необходимо использование двойного цикла, один из которых будет внешним, а другой – внутренним. Такие циклы называются *вложенными.*

Присвоить значения элементам двумерного массива можно теми же способами, что и для одномерного массива.

1. С помощью функции **InputBox**. **Например**:

Sub primer()

Dim A (3,4) As Integer

For I = 1 To 3 ‘ цикл по строкам

For J = 1 To 4 ‘ цикл по столбцам

A(I, J) = InputBox("Введите элемент массива", "Пример")

Next J

Next I

. . . . . . . . . . . . . ..

[продолжение программы]

Однако если массив большой, эта процедура становится долгой и неудобной, так как он требует ввода данных с клавиатуры каждый раз при новом появлении диалогового окна.

1. С помощью функции **Cells. Например:**

Sub primer()

Dim A (3,4) As Integer

For I = 1 To 3 ‘ цикл по строкам

For J = 1 To 4 ' цикл по столбцам

A(I, J) = Cells (I, J)

Next J

Next I

. . . . . . . . . . . . . . ..

[продолжение программы]

Этот способ наиболее удобен и имеет важное преимущество – данные не теряются при повторном выполнении программы. Пользователь вводит значения элементов массива в ячейки рабочего листа Excel, из которого после запуска программы VBA считывает заданные значения и присваивает очередной переменной текущее значение из списка.

1. Можно воспользоваться стандартной функцией RND (если значение массива не задано):

Sub primer()

Dim A(3, 4) As Single

For I = 1 To 3

For J = 1 To 4

A(I, J) = Rnd

Next J

Next I

. . . . . . . . . . . . . . ..

[продолжение программы]

Ввод, а так же вывод двумерного массива, производится с использованием двойного цикла. В зависимости от условий задачи ввод и вывод можно организовывать по строкам или по столбцам. При вводе и выводе по строкам параметром внешнего цикла является индекс обозначающий номер строки, параметром внутреннего цикла – индекс (номер) столбца. При вводе по столбцам – внешний цикл по столбцам, внутренний – по строкам.

Итак, ввод и выгод элементов двумерного массива осуществляется с помощью двух вложенных циклов For. Таким же образом (с помощью вложенного цикла) можно проводить обработку массива – находить максимальный (минимальный) элемент, вычислять сумму элементов и т.д. (см. раздел «Массивы»). Все это можно делать отдельно для каждой строки (столбца) или для всех элементов данного массива.

Двумерные массивы, у которых значения индексов одинаковые, т.е. число строк равно числу столбцов, называют *квадратной матрицей*. При работе с квадратными матрицами используются специальные математические понятия, значение которых рассмотрим на примере матрицы размерами 4х4 – *А*(4,4):



В зависимости от соотношения между первым и вторым индексами элемента определяется расположение этого элемента в матрице, если:

1) *i* = *j* – элемент находится на главной диагонали ();

2) *i* + *j* = *n* + 1 (*n* – размерность матрицы) – элемент находится на второй главной диагонали ();

3) *i* < *j* – элемент находится над главной диагональю ();

4) *i* > *j* – элемент находится под главной диагональю ().

*Транспонированной матрицей* *В(N,N)* называется такая квадратная матрица, у которой столбцы соответствуют строкам исходной квадратной матрицы *А(N,N)*:



Элементы главной диагонали у матриц *А* и *В* одни и те же. Следовательно операция транспортирования матрицы *А* сводится к перестановке строк матрицы *А* на столбцы матрицы *В*. При этом диагональные элементы у матриц *А* и *В* одни и те же. Это реализуется с помощью соотношения: *В*(*j, i*) = *А*(*i, j*).

При работе с двумерными массивами наиболее часто встречаются задачи какой-либо обработки элементов массива, расположенных в строках или столбцах. Такие задачи решаются конкретным обращением к соответствующим элементам массива, т.е. перебором индексов в определенной последовательности.

**Пример 11**.Ввести массив *А*(*N*, *N*). Транспонировать эту матрицу, т.е. заменить строки столбцами. Вывести на печать исходный и преобразованный массивы в виде матриц.

Sub primer\_11()

N = 3

ReDim A(N, N), B(N, N) As Integer

For I = 1 To N

For J = 1 To N

A(I, J) = Cells(I, J)

B(J, I) = A(I, J) 'транспонирование матрицы

Next J

Next I

For I = 1 To N

For J = 1 To N

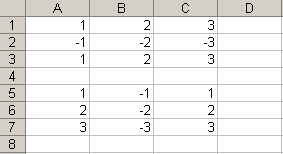
Cells(N + 1 + I, J) = B(I, J) ‘вывод преобразованной матрицы

Next J

Next I

End Sub

Результат программы:



**Пример 12**. Дан массив *А*(*N, М*). Составить программу для подсчета количества положительных элементов массива.

Sub primer\_12()

N = 3: M = 3

ReDim A(N, N) As Integer

For I = 1 To N

For J = 1 To M

A(I, J) = Cells(I, J)

Next J

Next I

k = 0 ‘ счетчик числа положительных элементов

For I = 1 To N

For J = 1 To M

If A(I, J) > 0 Then k = k + 1:

Next J

Next I

If k = 0 Then

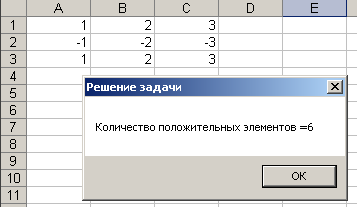
MsgBox "Положительных элементов нет", , "Решение задачи"

Else: MsgBox "Количество положительных элементов =" & k, , "Решение задачи"

End If

End Sub

Результат программы:



**Пример 13**. Дана квадратная матрица *D* (*N*, *N*). Вычислить суммы элементов в каждом столбце и произведение элементов главной диагонали.

Решение. Поскольку необходимо получить суммы элементов в столбцах, то после ввода значений элементов массива, на этапе решения для накопления суммы внешний цикл открывается по индексу столбца, а во внутреннем цикле (индекс по строке) перебираются все элементы сначала первого столбца, затем второго и т.д. При этом подсчитывается сумма по каждому столбцу. Одновременно во внутреннем цикле каждый элемент матрицы проверяется на принадлежность к главной диагонали, для которых подсчитывается произведение.

Sub primer\_13()

N = 3

ReDim D(N, N), S(N) As Integer

For I = 1 To N ' ввод элементов массива D(N,N)

For J = 1 To N

D(I, J) = Cells(I, J)

Next J, I 'одновременное закрытие вложенных циклов

Pr = 1

For J = 1 To N

S(J) = 0

For I = 1 To N

S(J) = S(J) + D(I, J) 'накопление суммы по столбцам

If I = J Then Pr = Pr \* D(I, J)

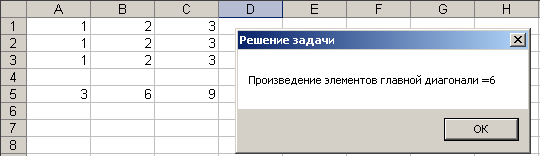
Next I 'закрытие внутреннего цикла Cells(5, J) = S(J)

Next J ' закрытие внешнего цикла

MsgBox "Произведение элементов главной диагонали =" & Pr, , "Решение задачи"

End Sub

Результат программы:



**ПРАВИЛА ВЫПОЛНЕНИЯ И ОФОРМЛЕНИЯ**

**КОНТРОЛЬНОЙ РАБОТЫ**

При выполнении контрольных работ необходимо придерживаться указанных ниже правил. Работы, выполненные без соблюдения этих правил, не зачитываются и возвращаются студенту для переработки.

1. Каждая контрольная работа должна быть выполнена в печатной форме в текстовом редакторе WORD (формат А4) или в отдельной тетради в клетку чернилами любого цвета, кроме красного. Пример оформления титульного листа показан в Приложении 1.

2. В работу должны быть включены все задачи, указанные в задании строго по положенному варианту. Контрольные работы, содержащие не все задачи задания, а также задачи не своего варианта, не зачитываются.

4. Решение задач надо располагать в порядке номеров, указанных в заданиях, сохраняя номера задач.

5. Перед решением каждой задачи надо полностью выписать условие.

6. Отчет должен содержать скан с экрана рабочего листа книги Excel (Приложение 2) или код программы, скопированный из VBA и результаты решения задачи, скопированные с рабочего листа Excel (Приложение 3).

7. Сдача контрольной работы проходит на лабораторной работе.

**ТАБЛИЦА ВАРИАНТОВ И ЗАДАЧИ**

**ДЛЯ ВЫПОЛНЕНИЯ КОНТРОЛЬНОЙ РАБОТЫ**

Каждый студент выполняет вариант задания, обозначенный двумя последними цифрами его шифра (номера в зачетной книжке).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № варианта | Номера задач | | | | | |
|  |  | 31 | 61 | 91 | 121 | 151 |
|  |  | 32 | 62 | 92 | 122 | 152 |
|  |  | 33 | 63 | 93 | 123 | 153 |
|  |  | 34 | 64 | 94 | 124 | 154 |
|  |  | 35 | 65 | 95 | 125 | 155 |
|  |  | 36 | 66 | 96 | 126 | 156 |
|  |  | 37 | 67 | 97 | 127 | 157 |
|  |  | 38 | 68 | 98 | 128 | 158 |
|  |  | 39 | 69 | 99 | 129 | 159 |
|  |  | 40 | 70 | 100 | 130 | 160 |
|  |  | 41 | 71 | 101 | 131 | 161 |
|  |  | 42 | 72 | 102 | 132 | 162 |
|  |  | 43 | 73 | 103 | 133 | 163 |
|  |  | 44 | 74 | 104 | 134 | 164 |
|  |  | 45 | 75 | 105 | 135 | 165 |
|  |  | 46 | 76 | 106 | 136 | 166 |
|  |  | 47 | 77 | 107 | 137 | 167 |
|  |  | 48 | 78 | 108 | 138 | 168 |
|  |  | 49 | 79 | 109 | 139 | 169 |
|  |  | 50 | 80 | 110 | 140 | 170 |
|  |  | 51 | 81 | 111 | 141 | 171 |
|  |  | 52 | 82 | 112 | 142 | 172 |
|  |  | 53 | 83 | 113 | 143 | 173 |
|  |  | 54 | 84 | 114 | 144 | 174 |
|  |  | 55 | 85 | 115 | 145 | 175 |
|  |  | 56 | 86 | 116 | 146 | 176 |
|  |  | 57 | 87 | 117 | 147 | 177 |
|  |  | 58 | 88 | 118 | 148 | 178 |
|  |  | 59 | 89 | 119 | 149 | 179 |
|  |  | 60 | 90 | 120 | 150 | 180 |

1. Ввести массив А(N). Найти среднее геометрическое значение массива. Четные элементы массива разделить на среднее геометрическое. Вывести преобразованный массив.
2. Ввести массив А(N). Найти среднее геометрическое значение массива. Нечетные элементы массива разделить на среднее геометрическое. Вывести преобразованный массив.
3. Ввести массив А(N). Найти среднее геометрическое значение массива. Все элементы массива разделить на среднее геометрическое. Вывести преобразованный массив.
4. Ввести массив А(N). Найти среднее геометрическое значение массива. Максимальный элемент массива разделить на среднее геометрическое. Вывести преобразованный массив.
5. Ввести массив А(N). Найти среднее геометрическое значение массива. Минимальный элемент массива разделить на среднее геометрическое. Вывести преобразованный массив.
6. Ввести массив А(N). Найти среднее геометрическое значение массива. Четные элементы массива заменить на среднее геометрическое. Вывести преобразованный массив.
7. Ввести массив А(N). Найти среднее геометрическое значение массива. Нечетные элементы массива заменить на среднее геометрическое. Вывести преобразованный массив.
8. Ввести массив А(N). Найти среднее геометрическое значение массива. Максимальный элемент массива заменить на среднее геометрическое. Вывести преобразованный массив.
9. Ввести массив А(N). Найти среднее геометрическое значение массива. Минимальный элемент массива заменить на среднее геометрическое. Вывести преобразованный массив.
10. Ввести массив А(N). Найти среднее геометрическое значение массива. Первый элемент массива заменить на среднее геометрическое. Вывести преобразованный массив.
11. Ввести массив А(N). Найти среднее геометрическое значение массива. Четных элементы массива уменьшить на среднее геометрическое. Вывести преобразованный массив.
12. Ввести массив А(N). Найти среднее геометрическое значение массива. Нечетные элементы массива уменьшить на среднее геометрическое. Вывести преобразованный массив.
13. Ввести массив А(N). Найти среднее геометрическое значение массива. Все элементы массива уменьшить на среднее геометрическое. Вывести преобразованный массив.
14. Ввести массив А(N). Найти среднее геометрическое значение массива. Найти разность максимального элемента массива и среднего геометрического.
15. Ввести массив А(N). Найти среднее геометрическое значение массива. Найти разность минимального элемента массива и среднего геометрического.
16. Ввести массив А(N). Найти среднее геометрическое значение массива. Четные элементы массива увеличить на среднее геометрическое. Вывести преобразованный массив.
17. Ввести массив А(N). Найти среднее геометрическое значение массива. Нечетные элементы массива увеличить на среднее геометрическое. Вывести преобразованный массив.
18. Ввести массив А(N). Найти среднее геометрическое значение массива. Все элементы массива увеличить на среднее геометрическое. Вывести преобразованный массив.
19. Ввести массив А(N). Найти среднее геометрическое значение массива. Максимальный элемент массива увеличить на среднее геометрическое. Вывести преобразованный массив.
20. Ввести массив А(N). Найти среднее геометрическое значение массива. Минимальный элемент массива увеличить на среднее геометрическое. Вывести преобразованный массив.
21. Ввести массив А(N). Найти среднее арифметическое элементов массива. Четные элементы массива разделить на среднее арифметическое. Вывести преобразованный массив.
22. Ввести массив А(N). Найти среднее арифметическое значение массива. Нечетные элементы массива разделить на среднее арифметическое. Вывести преобразованный массив.
23. Ввести массив А(N). Найти среднее арифметическое значение массива. Все элементы массива разделить на среднее арифметическое. Вывести преобразованный массив.
24. Ввести массив А(N). Найти среднее арифметическое значение массива. Максимальный элемент массива разделить на среднее арифметическое. Вывести преобразованный массив.
25. Ввести массив А(N). Найти среднее арифметическое значение массива. Минимальный элемент массива разделить на среднее арифметическое. Вывести преобразованный массив.
26. Ввести массив А(N). Найти среднее арифметическое значение массива. Четные элементы массива заменить на среднее арифметическое. Вывести преобразованный массив.
27. Ввести массив А(N). Найти среднее арифметическое значение массива. Нечетные элементы массива заменить на среднее арифметическое. Вывести преобразованный массив.
28. Ввести массив А(N). Найти среднее арифметическое значение массива. Максимальный элемент массива заменить на среднее арифметическое. Вывести преобразованный массив.
29. Ввести массив А(N). Найти среднее арифметическое значение массива. Минимальный элемент массива заменить на среднее арифметическое. Вывести преобразованный массив.
30. Ввести массив А(N). Найти среднее арифметическое значение массива. Первый элемент массива заменить на среднее арифметическое. Вывести преобразованный массив.
31. Ввести массив А(N). Найти среднее арифметическое значение массива. Четные элементы массива уменьшить на среднее арифметическое. Вывести преобразованный массив.
32. Ввести массив А(N). Найти среднее арифметическое значение массива. Нечетных элементы массива уменьшить на среднее арифметическое. Вывести преобразованный массив.
33. Ввести массив А(N). Найти среднее арифметическое значение массива. Все элементы массива уменьшить на среднее арифметическое. Вывести преобразованный массив.
34. Ввести массив А(N). Найти среднее арифметическое значение массива. Найти разность максимального элемента массива и среднего арифметического.
35. Ввести массив А(N). Найти среднее арифметическое значение массива. Найти разность минимального элемента массива и среднего арифметического.
36. Ввести массив А(N). Найти среднее арифметическое значение массива. Четные элементы массива увеличить на среднее арифметическое. Вывести преобразованный массив.
37. Ввести массив А(N). Найти среднее арифметическое значение массива. Нечетные элементы массива увеличить на среднее арифметическое. Вывести преобразованный массив.
38. Ввести массив А(N). Найти среднее арифметическое значение массива. Все элементы массива увеличить на среднее арифметическое. Вывести преобразованный массив.
39. Ввести массив А(N). Найти среднее арифметическое значение массива. Максимальный элемент массива увеличить на среднее арифметическое. Вывести преобразованный массив.
40. Ввести массив А(N). Найти среднее арифметическое значение массива. Минимальный элемент массива увеличить на среднее арифметическое. Вывести преобразованный массив.
41. Ввести массив А(N). Найти максимальный элемент массива. Четные элементы массива разделить на максимальный элемент. Вывести полученный массив.
42. Ввести массив А(N). Найти максимальный элемент массива. Нечетные элементы массива разделить на максимальный элемент. Вывести полученный массив.
43. Ввести массив А(N). Найти максимальный элемент массива. Все элементы массива разделить на максимальный элемент. Вывести полученный массив.
44. Ввести массив А(N). Найти максимальный элемент массива. Четные элементы массива заменить на максимальный элемент. Вывести полученный массив.
45. Ввести массив А(N). Найти максимальный элемент массива. Нечетные элементы массива заменить на максимальный элемент. Вывести полученный массив.
46. Ввести массив А(N), N>5. Найти максимальный элемент массива. Первые пять элементов массива заменить на максимальный элемент. Вывести полученный массив.
47. Ввести массив А(N), N>5. Найти максимальный элемент массива. Последние пять элементов массива заменить на максимальный. Вывести полученный массив.
48. Ввести массив А(N). Найти максимальный элемент массива. Четные элементы массива уменьшить на максимальный. Вывести полученный массив.
49. Ввести массив А(N). Найти максимальный элемент массива. Нечетные элементы массива уменьшить на максимальный. Вывести полученный массив.
50. Ввести массив А(N). Найти максимальный элемент массива. Все элементы массива уменьшить на максимальный. Вывести полученный массив.
51. Ввести массив А(N). Найти максимальный элемент массива. Среднее геометрическое элементов массива уменьшить на максимальный элемент массива.
52. Ввести массив А(N). Найти максимальный элемент массива. Среднее арифметическое элементов массива уменьшить на максимальный элемент массива.
53. Ввести массив А(N). Найти максимальный элемент массива. Четные элементы массива увеличить на максимальный. Вывести полученный массив.
54. Ввести массив А(N). Найти максимальный элемент массива. Нечетные элементы массива увеличить на максимальный. Вывести полученный массив.
55. Ввести массив А(N). Найти максимальный элемент массива. Все элементы массива увеличить на максимальный. Вывести полученный массив.
56. Ввести массив А(N), N>3. Найти максимальный элемент массива. Первые три элемента массива уменьшить на максимальный. Вывести полученный массив.
57. Ввести массив А(N), N>5. Найти максимальный элемент массива. Последние пять элементов массива увеличить на максимальный. Вывести полученный массив.
58. Ввести массив А(N). Найти минимальный элемент массива. Четные элементы массива разделить на минимальный. Вывести полученный массив.
59. Ввести массив А(N). Найти минимальный элемент массива. Нечетные элементы массива разделить на минимальный. Вывести полученный массив.
60. Ввести массив А(N). Найти минимальный элемент массива. Все элементы массива разделить на минимальный. Вывести полученный массив.
61. Ввести массив А(N). Найти минимальный элемент массива. Четные элементы массива заменить на минимальный. Вывести полученный массив.
62. Ввести массив А(N). Найти минимальный элемент массива. Нечетные элементы массива заменить на минимальный. Вывести полученный массив.
63. Ввести массив А(N), N>5.Найти минимальный элемент массива. Первые три элемента массива заменить на минимальный. Вывести полученный массив.
64. Ввести массив А(N), N>5. Найти минимальный элемент массива. Последние пять элементов массива заменить на минимальный. Вывести полученный массив.
65. Ввести массив А(N). Найти минимальный элемент массива. Четные элементы массива уменьшить на минимальный. Вывести полученный массив.
66. Ввести массив А(N). Найти минимальный элемент массива. Нечетные элементы массива уменьшить на минимальный. Вывести полученный массив.
67. Ввести массив А(N). Найти минимальный элемент массива. Все элементы массива уменьшить на минимальный. Вывести полученный массив.
68. Ввести массив А(N). Найти минимальный элемент массива, среднее геометрическое элементов массива уменьшить и разность среднего геометрического и минимального элемент массива. Вывести все результаты.
69. Ввести массив А(N). Найти минимальный элемент массива, среднее арифметическое элементов массива и разность среднего арифметического и минимального элемент массива. Вывести се результаты.
70. Ввести массив А(N). Найти минимальный элемент массива. Четные элементы массива увеличить на минимальный. Вывести полученный массив.
71. Ввести массив А(N). Найти минимальный элемент массива. Нечетные элементы массива увеличить на минимальный. Вывести полученный массив.
72. Ввести массив А(N). Найти минимальный элемент массива. Все элементы массива увеличить на минимальный. Вывести полученный массив.
73. Ввести массив А(N). Найти минимальный, максимальный элементы массива и частное от их деления. Вывести все результаты.
74. Ввести массив А(N). Найти минимальный, максимальный элементы массива и их сумму. Вывести все результаты.
75. Ввести массив А(N). Найти минимальный, максимальный элементы массива и их произведение. Вывести все результаты.
76. Ввести массив А(N), N>5. Найти минимальный элемент массива. Первые четыре элемента массива увеличить на минимальный. Вывести полученный массив.
77. Ввести массив А(N), N>5. Найти минимальный элемент массива. Последние пять элементов массива увеличить на минимальный. Вывести полученный массив.
78. Ввести массив А(N). Четные элементы массива разделить на сумму всех элементов массива. Вывести преобразованный массив.
79. Ввести массив А(N). Нечетные элементы массива разделить на сумму всех элементов массива. Вывести преобразованный массив.
80. Ввести массив А(N). Все элементы массива разделить на сумму элементов массива. Вывести преобразованный массив.
81. Ввести массив А(N). Найти частное от деления суммы элементов массива и максимального элемента массива.
82. Ввести массив А(N). Найти частное от деления суммы элементов массива и минимального элемента массива.
83. Ввести массив А(N). Найти частное от деления суммы элементов массива и минимального элемента массива.
84. Ввести массив А(N). Четные элементы массива заменить на сумму элементов массива. Вывести сумму и преобразованный массив.
85. Ввести массив А(N). Нечетные элементы массива заменить на сумму элементов массива. Вывести сумму и преобразованный массив.
86. Ввести массив А(N). Максимальный элемент массива заменить на сумму элементов массива. Вывести максимальный элемент и преобразованный массив.
87. Ввести массив А(N). Минимальный элемент массива заменить на сумму элементов массива. Вывести максимальный элемент и преобразованный массив.
88. Ввести массива А(N). Найти сумму элементов массива. Нечетные элементы массива уменьшить на сумму. Вывести сумму и преобразованный массив.
89. Ввести массива А(N). Найти сумму элементов массива. Четные элементы массива уменьшить на сумму. Вывести сумму и преобразованный массив.
90. Ввести массива А(N). Найти сумму элементов массива, максимальный элемент массива. Найти разность суммы и максимального элемента.
91. Ввести массива А(N). Найти сумму элементов массива, минимальный элемент массива. Найти разность суммы и минимального элемента.
92. Ввести массив А(N). Найти сумму элементов массива. Четные элементы массива увеличить на сумму. Вывести полученный массив.
93. Ввести массив А(N). Найти сумму элементов массива. Нечетные элементы массива увеличить на сумму. Вывести полученный массив.
94. Ввести массив А(N). Найти сумму элементов массива. Каждый элементов массива увеличить на сумму. Вывести полученный массив.
95. Ввести массив А(N). Найти частное от деления суммы четных элементов массива и максимального элемента. Вывести сумму, максимум и частное от деления.
96. Ввести массив А(N). Найти частное от деления суммы нечетных элементов массива и максимального элемента. Вывести сумму, максимум и частное от деления.
97. Ввести массив А(N). Найти частное от деления суммы четных элементов массива и минимального элемента. Вывести сумму, минимум и частное от деления.
98. Ввести массив А(N). Найти частное от деления суммы нечетных элементов массива и минимального элемента. Вывести сумму, минимум и частное от деления.
99. Ввести массив А(N). Найти сумму четных элементов массива. Четные элементы массива разделить на сумму. Вывести преобразованный массив.
100. Ввести массив А(N). Найти сумму нечетных элементов массива. Нечетные элементы массива разделить на сумму. Вывести преобразованный массив.
101. Ввести массив А(N). Найти сумму нечетных элементов массива. Четные элементы массива разделить на сумму. Вывести преобразованный массив.
102. Ввести массив А(N). Найти сумму четных элементов массива. Нечетные элементы массива разделить на сумму. Вывести преобразованный массив.
103. Ввести массив А(N). Найти максимальный элемент и сумму нечетных элементов массива. Найти частное от деления суммы и максимума.
104. Ввести массив А(N). Найти минимальный элемент и сумму нечетных элементов массива. Найти частное от деления суммы и минимума.
105. Ввести массив А(N). Найти максимальный элемент и сумму четных элементов массива. Найти частное от деления суммы и максимума.
106. Ввести массив А(N). Найти минимальный элемент и сумму четных элементов массива. Найти частное от деления суммы и минимума.
107. Ввести массив А(N), N>5. Найти произведение первых трёх элементов массива, сумму остальных и разность произведения и суммы. Вывести все результаты.
108. Ввести массив А(N), N>5. Найти произведение первых пяти элементов массива, сумму остальных и частное от деления произведения и суммы. Вывести все результаты.
109. Ввести массив А(N). Найти сумму всех элементов, произведение четных элементов массива и частное от деления произведения и суммы. Вывести все результаты.
110. Ввести массив А(N). Найти сумму всех элементов, произведение нечетных элементов массива и частное от деления произведения и суммы. Вывести все результаты.
111. Ввести массив А(N). Найти сумму нечетных элементов, произведение четных элементов массива и частное от деления произведения и суммы. Вывести все результаты.
112. Ввести массив А(N). Найти сумму четных элементов, произведение нечетных элементов массива и частное от деления произведения и суммы. Вывести все результаты.
113. Ввести массив А(N). Найти максимальный элемент массива, произведение нечетных элементов массива. Заменить максимум на произведение. Вывести полученный массив и произведение.
114. Ввести массив А(N). Найти минимальный элемент массива, произведение нечетных элементов массива. Заменить минимум на произведение. Вывести полученный массив и произведение.
115. Ввести массив А(N). Найти максимальный элемент массива, произведение нечетных элементов массива и частное от деления произведения и максимума. Вывести все результаты.
116. Ввести массив А(N). Найти максимальный элемент массива, произведение четных элементов массива и частное от деления произведения и максимума. Вывести все результаты.
117. Ввести массив А(N). Найти минимальный элемент массива, произведение нечетных элементов массива и частное от деления произведения и минимума. Вывести все результаты.
118. Ввести массив А(N). Найти минимальный элемент массива, произведение четных элементов массива и частное от деления произведения и максимума. Вывести все результаты.
119. Ввести массив А(N). Найти произведение и сумму четных элементов массива, разность произведения и суммы. Вывести все результаты.
120. Ввести массив А(N). Найти произведение четных элементов массива, сумму нечетных элементов, разность произведения и суммы. Вывести все результаты.
121. Ввести массив А(N). Найти произведение нечетных элементов массива, сумму четных элементов, разность произведения и суммы. Вывести все результаты.
122. Ввести массив А(N). Найти произведение и сумму нечетных элементов массива, разность произведения и суммы. Вывести все результаты.
123. Ввести массив А(N). Найти минимальный элемент массива. Заменить минимум на произведение элементов массива. Вывести преобразованный массив.
124. Ввести массив А(N). Найти произведение элементов массива. Четные элементы массива заменить на произведение. Вывести произведение и преобразованный массив.
125. Ввести массив А(N). Найти произведение элементов массива. Нечетные элементы массива заменить на произведение. Вывести произведение и преобразованный массив.
126. Ввести массив А(N). Найти сумму положительных, сумму отрицательных элементов массива и частное от деления положительных и отрицательных. Вывести все результаты.
127. Ввести массив А(N). Найти сумму элементов массива, сумму отрицательных элементов и частное от их деления. Вывести все результаты.
128. Ввести массив А(N). Найти сумму положительных элементов массива. Положительные элементы массива заменить на сумму. Вывести сумму и преобразованный массив.
129. Ввести массив А(N). Найти сумму положительных элементов массива. Отрицательные элементы массива заменить на сумму. Вывести сумму и преобразованный массив.
130. Ввести массив А(N). Найти сумму отрицательных элементов массива. Положительные элементы массива заменить на сумму. Вывести сумму и преобразованный массив.
131. Ввести массив А(N). Найти сумму отрицательных элементов массива. Отрицательные элементы массива заменить на сумму. Вывести сумму и преобразованный массив.
132. Ввести массив А(N). Найти произведение элементов массива, произведение положительных элементов и частное от их деления. Вывести все результаты.
133. Ввести массив А(N). Найти произведение элементов массива. Положительные элементы массива заменить на произведение. Вывести произведение и преобразованный массив.
134. Ввести массив А(N). Найти произведение элементов массива. Отрицательные элементы массива заменить на произведение. Вывести произведение и преобразованный массив.
135. Ввести массив А(N). Найти сумму элементов массива. Положительные элементы массива заменить на сумму. Вывести сумму и преобразованный массив.
136. Ввести массив А(N). Задать число L. Вывести на печать исходный массив, значение первого элемента массива больше L, число элементов массива больших L.
137. Ввести массив А(N). Задать число L. Все элементы массива, большие L заменить на L. Вывести на печать исходный и преобразованный массивы.
138. Ввести массив А(N). Задать число L. Вывести на печать данные о количестве элементов массива равных L. Если таковых нет, вывести на печать сообщение об этом.
139. Ввести массив А(N). Элементы с индексами от 0 до N/2 увеличить в два раза. Остальные оставить неизменными. Вывести на печать исходный и преобразованный массивы.
140. Ввести массив А(N). Вывести на печать данные о числе нулевых элементов массива и их номера. Если таковых нет, вывести на печать сообщение об этом.
141. Ввести массив А(N). Элементы с индексами от N/2 до N увеличить в два раза. Остальные элементы оставить неизменными. Вывести на печать исходный и преобразованный массивы.
142. Ввести массив А(N). Разделить его на два массива. Первый из элементов А(N) с четными индексами, второй - с нечетными. Вывести на печать исходный и вновь образованный массивы.
143. Ввести массив А(N). Элементы массива, индекс которых кратен 4, увеличить вдвое. Вывести на печать исходный и преобразованный массивы.
144. Ввести массив А(N). Его четные элементы увеличить вдвое, а нечетные уменьшить в три раза. Вывести полученный массив на печать.
145. Ввести массив А(N). Ввести произвольное число L. Четные элементы массива большие L уменьшить на 5. Вывести на экран исходный и преобразованный массивы.
146. Ввести массив А(N), состоящий из положительных и отрицательных чисел. Образовать из него два массива - один из отрицательных, другой из положительных чисел. Вывести все три массива на печать.
147. Ввести массив А(N). Элементы с индексами кратными трем уменьшить в два раза, а элементы с индексами кратными 5 увеличить в три раза. Вывести полученный массив на печать.
148. Ввести массив А(N). Преобразовать его так, чтобы значения элементов являлись суммой элемента массива и его индекса. Вывести на печать исходный и преобразованный массивы.
149. Ввести массив А(N), состоящий из положительных и отрицательных чисел. Образовать из него массив, где все положительные числа заменить на1, отрицательные на – 1. Вывести на печать исходный и преобразованный массивы и информацию о количестве положительных и отрицательных чисел.
150. Ввести массив А(N), N>10. Первые пять элементов умножить на 2, вторые пять на 3, остальным присвоить значения равные нулю. Вывести на печать исходный и преобразованный массивы.
151. Ввести массив А(N) из положительных и отрицательных единиц и нулей. Вывести на печать информацию о количестве положительных и отрицательных единиц, нулей, а так же об их индексах.
152. Ввести массив А(N) и три числа *m*, *n*, *k*. Преобразовать исходный массив так, чтобы элементы A(I) ≤ *m* заменить на *m*, *m*≤ A(I) ≤ *n* – на *n*, *n*≤ A(I) ≤ *k –*  на *k*, А(I) > *k* – на 2*k*.
153. Ввести массив А(N). Найти сумму элементов, индексы которых кратны 3, а так же сумму остальных элементов.
154. Ввести массив А(N). Найти минимальный элемент среди четных и максимальный среди нечетных элементов массива.
155. Ввести массив А(N). Найти произведение элементов массива с четными номерами и сумму элементов массива с нечетными номерами.
156. Ввести целочисленный массив А(N). Найти минимальное значение среди элементов с четными номерами и максимальное значение среди элементов с нечетными номерами.
157. Ввести массив А(N). Найти минимальное и максимальное значение раздельно для отрицательных и положительных элементов массива.
158. Задать массив А(N), N = К + М. Элементы массива с индексом от 1 до К – члены натурального ряда кратные 2, остальные – кратные 5. Найти сумму и произведение элементов массива и частное от деления суммы на произведение.
159. Ввести массив А(N). Найти произведение Р элементов и сумму последних К элементов. Определить, что больше - произведение или сумма.
160. Ввести массив А(N). Найти минимальный элемент массива среди кратных трем и заменить его значением минимального элемента среди кратных двум. Вывести на печать исходный и преобразованный массивы.
161. Заполнить массив членами натурального ряда (1, 2, 3.....) до тех пор пока их произведение не станет больше наперед заданного числа М. Вывести на печать произведение и число элементов массива.
162. Заполнить массив числами натурального ряда (1, 2 .....) до тех пор пока их сумма не станет больше наперед заданного числа М. Вывести на печать сумму и число элементов массива.
163. Заполнить массив А(N) членами натурального ряда, кратными трем (3, 6, 9...) до тех пор пока сумма членов ряда не станет больше наперед заданного числа М. Найти произведение элементов массива и их число.
164. Заполнить массив А(N) четными членами натурального ряда (2, 4, 6.....) до тех пор пока их произведение не станет больше произвольного числа М. Найти сумму элементов ряда и их числа.
165. Заполнить массив А(N) нечетными членами натурального ряда (1, 3, 5.....) до тех пор пока их произведение не станет больше произвольного числа М. Найти сумму элементов массива и их число.
166. Заполнить массив А(N) нечетными членами натурального ряда (1,3,5...) до тех пор пока их сумма не станет больше произвольного числа М. Найти произведение членов ряда и их число.
167. Заполнить массив А(N) четными числами натурального ряда (2, 4, 6.....) до тех пор пока их сумма не станет больше произвольного числа М. Найти произведение членов ряда и их число.
168. Ввести массив А(N) из членов ряда А(I) = . Число членов ряда ограничены по условию А(I) ≤ М, где М >> 1. Найти сумму членов ряда кратных двум и произведение членов ряда кратных трем.
169. Задать массив А(N), N = К + М. Элементы массива с индексом от 1 до К – члены натурального ряда кратные 2, остальные – кратные 5. Найти сумму и произведение элементов массива и частное от деления суммы на произведение.
170. Задать массив А(N), элементы которого являются членами ряда А(I) = I2 не большими наперед заданного числа М ≥ 1000. Найти среднее значение элементов массива.
171. Задать массив А(N) из членов ряда А(I) = . Число элементов массива ограничить по условию А(I) ≤ К, где К >> 1. Найти сумму, произведение и среднее значение элементов массива.
172. Задать массив А(N) из ненулевых членов ряда А(I) = I3 не больших наперед заданного числа М ≥ 2000. Найти произведение элементов массива и их сумму.
173. Задать массив А(N), N = К + М. Элементы массива с индексом от 1 до К – четные члены натурального ряда (2, 4, 6...), остальные нечетные. Найти сумму и произведение элементов массива.
174. Заполнить массив А(N) членами натурального ряда, кратным трем (3, 6, 9...) до тех пор пока их произведение не станет больше произвольного числа М. Найти сумму элементов массива и их число.
175. Задать массив А(N) из членов ряда А(I) = 1/I, где I = 1, 2... Число элементов массива ограничивается по условию А(I) ≤ К, где К << 1. Найти сумму элементов массива, их произведение и среднее значение.
176. Задать массив А(N) из первых двадцати членов ряда А(I) = . Найти сумму первых десяти элементов массива и произведение вторых десяти элементов массива.
177. Ввести массив А(N) из членов ряда А(I) = . Число элементов ограничены по условию А(I) ≤ М, где М >> 1. Найти сумму четных членов ряда и произведение нечетных членов ряда.
178. Ввести массив А(N). Упорядочить его по возрастанию.
179. Ввести массив А(N). Упорядочить его по убыванию.

180. Заполнить массив А(N) числовыми данными целого типа. Образовать из него два массива В(N) и C(N), состоящих из положительных и отрицательных элементов и подсчитать их количество. Вывести все результаты на экран.

ПРИЛОЖЕНИЕ 1

*Образец оформления титульного листа*

|  |  |
| --- | --- |
| **КГЭУ** | МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ **Федеральное государственное бюджетное образовательное**  **учреждение высшего образования** «КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ» (ФГБОУ ВО «КГЭУ») |

Кафедра «Информатика и информационно-управляющие системы»

Контрольная работа

по дисциплине

«Информационные и компьютерные технологии»

Студент Шкапин А.В.

Группа ЗЭЭ-1-17

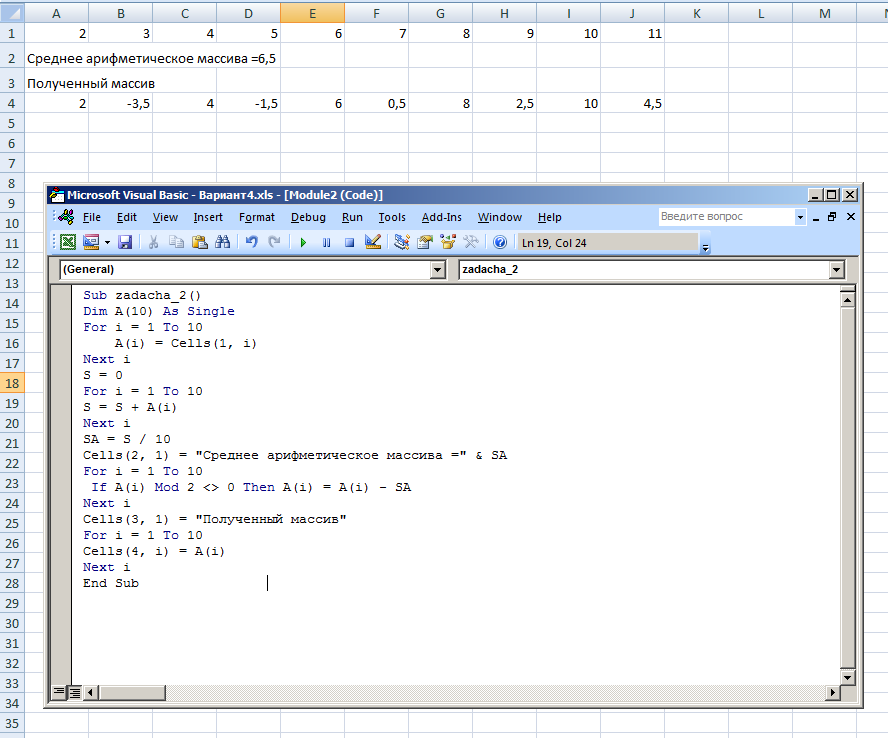
Шифр \_\_\_\_\_\_\_

# Казань 2017

ПРИЛОЖЕНИЕ 2

*Образец оформления задания*

34. Ввести массив А(N). Найти среднее арифметическое значение массива. Из нечетных элементов массива вычесть среднее арифметическое. Вывести преобразованный массив.



ПРИЛОЖЕНИЕ 3

*Образец оформления задания*

1. Ввести массив А(N). Найти среднее арифметическое значение массива. Первый элемент массива заменить на среднее арифметическое. Вывести преобразованный массив.

Решение:

Sub zadacha\_2()

Dim A(10) As Single

For i = 1 To 10

A(i) = Cells(1, i)

Next i

For i = 1 To 10

S = S + A(i)

Next i

SA = S / 10

Cells(2, 1) = " Ср.арифм.=" & SA

A(1) = SA

Cells(3, 1) = " Преобразованный массив:"

For i = 1 To 10

Cells(4, i) = A(i)

Next i

End Sub

Результат:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 5 | -6 | 7 | 8 | 9 | 10 | 11 |
| Ср.арифм.=5,3 | |  |  |  |  |  |  |  |  |
| Преобразованный массив: | | |  |  |  |  |  |  |  |
| 5,3 | 3 | 4 | 5 | -6 | 7 | 8 | 9 | 10 | 11 |