МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

ИНФОРМАТИКА

Методические рекомендации к самостоятельной работе для студентов специальности 1-36 01 06 «Оборудование и технология сварочного производства» заочной формы обучения

Могилев 2022

УДК 004
ББК 32.81
И74

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Автоматизированные системы управления» «07» декабря 2021 г., протокол № 5

Составители: канд. техн. наук, доц. В. М. Ковальчук;
канд. техн. наук, доц. В. А. Широченко

Рецензент канд. техн. наук, доц. В. В. Кутузов

Методические рекомендации предназначены для студентов специальности 1-36 01 06 «Оборудование и технология сварочного производства» заочной формы обучения.

Учебно-методическое издание

ИНФОРМАТИКА

Ответственный за выпуск А. И. Якимов

Корректор А. А. Подошевко

Компьютерная верстка Н. П. Полевничая

Подписано в печать . Формат 60 × 84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 21 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

Введение

Цель методических рекомендаций к самостоятельной работе по дисциплине «Информатика» заключается в овладении и закреплении студентами практических навыков работы в среде приложений MS Office.

Целью преподавания дисциплины является изучение основных современных операционных систем и программных сред, пакетов прикладных программ для научных и инженерных расчетов, изучение основ программирования, общих вопросов алгоритмизации и приобретение навыков решения задач с применением средств вычислительной техники.

Дисциплина «Информатика» является неотъемлемой частью современных инженерных знаний и входит в состав естественно-научных дисциплин, компонентов учреждения высшего образования.

Полученные при изучении дисциплины знания и навыки будут востребованы при изучении специальных дисциплин инженерной направленности и станут инструментом для грамотного выполнения и оформления рефератов, курсовых и дипломных работ.

Выполнение каждой работы производится в следующем порядке:

1) ознакомиться с теоретическими положениями работы;

2) из таблицы «Варианты заданий для выполнения работы» по последней цифре зачетной книжки выбрать исходные данные для выполнения задания;

3) ответить на контрольные вопросы, приведенные в конце каждой работы.

1 Самостоятельная работа № 1. Операционная система Windows и файловые менеджеры

Цель работы.

Ознакомиться с системным программным обеспечением персональной ЭВМ и файловыми менеджерами.

Методические указания.

Программное обеспечение (ПО) компьютера можно разделить на системное, сервисное и прикладное.

Системное ПО – это операционные системы (ОС), которые не решают конкретные прикладные задачи, а лишь обеспечивают их выполнение, предоставляя им сервисные функции, и управляют аппаратными ресурсами компьютера, такими как процессор, оперативная память, устройства ввода-вывода, сетевое оборудование.

В настоящее время известно большое количество ОС: MS DOS; OS/2; MS Windows 95, 98, 2000, XP, 7; Linux и др.

Любая ОС выполняет три базовые функции: ведение диалога с пользователем; управление файловой системой ПК; запуск и завершение приложений (программ).

Одной из наиболее распространенных ОС является MS Windows – многозадачная операционная система с развитым пользовательским интерфейсом, основой которого является рабочий стол. На рабочем столе располагаются значки и ярлыки. Значок – это графическое изображение, связанное с каким-либо объектом ОС, таким как файл, папка, диск. Ярлык представляет собой ссылку на файл, папку или диск.

Сетевое ПО предназначено для управления общими ресурсами в компьютерных сетях: сетевыми накопителями на магнитных дисках, принтерами, сканерами, передаваемыми сообщениями и т. д. К сетевому ПО относят ОС, поддерживающие работу компьютеров в сети, а также отдельные сетевые программы и пакеты программ.

Сервисное ПО используется для расширения возможностей ОС и представления набора дополнительных услуг, таких как программы контроля, тестирования и диагностики, программы-драйверы, программы-упаковщики и антивирусные программы. Сервисные программы можно разделить на две группы – программы-оболочки операционных систем и утилиты.

Программы-оболочки операционных систем помогают работать с файловой системой компьютера. Они существенно упрощают выполнение всех операций с файлами и каталогами, поэтому их также называют файловыми менеджерами. Известны такие программы-оболочки, как Total Commander, Windows Commander, FAR, Norton Commander и др.

Утилиты предоставляют пользователям средства обслуживания компьютера и его программного обеспечения. Они обеспечивают реализацию следующих функций: обслуживание файлов, каталогов и дисков, предостав-

ление информации о ресурсах компьютера, архивация (упаковка) файлов и др.

Для работы с файловой системой компьютера в ОС MS Windows используется файловый менеджер – программа Проводник, которая запускается командой меню: Пуск → Программы → Стандартные → Проводник. Окно Проводника разделено на две области, что значительно увеличивает объем информации о файловой системе компьютера. В левой области отображается структура пространства имен Windows – деревья папок жестких дисков, USB флеш-накопителей, компакт-дисков, а также системные папки, сетевые диски и другие, подключенные к компьютеру, ресурсы. Правая область Проводника называется окном содержимого. Она похожа на окно папки и функционирует аналогично ему.

Порядок выполнения работы.

1 Выбрать вариант задания из таблицы 1.1.

2 Выполнить операции с заданной программой и их описание.

Таблица 1.1 – Варианты заданий для выполнения работы

Номер варианта	Прикладная программа	Номер варианта	Прикладная программа
1	Для дефрагментации дисков	6	Для архивации файлов разработки ПО
2	Для рисования	7	Для работы с текстом в формате DOS
3	Для работы с текстом	8	Для работы с файлами и папками
4	Для работы с таблицами	9	Для работы с графикой
5	Для работы в интернете	10	Для разработки презентаций

Контрольные вопросы

1 Для чего в ОС MS Windows используется панель управления?
2 Где можно получить информацию об ОС, дисках и папках?
3 Перечислите основные функции ОС MS Windows.
4 Как классифицируется ПО компьютера?
5 Назовите основные элементы интерфейса Windows.
6 Дайте определения папки, файла, ярлыка.
7 Для каких целей используется панель управления Windows?
8 Для чего в ОС MS Windows предназначен буфер обмена?

2 Самостоятельная работа № 2. Текстовый процессор Microsoft Word

Цель работы.

Приобрести навыки работы с текстовым редактором Word.

Методические указания.

Редактор MS Word является одним из наиболее популярных и мощных редакторов. Word обладает удобством набора, редактирования и оформления текста, содержит большое количество значков-кнопок для ускоренного выполнения команд, имеет развитые системы стилей оформления и для работы с таблицами, возможность украшать текст разнообразными линиями, обеспечивает обтекание текстом вставленных в документ объектов и широкие возможности создания бланков, шаблонов, веб-страниц, построения сносок и примечаний, оглавлений, формул; проверки правописания, подбора синонимов, автоматического переноса слов и многое др.

Порядок выполнения работы.

1 Выбрать вариант задания по таблице 2.1.

2 Создать документ, имеющий параметры согласно варианту задания.

Таблица 2.1 – Варианты заданий к лабораторной работе

Номер варианта	Поле, см				Шрифт		Межстроч-ный интервал	Отступ в первой строке абзаца, см
	верхнее	нижнее	левое	правое	Тип*	Размер, пт		
1	3	2	2	4	1	10	1	5
2	3	1	2	4	1	11	1	5
3	3	3	2	4	1	12	1	5
4	2	3	1	4	1	13	1,5	4
5	2	1	1	3	1	14	1,5	4
6	2	2	1	3	1	10	1,5	4
7	2	4	3	3	1	11	1,5	4
8	4	1	3	3	2	12	2	3
9	4	2	3	2	2	13	2	3
10	4	3	4	2	2	14	2	3
Примечание – * – первая цифра – тип шрифта: 1 – Times New Roman; 2 – Arial								

Контрольные вопросы

1 Назначение текстового редактора Word. Что такое окно прикладной программы и окно документа?

2 Из чего состоит строка заголовка? Для чего предназначены меню Файл, Правка, Вид, Вставка, Формат, Таблица, Окно?

3 Опишите элементы стандартной панели инструментов.
4 Опишите элементы инструментов форматирования.
5 Как выделить необходимый текстовый фрагмент?

3 Самостоятельная работа № 3. Текстовый процессор Microsoft Word

Цель работы.

Приобретение практических навыков по работе с таблицами, при наборе формул, создании диаграмм и рисунков в редакторе Microsoft Word.

Методические указания.

Таблицы предназначены для упорядочивания данных и создания интересных макетов страницы с последовательно расположенными столбцами текста или графики. Для работы с таблицами используется пункт меню *Таблица*. Наиболее быстрый путь создания простой таблицы, например такой, которая имеет одинаковое количество строк и столбцов, с помощью команды *Добавить таблицу*.

Создание простой таблицы:

– выберите место создания таблицы;

– выберите команду *Добавить таблицу* из меню *Таблица*;

– установите нужные параметры (число столбцов и строк, ширина столбцов, автоформат таблицы).

Редактор математических формул *MicrosoftEquationEditor* вызывается по OLE-технологии и ориентирован на создание сложных формул, как правило, содержащих матрицы, дроби, знаки суммирования и другие математические конструкции.

Из текстового редактора Word приложение вызывается выбором команды основного меню *Вставка–Объект*. Окно приложения содержит пиктографическое и основное меню. Пиктографическое меню включает наборы шаблонов и математических конструкций и символов, упорядоченных по функциональному назначению. Для включения в формулу нужной конструкции или символа пользователю необходимо зафиксировать курсор мыши на соответствующей кнопке пиктографического меню и, перемещая курсор, выбрать нужное. Для закрытия приложения *Microsoft Equation Editor* и возврата в исходный документ достаточно зафиксировать курсор мыши вне поля приложения.

Текстовый редактор Word позволяет создавать и редактировать рисунки. Рисунок можно импортировать из набора стандартных файлов с помощью команды *Вставка-рисунок* или создавать с помощью инструментов для рисования.

Вызов пиктографического меню для рисования осуществляется с помощью команды *Вид / Панель инструментов* / *Рисование*.

Порядок выполнения работы.

1 Создать новый документ, который должен содержать таблицу.

2 Создать новый документ, который должен содержать формулы и рисунок.

Контрольные вопросы

1 Какой пункт меню Word используется для работы с таблицами?

2 Как удалить строку или столбец из таблицы?

3 Как объединить несколько ячеек?

4 Как добавить в текстовый документ математическую формулу?

5 Как сгруппировать несколько нарисованных элементов?

4 Самостоятельная работа № 4. Электронные таблицы и табличный процессор Excel

Цель работы.

Изучение технологии работы с функциями в среде MS Excel.

Методические указания.

При построении расчетной таблицы в MS Excel имеется возможность использования большого количества встроенных функций, которые сосредоточены в удобном инструменте их ввода – мастере функций, объединяющем девять следующих категорий: финансовые, дата и время, математические, статистические, ссылки и массивы, работа с базой данных, текстовые, логические, проверки свойств и значений.

Любая функция имеет вид: *имя* (*список аргументов*), где *имя* – это символическое имя функции, а *список аргументов* – величины, над которыми функция выполняет операции.

Аргументами функции могут быть адреса ячеек, константы, формулы, а также другие функции. Например, функция *КОРЕНЬ*(*ABS*(*A*2)) вычисляет квадратный корень из абсолютного значения числа из ячейки *A*2.

Для использования мастера функций необходимо кликнуть на кнопке *fx* в строке ввода окна Excel, в открывшемся окне *Мастер функций* выбрать категорию, содержащую вызываемую функцию, выбрать требуемую функцию и нажать кнопку *OK*. Откроется окно мастера вызываемой функции, в котором необходимо ввести требуемые параметры, и нажать *OK*.

При решении многих задач в Excel требуется использование матриц, например, решение систем линейных алгебраических уравнений, задач линейного программирования и др. Для работы с матрицами эффективным является использование формул специального вида, называемыми формулами массива, которые отличаются от обычных формул тем, что их аргументами и результатом является не одно число, а набор величин – матрица. При завершении таких операций требуется специальное подтверждение – вместо клавиши ENTER используется комбинация из трех клавиш – *Ctrl* + *Shift* + *Enter*.

Пример – Пусть требуется выполнить автошкалирование (стандартизацию) данных, записанных в матрице *X*. Для этого сначала нужно вычислить средние значения m_j и среднеквадратичные отклонения s_j для каждого (j-го) столбца матрицы *X*, а затем вычесть из каждого элемента столбца величину m_j и поделить на величину s_j:

$$\tilde{x}_{ij} = \left(x_{ij} - m_j \right) / s_j .$$

Такое преобразование можно сделать с помощью обычных формул. Но если матрица *X* велика, то удобнее воспользоваться формулой массива. Назовем соответствующие области на листе: *X*, *m* и *s* (*Формулы → Присвоить имя*). Выделим пустую область *O3:S7*, щелкнем в строке формул, введем = (*X* – *m*)/*s* и завершим ввод комбинацией клавиш *Ctrl* + *Shift* + *Enter*. Если все сделано правильно, то в выделенной области появится формула массива {= (*x* – *m*)/*s* }, заключенная в фигурные скобки (рисунок 4.1).

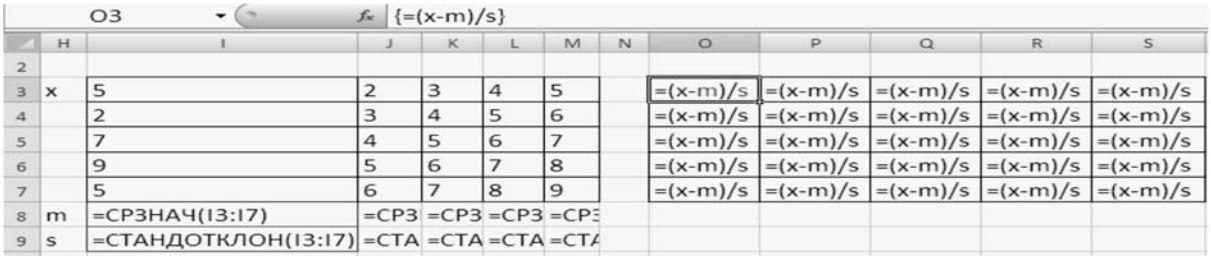


Рисунок 4.1 – Формула массива

Порядок выполнения работы.

1 Выбрать задание к работе (таблица 4.1).

2 Разработать электронную таблицу «Формулы массива».

3 Разработать электронную таблицу «Встроенные функции».

Таблица 4.1 – Варианты заданий для выполнения работы

Вариант	Вычислить, используя формулы массива	Вычислить функции
1	Матрицу B, обратную матрице |C| размерности 6 × 6	$y = x^2 + \cos(x)$
2	|C| = |A| · |B|, размерность |A| – (4 × 4), |B| – выбрать самостоятельно	$y = \mathrm{tg}(x) - x$
3	Таблицу умножения от 5 до 14	$y = x - \cos(x)$
4	|C| = |A| · |B|, размерность |A| – (4 × 2), |B| – выбрать самостоятельно	$y = \mathrm{ctg}(x) - x^2$
5	Таблицу умножения от 1 до 12	$y = \ln(x) - \sin(x)$
6	Матрицу B, обратную матрице |C| размерности 4 × 4	$y = x - \sin(x)$
7	Таблицу умножения от 1 до 15	$y = \sqrt{x} + \sin(x)$
8	|C| = |A| · |B|, размерность |A| – (3 × 5), |B| – выбрать самостоятельно	$y = \sqrt{x^2 + 3} + \cos(x)$

Контрольные вопросы

1 Какие категории встроенных функций программы MS Excel Вы знаете?
2 Как ознакомиться с технологией использования встроенной функции?
3 Какие функции содержатся в категории *Логические*?
4 Что такое формулы массива и для чего они используются?
5 Как завершается ввод формулы массива?

5 Самостоятельная работа № 5. Электронные таблицы и табличный процессор Excel

Цель работы.

Изучение технологии сортировки, фильтрации и консолидации данных в MS Excel.

Методические указания.

Таблицы MS Excel могут содержать большое количество данных, представленных в виде списка, который состоит из записей (строк), а столбцы содержат однотипные данные (поля). Список не должен содержать пустых строк или столбцов. Стиль оформления заголовка списка должен быть отличным от стиля оформления его информационных строк – записей.

Со списками можно выполнять такие операции, как сортировка и фильтрация, которые значительно облегчают поиск в них информации.

Сортировка позволяет расположить записи списка в порядке, определенном значениями выбранных столбцов: по алфавиту, по возрастанию/убыванию и т. д.

Смысл фильтрации данных состоит в том, что после ее применения в списке остаются только те записи (строки), которые удовлетворяют заданным условиям отбора. В MS Excel используются фильтры двух типов: автофильтр и расширенный фильтр.

Автофильтр вызывается: вкладка *Данные* → группа *Сортировка и фильтр* → кнопка *Фильтр* или вкладка *Главная* → группа *Редактирование* → кнопка *Сортировка и фильтр* → команда *Фильтр*. При выполнении операций со списком курсор должен находиться на территории списка. Поля, на которых установлен фильтр, отображаются со значком воронки, подведение указателя мыши к которой, приводит к отображению условий фильтрации.

Расширенный фильтр используется в тех случаях, когда результат отбора необходимо поместить отдельно от основного списка, и требуется наличие диапазона условий (критериев), которые размещаются в блоке ячеек, указанном в поле *Диапазон условий*, окна *Расширенный фильтр*. Вызов расширенного фильтра выполняется в следующем порядке: вкладка *Данные* → группа *Сортировка и фильтр* → кнопка *Дополнительно*.

Диапазон условий отбора необходимо формировать по следующим правилам: первая строка должна содержать строку заголовков списка, а вторая

и последующие – условия отбора, которые строятся на основе значений полей с использованием подстановочных знаков (? – один любой символ, * – любое количество любых символов, ~ – один из знаков: ?, * или ~), а также операций сравнения (>, >=, <, <=, =, <>). Предполагается, что условия, построенные в одной строке, соединены логической операцией «И», а в разных строках – операцией «ИЛИ». Между диапазоном условий и списком (базой данных) должна быть хотя бы одна пустая строка.

Консолидация данных – это способ получения итоговой информации из разных листов одинаковой структуры. Итоговая информация формируется в сводной таблице на отдельном листе. Структура этого листа аналогична структуре консолидируемых листов, которые должны быть упорядочены по какому-либо полю.

Для консолидации необходимо, находясь на листе сводной таблицы, выполнить на вкладке *Данные* в группе *Работа с данными* команду *Консолидация* и в открывшемся окне *Консолидация* выбрать функцию, например, *Сумма*, щелкнуть мышью в поле *Ссылка*, перейти на первый консолидируемый лист и выделить итоговую сумму поля. Данные появятся в поле *Ссылка*. Затем нажать кнопку *Добавить* и выполнить аналогичные действия для остальных консолидируемых листов.

После этого можно отметить флажок *Создавать связи с исходными данными*, чтобы при изменении исходных таблиц автоматически пересчитывалась и сводная таблица.

По данным сводной таблицы можно рассчитать промежуточные итоги.

Порядок выполнения работы.

1 Выбрать задание к работе (таблица 5.1).

Таблица 5.1 – Варианты заданий для выполнения работы

Вариант	Сортировка	Фильтр	Расширенный фильтр	Консолидация
1	Времена года	Реки	Имена президентов	Озера
2	Деревья	Посуда	Мебель	Улицы города
3	Озера	Марки авто	Факультеты вуза	Озера
4	Улицы города	Факультеты вуза	Имена художников	Посуда
5	Реки	Реки	Посуда	Мебель
6	Имена президентов	Посуда	Моющие средства	Реки
7	Фамилии писателей	Мебель	Учебные заведения	Посуда
8	Учебные заведения	Реки	Фамилии писателей	Марки авто
9	Моющие средства	Марки авто	Имена президентов	Реки
10	Посуда	Мебель	Учебные заведения	Улицы города

2 На листе 1 первого файла построить таблицу (базу данных), содержащую поля: №, Наименование, Цена, Количество, Сумма, и ввести в нее семь строк с данными.

3 Выполнить операцию «Сортировка».

4 Скопировать базу данных на лист 2 первого файла и построить «Фильтр».

5 Скопировать базу данных на лист 3 и построить «Расширенный фильтр».

6 Скопировать таблицу из листа 1 файла 1 на четыре листа второго файла и выполнить операцию *Консолидация данных*.

Контрольные вопросы

1 Как выполнить сортировку списка (базы данных)?
2 Как построить фильтр списка MS Excel?
3 Что такое расширенный фильтр MS Excel и как его построить?
4 Как построить критерии отбора расширенного фильтра?
5 Что такое консолидация данных в MS Excel?

6 Самостоятельная работа № 6. Электронные таблицы и табличный процессор Excel

Цель работы.

Изучение технологии решения нелинейных и линейных уравнений с помощью инструментов MS Excel.

Методические указания.

Инструмент Excel *Подбор параметра* можно использовать в тех случаях, когда необходимо найти значение одной из переменных, которое используется для получения уже известного результата, заданного формулой.

Пусть, например, требуется найти корни квадратного уравнения

$$2x^2 - 3x - 2 = 0\,.$$

Для решения будем полагать, что неизвестная x – в ячейке $B1$, а в ячейку $B2$ введено уравнение (рисунок 6.1). Вызовем инструмент *Подбор параметра*: *Данные → Работа с данными → Анализ «Что если»*. Введем необходимые параметры в открывшееся окно *Подбор параметра* и нажмем кнопку *ОК*.

В результате найдем решение; $x_1 = -0{,}499952$. Чтобы найти второй корень уравнения введем в ячейку $B1$ начальное приближение 10 и повторим указанные выше действия, в результате получим $x_2 = 2{,}000$.

С помощью формул массива удобно решать такие задачи, как вычисление определителей, обратной матрицы, решение систем линейных алгебраических

уравнений (СЛАУ) и др. Например, чтобы решить СЛАУ $|A||x| = |B|$, можно воспользоваться известной формулой $x = |A^{-1}||B|$.

Тоесть достаточно найти обратную матрицу A^{-1} и умножить ее на столбец свободных членов B (рисунки 6.2 и 6.3).

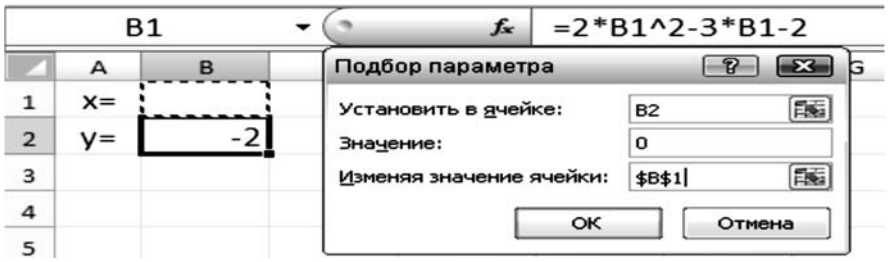


Рисунок 6.1 – Окно инструмента *Подбор параметра*

	A	B	C	D	E	F	G	H	I	J	K	L	M
1			A			B			A-1				x
2	2	5	7	8		4		=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)		=МУМНОЖ(H2:K5;F2:F5)
3	1	6	3	2		5		=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)		=МУМНОЖ(H2:K5;F2:F5)
4	7	4	9	3		7		=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)		=МУМНОЖ(H2:K5;F2:F5)
5	2	4	7	5		9		=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)	=МОБР(A2:D5)		=МУМНОЖ(H2:K5;F2:F5)

Рисунок 6.2 – Таблица с формулами решения СЛАУ

	A	B	C	D	E	F	G	H	I	J	K	L	M
1			A			B			A-1				x
2	2	5	7	8		4		0,258	0,000	0,226	-0,548		-2,323
3	1	6	3	2		5		0,002	0,231	-0,007	-0,092		0,285
4	7	4	9	3		7		-0,313	-0,077	-0,062	0,568		3,045
5	2	4	7	5		9		0,333	-0,077	0,002	-0,303		-1,762

Рисунок 6.3 – Таблица с результатами решения СЛАУ

Порядок выполнения работы.

1 Выбрать задание на выполнение работы (таблицы 6.1 и 6.2).

2 Разработать электронную таблицу «Подбор параметра».

3 Разработать таблицу «Решение СЛАУ».

Таблица 6.1 – Варианты заданий для выполнения работы

Номер варианта	Вид функции	Номер варианта	Вид функции
1	$2x^2 - x - 4 = 0$	6	$x^2 - 0,5x - 2,6 = 0$
2	$2x^2 - 3x - 2,4 = 0$	7	$x^2 - 0,8x - 2,2 = 0$
3	$2x^2 - 0,7x - 2,4 = 0$	8	$2,7x^2 - 0,2x - 2,8 = 0$
4	$2,2x^2 - 1,7x - 1,4 = 0$	9	$2x^2 - 1,5x - 1,4 = 0$
5	$3x^2 - 0,5x - 2 = 0$	10	$1,2x^2 - x - 2 = 0$

Таблица 6.2 – Варианты заданий для решения СЛАУ

Номер варианта	СЛАУ	Номер варианта	СЛАУ
1	$-2x_1 + x_2 - 4x_3 = 4$ $3x_1 + 2x_2 - 4x_3 = 1$ $-2x_1 - 5x_2 - 4x_3 = 3$	4	$-x_1 + x_2 - 3x_3 = 2$ $2x_1 + 2x_2 + 4x_3 = 2$ $-2x_1 - 3x_2 + 2x_3 = 3$
2	$2x_1 - x_2 + 4x_3 = 2$ $4x_1 + 2x_2 - 4x_3 = 1$ $-x_1 - 5x_2 + 4x_3 = 2$	5	$x_1 - 2x_2 + 4x_3 = 3$ $-3x_1 + 2x_2 - 5x_3 = 4$ $-2x_1 - 2x_2 - 5x_3 = 4$
3	$-3x_1 - 2x_2 - 5x_3 = 2$ $2x_1 + 4x_2 - 3x_3 = 6$ $-2x_1 - 5x_2 - 4x_3 = 4$	6	$2x_1 - x_2 - 4x_3 = 1$ $3x_1 + 2x_2 - 4x_3 = 2$ $-2x_1 - 5x_2 - 4x_3 = 2$

Контрольные вопросы

1 Для чего предназначен инструмент Ms Excel *Подбор параметра*?
2 Как, используя *Подбор параметра,* решить квадратное уравнение?
3 Какие уравнения относятся к нелинейным?

7 Самостоятельная работа № 7. Электронные таблицы и табличный процессор Excel

Цель работы.

Изучение технологии решения нелинейных и линейных уравнений с помощью инструментов MS Excel.

Методические указания.

Нелинейные уравнения можно решить также с помощью надстройки *Поиск решения* (вкладка *Данные /* группа *Анализ*) (рисунок 7.1). Здесь в ячейке *B*1 было введено начальное приближение «0». Нажав кнопку *Выполнить* в ячейке *A2* получим результат: $x_1 = 2{,}000$.

Чтобы найти второй корень, в ячейку *A*2 введем начальное приближение, например, «0». В результате получим $x_2 = -0{,}5000$.

Надстройку *Поиск решения* можно использовать также для решения систем линейных алгебраических уравнений (СЛАУ). Для этого подготовим таблицу, в ячейках *A2:D*5 которой запишем коэффициенты при неизвестных *x*, а ячейках *F2:F*5 – свободные члены (рисунок 7.2). В ячейках *F2:F*5 запишем формулы для вычисления так называемых невязок. После этого вызовем надстройку *Поиск решения*. Откроется окно *Поиск решения*. В поле *Установить целевую ячейку* введем первую невязку – ячейку *I*2, в поле *Равной значению* введем «0», а в окно *Ограничения* добавим остальные три невязки: *I*2 = 0; *I*4 = 0; *I*5 = 0. Блок результатов решения СЛАУ *H2:H*5 зададим в поле

Изменяя ячейки. Нажмем кнопку *Выполнить*. Тогда в ячейках *H2:H5* (рисунок 7.2) получим решение.

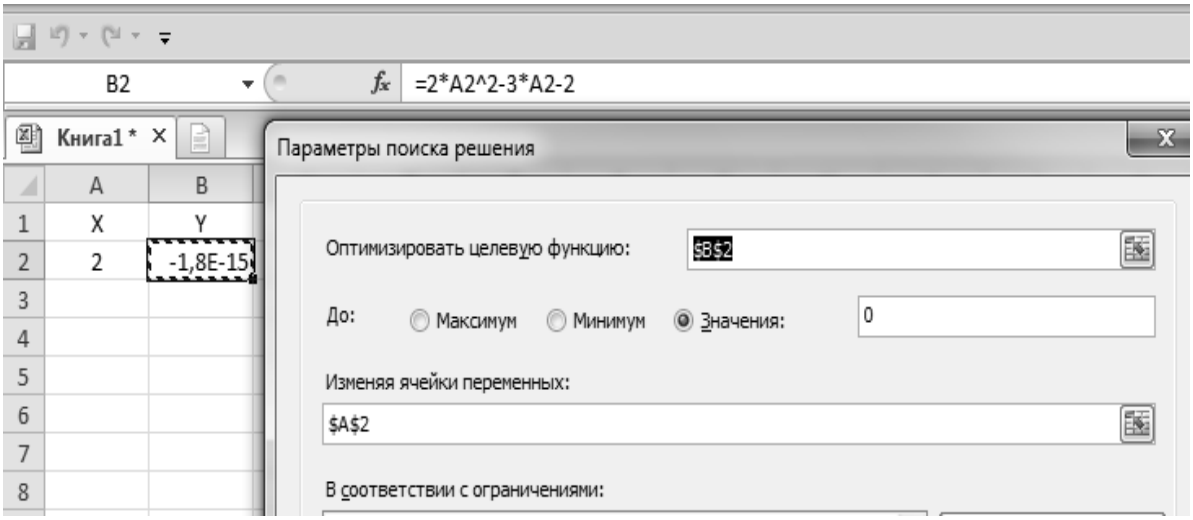


Рисунок 7.1 – Окно надстройки *Поиск решения*

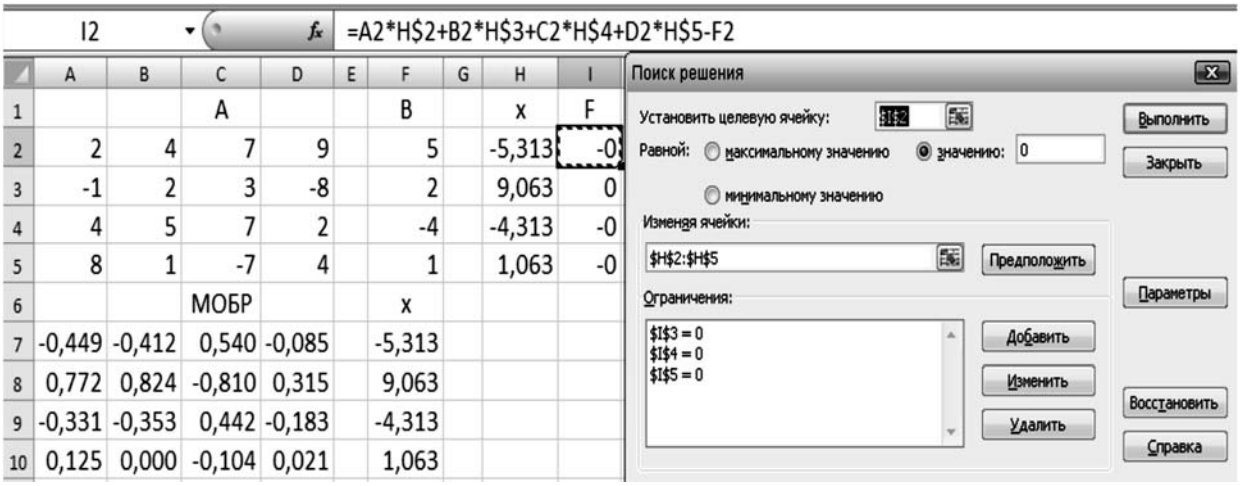


Рисунок 7.2 – Окно решения СЛАУ

Порядок выполнения работы.

1 Получить задание к работе (таблицы 7.1 и 7.2).

2 Разработать электронную таблицу «Поиск решения».

3 Разработать таблицу «Решение СЛАУ».

Таблица 7.1 – Варианты заданий для выполнения работы

Номер варианта	Вид функции	Номер варианта	Вид функции
1	$2x^2 - x - 4 = 0$	6	$x^2 - 0,5x - 2,6 = 0$
2	$2x^2 - 3x - 2,4 = 0$	7	$x^2 - 0,8x - 2,2 = 0$
3	$2x^2 - 0,7x - 2,4 = 0$	8	$2,7x^2 - 0,2x - 2,8 = 0$
4	$2,2x^2 - 1,7x - 1,4 = 0$	9	$2x^2 - 1,5x - 1,4 = 0$
5	$3x^2 - 0,5x - 2 = 0$	10	$1,2x^2 - x - 2 = 0$

Таблица 7.2 – Варианты заданий для решения СЛАУ

Номер варианта	СЛАУ	Номер варианта	СЛАУ
1	$-2x_1 + x_2 - 4x_3 = 4$ $3x_1 + 2x_2 - 4x_3 = 1$ $-2x_1 - 5x_2 - 4x_3 = 3$	4	$-x_1 + x_2 - 3x_3 = 2$ $2x_1 + 2x_2 + 4x_3 = 2$ $-2x_1 - 3x_2 + 2x_3 = 3$
2	$2x_1 - x_2 + 4x_3 = 2$ $4x_1 + 2x_2 - 4x_3 = 1$ $-x_1 - 5x_2 + 4x_3 = 2$	5	$x_1 - 2x_2 + 4x_3 = 3$ $-3x_1 + 2x_2 - 5x_3 = 4$ $-2x_1 - 2x_2 - 5x_3 = 4$
3	$-3x_1 - 2x_2 - 5x_3 = 2$ $2x_1 + 4x_2 - 3x_3 = 6$ $-2x_1 - 5x_2 - 4x_3 = 4$	6	$2x_1 - x_2 - 4x_3 = 1$ $3x_1 + 2x_2 - 4x_3 = 2$ $-2x_1 - 5x_2 - 4x_3 = 2$

Контрольные вопросы

1 Какие надстройки MS Excel Вы знаете?

2 Какие методы решения СЛАУ Вы знаете?

3 Для решения каких задач предназначена надстройка *Поиск решения*?

4 Какова структура электронной таблицы решения СЛАУ с помощью надстройки *Поиск решения*?

8 Самостоятельная работа № 8. Создание презентаций в MS PowerPoint

Цель работы.

Изучение технологии создание презентаций в MS PowerPoint.

Методические указания.

Пакет динамических презентаций PowerPoint (в дальнейшем PowerPoint) – один из лучших пакетов для подготовки и проведения презентаций.

Программа PowerPoint позволяет открыть уже готовую презентацию или создать новую с помощью команд Файл/Открыть, Файл/Создать или с использованием команд Открыть, Создать на панели Область задач.

Для обеспечения удобства работы пользователя в программе PowerPoint существуют следующие режимы работы с презентациями:

– режим слайда (просмотр или изменение текущего слайда);

– режим структуры (показ общей структуры – плана, сценария – презентации, что позволяет быстро изменить текстовую ее часть);

– режим страницы заметок (для заполнения или просмотра страниц заметок для каждого слайда);

– режим сортировщика слайдов (на экране отображаются миниатюры всех слайдов презентации).

Переход из одного режима в другой осуществляется с помощью команды Вид. Оформление слайда удобнее всего осуществлять в режиме слайдов.

Порядок оформления слайда.

1 Выбрать макет слайда на панели Разметка слайда (команда Формат/Разметка слайда).

2 Выбрать шаблон оформления, цветовую схему и эффект анимации при переходе к следующему слайду на панели Дизайн слайда (команда Формат/Оформление слайда).

3 В метки-заполнители ввести необходимую информацию.

4 Настроить анимацию объектов на слайде.

С помощью команды *Показ слайдов/Настройка анимации* можно установить все анимационные эффекты для слайда, например, обеспечить появление текста по буквам, словам или абзацам. Графические изображения и другие объекты (диаграммы, кино) могут появляться постепенно; также возможна анимация элементов диаграммы.

Для применения гиперссылок необходимо выделить текст или объект, представляющий гиперссылку, выполнить команду *Показ слайдов/Настройка действия*, затем выбрать способ перехода: страницу *По щелчку мыши* или страницу *По наведению указателя мыш*и (чтобы назначить объекту более одного действия, например, переход по гиперссылке и звук, следует установить гиперссылку на странице *По щелчку мыши*, а звук – на странице *По наведению указателя мыши*) → установить флажок Перейти по гиперссылке →выбрать место назначения гиперссылки → установить другие необходимые параметры.

Для создания кнопки действия необходимо выбрать на панели инструментов *Рисование* команду *Автофигуры/Управляющие кнопки* → выбрать нужную кнопку → щелчком мыши в поле слайда установить ее местоположение, в результате чего в слайд будет вставлена кнопка стандартного размера, принимаемого по умолчанию, в диалоговом окне Настройки действия определить назначение кнопки (для каждой кнопки предусмотрено действие по умолчанию: при нажатии кнопки [Отмена] все связанные с данной кнопкой действия отменяются). Определив порядок использования кнопки, можно ее перенести, а также изменить местоположение или размеры.

Порядок выполнения работы.

1 Подготовить презентацию для выбранной организации (список вариантов для выбора – таблица 8.1).

2 В презентации к каждому слайду оформить заметки, настроить переходы слайдов, анимацию объектов на слайдах, создать итоговый слайд из заголовков других слайдов, вставить гиперссылки из итогового слайда на все остальные слайды и кнопки возврата со слайдов на итоговый слайд. Настроить демонстрацию готовой презентации.

Таблица 8.1 – Варианты заданий для выполнения работы

Вариант	Организация
1	Отдел маркетинга крупного предприятия
2	Отдел рекламы крупного предприятия
3	Конструкторский отдел крупного предприятия
4	Издательство
5	Туристическая фирма
6	Дизайнерская фирма по созданию интерьера
7	Рекламное агентство
8	Торговая фирма
9	Производственно-коммерческое предприятие
10	Торгово-закупочная фирма

Контрольные вопросы

1 Назовите функциональные возможности программы PowerPoint.

2 Какие способы создания презентации существуют в программе PowerPoint? Какие возможности они предоставляют?

3 Как осуществляется настройка переходов слайдов?

4 Каким образом настроить анимацию объектов слайда?

5 Как создать итоговый слайд из заголовков других слайдов?

6 Каким образом создать гиперссылку?

7 Как создаются активные кнопки на слайдах?

9 Самостоятельная работа № 9. Ознакомление с интегрированной системой программирования. Разработка линейной программы со вводом и выводом данных

Цель работы.

Ознакомиться с основными окнами среды VBA и с процессом конструирования визуального проявления программы.

Методические указания.

Корпорация Microsoft интегрировала в свои офисные продукты язык программирования Visual Basicfor Applications (VBA). С помощью этого языка каждый пользователь может автоматизировать работу приложения и максимально приспособить его работу для решения текущих задач.

Код VBA набирается в редакторе Visual Basic. Для того чтобы попасть в этот редактор, выберите в MS Excel команду Сервис | Макрос | Редактор Visual Basic или нажмите комбинацию клавиш *Alt* + *F*11. В результате – интегрированная среда разработки приложений (IDE).

Возвратиться из редактора Visual Basic в рабочую книгу можно, нажав кнопку с пиктограммой Excel.

Все приложения, написанные на VBA, создаются как проекты. В проект входят несколько модулей: код программы, параметры формы, конфигурация интегрированной среды разработки и др.

Среда разработки приложений имеет стандартный для Windows-приложений вид: строка меню, панель инструментов и еще несколько открытых окон. На рисунке 9.1 открыты два окна: **Project – VBA Project** и **Properties**.

Окно **Project − VBA Project** активизируется выбором команды **View | Project Explorer** (1) или нажатием кнопки **Project Explorer** (2) панели инструментов. В окне **Project – VBAProject** представлена иерархическая структура форм и модулей текущего проекта. В этом окне отображается реестр модулей и форм, входящих в создаваемый проект. Двойным щелчком на значке модуля в окне **Project − VBA Project** можно открыть соответствующий модуль.

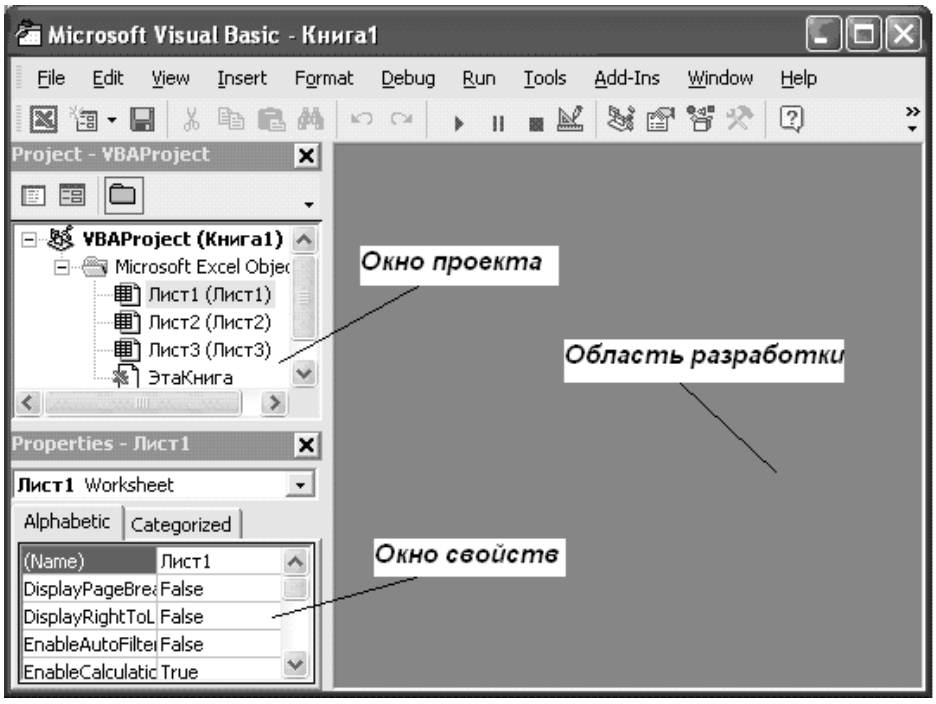


Рисунок 9.1 − Интегрированная среда разработки приложений (IDE)

В проекте автоматически создается по модулю для каждого рабочего листа и для всей книги. Кроме того, можно создавать и другие модули.

При написании кода программы необходимо придерживаться следующей структуры:

SubИмяПрограммногоМодуля ()
Объявление переменных и констант
Тело программы (последовательность исполняемых операторов)
EndSub

Переменные в программе используются для хранения данных, которые могут изменяться в процессе выполнения процедуры. Объявление типа переменной означает, что пользователь устанавливает определённые границы, в которых может изменяться переменная. Тип переменной можно вообще не определять. Если тип переменной не объявляется, по умолчанию он принимается как тип Variant. В таблице 9.1 содержится информация о размере данных, т. е. об объёме памяти, выделяемом для хранения данных.

Таблица 9.1 – Основные типы переменных языка VBA

Тип	Содержание	Объем	Диапазон значений
Byte	Короткое неотрицательное число	1 байт	0...255
Integer	Целое число	2 байта	–32768...+32767
Long	Длинное целое число	4 байта	–2147483648...+2147483647
Single	Десятичное число обычной точности	4 байта	±(1.4011298e–45…3.402823e+38)
Double	Десятичное число двойной точности	8 байт	±(1e–324...1e+308)
String	Набор символов (строка)	Зависит от числа символов в строке	
Boolean	Логическая величина	2 байта	Истина (True) или Ложь (False)
Date	Дата	8 байт	Информация о дате
Object	Указатель на объект	4 байта	Значением является ссылка на объект
Variant	Произвольное значение	Не менее 16 байт	Может быть переменной любого типа

Для объявления переменной используется оператор Dim. Этот оператор имеет следующий синтаксис:

DimИмяПеременнойAsTипДанных

Имена переменных должны начинаться с буквы, могут содержать так же цифры и знаки подчёркивания. Имя не может содержать пробелы, точки, запятые, восклицательные знаки и символы @, &, $, # и не должно иметь более 255 символов.

Если в вычислениях нужна величина, которая бы не меняла своего значения, то применяются константы. Для их объявления используется оператор Const, имеющий следующий синтаксис:

ConstИмяКонстантыAsTипДанных = Значение

Например:

ConstGruppaAsInteger = 25

Для задания переменным исходных значений используется функция InputBox, а для вывода значений переменных на экран − процедура MsgBox. Синтаксис их написания можно посмотреть в справочной системе при работе с кодом программы с помощью клавиши **F1**.

Для вычисления значений переменных в программе используется оператор присваивания, имеющий следующий синтаксис:

Переменная = Выражение

Если *Выражение* математическое, то в нем используются знаки арифметических действий, математические функции и круглые скобки, например:

A = 2 * sin(x + 2) / cos(x − 2)

Задание

1 Составить необходимый код программы с обеспечением ввода исходных данных и вывода результата на экран.

2 Запустить программу на выполнение и сравнить полученный результат со значением, вычисленным другим способом, например, с помощью калькулятора.

Контрольные вопросы

1 Как объявить переменную, которая может иметь целые значения в диапазоне от 0 до 100?

2 Как в программе присвоить значение переменной?

3 Как показать результат вычислений?

10 Самостоятельная работа № 10. Разработка разветленной программы с использованием оператора IF

Цель работы.

Изучить оператор ветвления ***If***; создать программу с использованием оператора ветвления ***If*** на языке программирования VBA.

Методические указания.

Алгоритм называется разветвляющимся, если последовательность выполнения его шагов изменяется в зависимости от выполнения некоторых условий. Условие – это логическое выражение, которое может принимать одно из двух значений: «*ДА*» – если условие верно (истинно, *TRUE*); «*НЕТ*» – если условие неверно (ложно, *FALSE*).

Схема алгоритма конструкции условного оператора ***If*** представлена на рисунке 10.1. Синтаксис условного оператора ***If*** в однострочной форме следующий:

If<лог. выраж.>Then $P_1 : P_2 : ... : P_N$*Else*$M_1 : M_2 : ... : M_N$

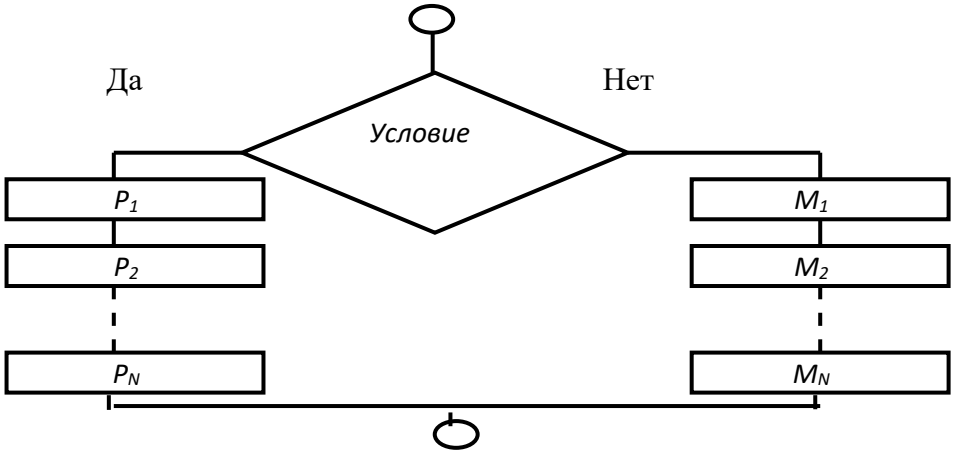


Рисунок 10.1 – Схема алгоритма конструкции условного оператора *If*

Возможна и другая синтаксическая форма – блочная (структурная):

If<логическое выражение> Then
    P_1
...
    P_N
Else
    M_1
...
    M_N
End If

где *If*, *Then*, *Else*, *End If* – зарезервированные слова;
P_1, P_2, P_N, M_1, M_2, M_N – операторы.

Задание

1 Запросить у пользователя ввод числа.
2 Сравнить введенное число с другим заданным числом, например 20.
3 По результатам сравнения вывести соответствующее сообщение:

«25 > 20» или «15 < 20»,

где 25, 15 – введенные пользователем числа.

Контрольные вопросы

1 Можно ли вставлять инструкцию Else перед инструкцией ElseIf в блочном варианте оператора ветвления?

2 В каких случаях в программе используется полный условный оператор? Как он оформляется? Как он работает (что происходит при его выполнении)?

3 В каких случаях в программе используется неполный условный оператор? Как он оформляется?

11 Самостоятельная работа № 11. Разработка разветленной программы с использованием вложенного оператора IF

Цель работы.

Изучить оператор многозначного ветвления ***If***; разработать программу, реализующую выбор из нескольких альтернатив (более 2).

Методические указания.

Схема алгоритма конструкции оператора многозначных ветвлений ***If*** представлена на рисунке 11.1.

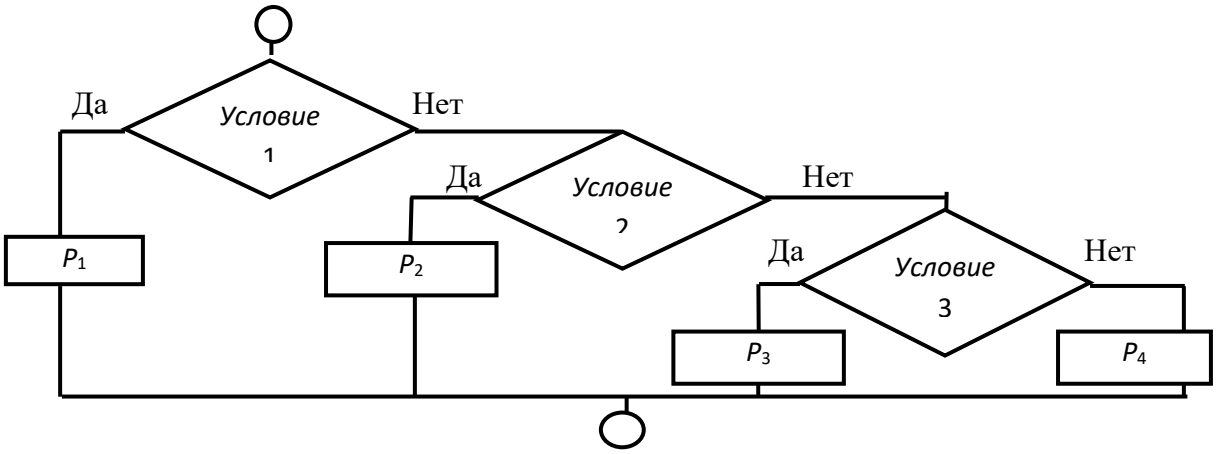


Рисунок 11.1 – Схема алгоритма конструкции оператора многозначных ветвлений ***If***

Синтаксис оператора многозначных ветвлений ***If*** в блочной (структурной) форме следующий:

If<лог.выражение1>Then
    P_1
ElseIf<лог.выражение2>Then
    P_2
Else
    P_3
End If

где *If*, *Then*, *Else*, *End If* – зарезервированные слова;
P_1, P_2, P_3 – операторы.

Алгоритм работы такой конструкции:

– если логическое выражение **1** истинно, то выполняется оператор $\boldsymbol{P_1}$ (или блок операторов), следующий за конструкцией ***Then***, а остальные операторы пропускаются;

– если логическое выражение **1** ложно, то оператор $\boldsymbol{P_1}$ пропускается и анализируется логическое выражение **2**, следующее за ***ElseIf***. Если оно

истинно, то выполняется оператор P_2 (или блок операторов), следующий за ***Then***, а остальные операторы пропускаются;

– оператор P_3 (или блок операторов), следующий за последним ***Else***, выполняется лишь в том случае, если ложны все логические выражения.

Далее представлены примеры использования условного оператора ***If*** и многозначного ветвления, позволяющего определить возможность поступления абитуриента в учреждения образования. На первом шаге происходит ввод пользователем балла, полученного на тестировании. На втором шаге, если полученный балл более 60, приложение выдает сообщение: «Вам можно поступать в ВУЗ». На третьем шаге, если полученный балл более 20, приложение выдает сообщение: «Вам можно поступать в СУЗ». На четвертом шаге, если полученный балл более 5, приложение выдает сообщение: «Вам можно поступать в ПТУ».

```
Subball()
DimballAsVariant
ball = InputBox(«Введите полученный Вами балл на тестировании»)
Ifball>=60 Then
MsgBox («Вам можно поступать в ВУЗ»)
ElseIfball>=20 Then
        MsgBox («Вам можно поступать в СУЗ»)
ElseIfball>=5 Then
        MsgBox («Вам можно поступать в ПТУ»)
EndIf
```

Задание

1 Запросить у пользователя ввод целого числа от 0 до 100 включительно – оценка по 100-балльной системе.

2 С применением оператора многозначных ветвлений ***If*** осуществить преобразование введенной оценки по 100-балльной системе в 5-балльную по шкале, предварительно созданной на листе Excel (таблица 11.1).

Таблица 11.1 – Преобразование оценок

Значение оценки по 100-балльной системе	Значение оценки по 5-балльной системе
От 0 до 20	Оценка 1
От 20 до 40	Оценка 2
От 40 до 60	Оценка 3
От 60 до 80	Оценка 4
От 80 до 100 включ.	Оценка 5
Меньше 0 или больше 100	Ошибка ввода данных

3 По результатам вывести сообщение с введенной оценкой по 100-балльной системе и полученной оценкой по 5-балльной системе либо сообщение об ошибке.

Контрольные вопросы

1 В каких случаях в программе используется вложенный условный оператор?

2 Сколько инструкций ElseIf может быть в блочном варианте оператора ветвления?

3 Нарисуйте алгоритмическую схему выполнения вложенного условного оператора.

12 Самостоятельная работа № 12. Разработка разветленной программы с использованием оператора выбора

Цель работы.

Ознакомиться с оператором выбора SelectCase и закрепить полученные знания на практике.

Методические указания.

При наличии большого количества ветвлений конструкция многозначных ветвлений ***If*** становится тяжёлой для восприятия. В подобных случаях хорошей альтернативой оператору ***If*** служит оператор выбора ***SelectCase***, который позволяет выбрать одно из нескольких возможных продолжений программы.

В то время как ***If ... Then ... Else*** для каждой инструкции ***ElseIf*** оценивает разные выражения, инструкция ***SelectCase*** оценивает выражение только один раз, в начале управляющей структуры. ***SelectCase*** выполняет одну из нескольких групп инструкций в зависимости от значения выражения.

Синтаксис:

SelectCase<*выражение*>
*[**Case***<*списокВыражений-n*>
[инструкции-n]] ...
*[**CaseElse***
[инструкции_else]]
EndSelect

<выражение> − обязательный. Любое числовое выражение или строковое выражение.

<списокВыражений-n> − обязательный при наличии предложения Case.

Список с разделителями, состоящий из одной или нескольких форм следующего вида:

<инструкции-n> − необязательный. Одна или несколько инструкций, выполняемых в том случае, если выражение совпадает с любым компонентом списка <**списокВыражений-n**>;

<инструкции_else> − необязательный. Одна или несколько инструкций, выполняемых в том случае, если выражение не совпадает ни с одним из

предложений ***Case***.

Синтаксис оператора **SelectCase** также может содержать следующие элементы:

– **выражение To выражение**;

– ***Is*** оператор сравнения **выражение**.

Ключевое слово **To** задает диапазон значений. При использовании ключевого слова **To** перед ним должно находиться меньшее значение. Ключевое слово **Is** с операторами сравнения (кроме **Is** и **Like**) задает диапазон значений. Если ключевое слово **Is** не указано, оно вставляется по умолчанию.

Если выражение совпадает с любым выражением из **списка Выражений** в предложении **Case**, выполняются все инструкции, следующие за данным предложением **Case** до следующего предложения **Case**, или для последнего предложения до инструкции **EndSelect**. Затем управление передается инструкции, следующей за **EndSelect**. Если выражение совпадает с выражениями из списка в нескольких предложениях **Case**, выполняется только первый подходящий набор инструкций.

Предложение **CaseElse** задает список **инструкции_else**, которые будут выполнены, если не обнаружено ни одно совпадение выражения и компонента **список Выражений** ни в одном из остальных предложений **Case**. Хотя данное предложение не является обязательным, рекомендуется помещать предложение **CaseElse** в блок **SelectCase**, чтобы предусмотреть неожиданные значения выражения. Если ни в одном предложении **Case список Выражений** не содержит компонента, отвечающего аргументу выражения, и отсутствует инструкция **CaseElse**, выполнение продолжается с инструкции, следующей за инструкцией **EndSelect**. Пример использования оператора **SelectCase** представлен в таблице 12.1.

Таблица 12.1 – Синтаксис и пример использования оператора *SelectCase*

Синтаксис оператора *Select Case*	Пример использования оператора *SelectCase*
SelectCase Ключ Выбора *Case Is выражение* *оператор* *Case диапазон значений* *оператор* *Case диапазон значений* *оператор* *Case диапазон значений* *оператор* *CaseElse* *оператор* *End Select*	*SelectCase vozrast* *Case Is <=7* *Msgbox «Ты дошкольник»* *Case 8 to 16* *Msgbox «Ты учишься в школе»* *Case 17 to 30* *Msgbox «Тебе пора заняться делом»* *Case 31 to 60* *Msgbox «Кто не работает, тот не ест»* *CaseElse* *Msgbox «Вы заслужили отдых»* *EndSelect*

Если значение переменной ***vozrast*** меньше или равно **7**, отображается сообщение *«Ты дошкольник»*; если значение находится в диапазоне от 8 до 16 – сообщение *«Ты учишься в школе»*; если в диапазоне от 17 до 30 – сообщение

«Тебе пора заняться делом»; если в диапазоне от 31 до 60 – сообщение *«Кто не работает, тот не ест»*. Если значение возраста не равно ни одному из предложенных диапазонов значений, выводится сообщение *«Вы заслужили отдых»*.

Из представленного в таблице 12.1 примера видно, что код этой процедуры более прост для восприятия, чем многозначные ветвления ***If***.

Задание

Выполнить задание самостоятельной работы № 11 с применением оператора выбора ***SelectCase***.

Контрольные вопросы

1 В каких случаях в программе используется оператор выбора?

2 В чем преимущество оператора выбора варианта перед многовариантным оператором ветвления?

3 Когда применение оператора SelectCase эффективнее оператора IfThenElseEndIf ?

13 Самостоятельная работа № 13. Разработка программы с использованием оператора выбора FOR. Ввод и вывод одномерного массива

Цель работы.

Изучить оператор цикла ***For***.

Методические указания.

Часто в программах необходимо реализовать определённые операторы несколько раз. В этих случаях организуют циклические вычисления. Алгоритм называется ***циклическим***, если определенная последовательность шагов выполняется несколько раз в зависимости от заданных условий. Циклические алгоритмы могут быть осуществлены с применением следующих операторов цикла: ***For … Next***, ***While … Wend***, ***Do … Loop,*** которые позволяют повторить группу операторов или один оператор заданное количество раз.

Общий вид алгоритма конструкции оператора цикла ***For … Next*** представлен на рисунке 13.1.

Синтаксис конструкции оператора цикла ***For … Next*** следующий:

For i = N_1 To N_2 [Step h]
P_1
…
[Exit For]
P_N
Next i

Тело цикла

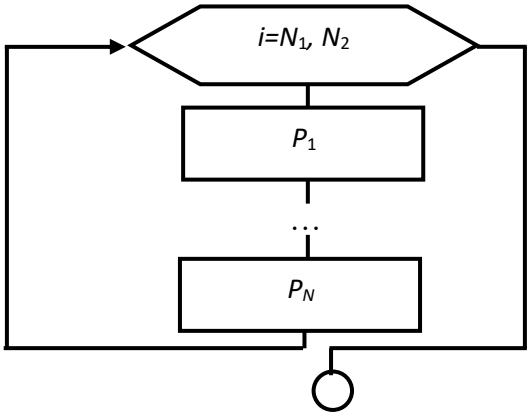


Рисунок 13.1 – Схема алгоритма конструкции оператора цикла *For … Next*

For (для), ***To*** (до), ***Step*** (шаг), ***Exit For*** (выход из ***For***), ***Next*** (следующий) – служебные слова ***VBA***, ***P*1**, ***PN*** – операторы. ***Step*** является необязательным параметром. Если он опущен в программе, то значение параметра ***i*** увеличивается на **1**. Параметр ***Step*** может быть любым действительным числом: как целым, так и дробным, как положительным, так и отрицательным. Оператор ***Exit For*** позволяет выйти из цикла ***For … Next*** до его завершения. Тем самым программа сможет среагировать на определённое событие, не выполняя цикл заданное число раз.

Задание

1 Запросить у пользователя ввод целого числа больше 0.

2 Определить произведение последовательности чисел от 1 до *n* включительно (*n!* – «*n* факториал»).

3 Результат вычисления вывести в виде сообщения.

Пример сообщения: «Факториал 10! = 1 · 2 · 3 · 4 · 5 ·6 · 7 · 8 · 9 · 10 = = 3 628 800».

Контрольные вопросы

1 Может ли тело оператора цикла с параметром не выполниться ни разу?

2 Как должен быть оформлен оператор цикла с параметром, чтобы тело цикла выполнялось при уменьшающихся значениях параметра цикла?

3 Чему равно количество повторений тела оператора цикла с параметром, если параметр цикла принимает:

а) все целые значения от 1 до 10;

б) все значения от 10 до 100 с шагом 7;

в) все значения от а до б с шагом step?

4 Можно ли в теле оператора цикла использовать условный оператор?

5 Какие Вы знаете операторы для принудительного (преждевременного) выхода из оператора цикла?

14 Самостоятельная работа № 14. Обработка одномерного массива

Цель работы.

Изучить одномерные массивы, разработать программу с вводом, выводом и обработкой одномерных массивов.

Методические указания.

Массив – совокупность однотипных элементов данных (чисел, логических данных, символов), которой при обработке присвоено определенное имя. Совокупность представлена набором переменных с одним именем и с разными индексами. Массивы бывают разной размерности: одномерные, двумерные, трехмерные, …, *n*-мерные.

Массивы бывают статические и динамические. Статическими называются массивы, количество элементов в которых заранее известно и не изменяется в ходе выполнения программы. Динамические массивы – массивы, в которых либо не известно начальное количество элементов, либо размерность массива (количество элементов) изменяется при выполнении программы.

Описание массивов:

1) ***одномерный статический массив***

Dim<*имя массива*>**(**<*начальное значение индекса*>**To**<*конечное значение индекса*>**)** [**As**<*тип элементов массива*>]

или

Dim<*имя массива*>**(**<*количество элементов массива*>**)** [**As**<*тип элементов массива*>];

2) ***двумерный статический массив***

Dim<*имя массива*>**(**<*начальное значение индекса по строкам*>**To**<*конечное значение индекса по строкам* >**,**< *начальное значение индекса по столбцам*>**To**< *конечное значение индекса по столбцам*>**)** [**As**<*тип элементов массива*>]

или

Dim<имя массива>**(**<количество строк>**,**<количество столбцов>**)** [**As**<тип элементов массива>].

Первый способ отличается от второго тем, что в первом случае указывается индекс первого и последнего элементов, во втором же – только количество элементов, нумерация которых может начинаться как с 0, так и с 1. Это зависит от опции **Base** (задает базовый индекс). Если опция не указана, то нумерация элементов массива начинается с нуля. Для изменения базового индекса в начале листа модуля необходимо написать **OptionBase 1**.

Пример 1:

− DimA(1 To 10) AsInteger – массив А состоит из 10 элементов целого типа, индексы которых 1, 2, …, 10;

− Dim A(10) AsInteger – массив состоит из 10 значений целого типа. Индексация зависит от опции **Base**. Если опция не указана, то номера элементов − от 0 до 9, если же указана (т. е. вначале модуля записано **OptionBase 1**), то номера элементов изменяются от 1 до 10;

3) ***динамический массив***

Dim<имя массива>**( )** [**As**<тип элементов массива>].

После определения количества элементов массива выполняется его переопределение:

ReDim<имя массива>**(**<задается размерность массива (одномерного/двумерного >**)**.

Пример 2

DimA( ) AsSingle – динамический массив А вещественных элементов *n* = 7;

ReDimA (1 Ton) – переопределение одномерного массива из *n* значений;

ReDimA (5,*n*) – переопределение двумерного динамического массива, состоящего из пяти сток и *n* столбцов (начало индексации элементов определяется по опции **Base**).

Обращение к элементу массива осуществляется следующим образом: указывается имя массива, а затем в круглых скобках − номер элемента в массиве. Если массив двумерный – указывается вначале номер строки, затем через запятую номер столбца.

Примеры объявления массивов:

DimA (12) AsInteger ‘ одномерный массив из 12 элементов типа Integer

Dim A(1 To 12) As Integer

DimB (3, 3) AsSingle ‘ двумерный массив элементов 3 × 3 (матрица) типа Single

Dim B(1 To 3, 1 To 3) As *Single*

Причем по умолчанию первый элемент массива А будет А(0), а последний – А(11). В этом случае говорят, что 0 – базовый индекс. Можно изменить базовый индекс, написав в начале листа модуля инструкцию OptionBase 1. После этого индексы массивов А и В будут начинаться с единицы.

Массив в программе определяется поэлементно.

Удобным способом определения одномерных массивов является функция Array, преобразующая список элементов, разделенных запятыми, в вектор из этих значений и присваивающая их переменной тип Variant.

Задание

1 Объявить одномерный массив размерностью 10 элементов целочисленного типа.

2 Инициализировать все элементы массива произвольными числами: $A(1) = 5$ и т. д.

3 Реализовать расчет среднего арифметического всех чисел, содержащихся в массиве, с помощью оператора цикла.

4 Вывести результат расчета в виде текстового сообщения:

«Среднее арифметическое: ____».

Контрольные вопросы

1 Опишите синтаксис объявления массива с использованием оператора Dim.

2 Какое значение нижнего индекса элемента массива принято в VBA по умолчанию? Каким образом можно для него задать значение 1?

3 Как установить или изменить размерность многомерного динамического массива?

4 Как изменить размерность динамического массива без потери имеющихся значений его элементов?

5 Сколько индексов характеризуют конкретный элемент двумерного массива?

15 Самостоятельная работа № 15. Разработка программы с вложенными циклами. Обработка многомерных массивов

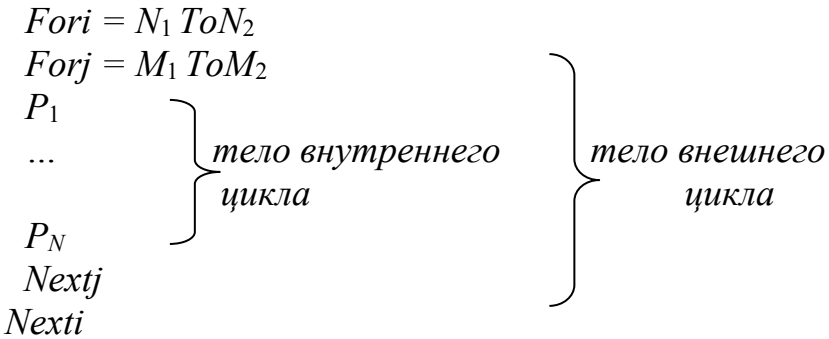
Цель работы.

Изучить вложенные операторы цикла ***For***.

Методические указания.

Если телом цикла является циклическая структура, то такие циклы называются вложенными. Цикл, содержащий в себе другой цикл, – внешний, а цикл, содержащийся в теле другого цикла, – внутренний.

Синтаксис конструкции вложенных операторов цикла ***For…Next*** следующий:



Общий вид алгоритма конструкции вложенных операторов цикла ***For … Next*** представлен на рисунке 15.1.

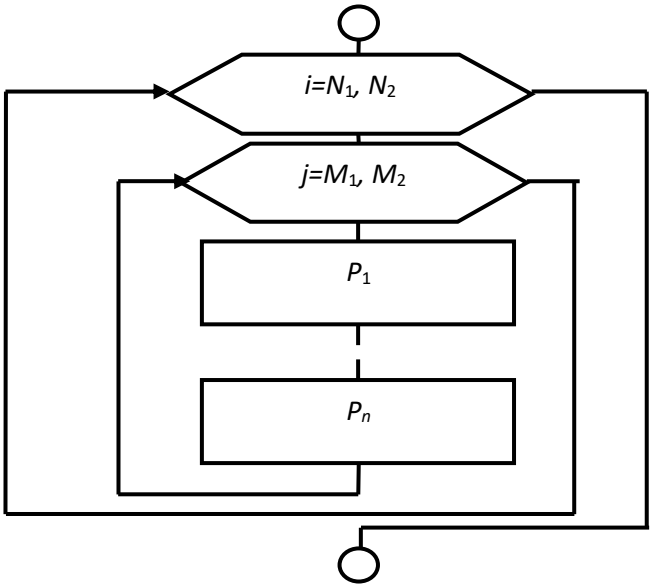


Рисунок 15.1 – Схема алгоритма конструкции вложенных операторов цикла *For … Next*

При первом вхождении в цикл параметр внешнего цикла ***i*** принимает значение, равное N_1, и управление передаётся во внутренний цикл, в котором параметр цикла ***j*** принимает значение, равное M_1, и выполняется оператор (операторы), который записан во внутреннем цикле. Затем параметр внутреннего цикла ***j*** увеличивается на **1**, и вновь повторяется тело цикла.

Операторы P_1, P_N будут реализовываться до тех пор, пока параметр цикла ***j*** не станет больше величины M_2. Затем параметр внешнего цикла ***i*** увеличивается на **1**, и вновь начинает свою работу внутренний цикл, в котором параметр цикла ***j*** будет изменяться от M_1 до M_2, и при каждом прохождении цикла будут выполняться операторы P_1 и P_N. Внешний цикл закончит свою работу, когда параметр цикла ***i*** станет больше величины N_2.

Задание

С помощью вложенных операторов цикла ***For…Next*** реализовать запись на рабочий лист ***Excel*** матрицы размерностью $n \times n$ (значение числа ***n*** запросить у пользователя), элементы которой должны являться произведением номера строки на номер колонки.

Контрольные вопросы

1 В каких случаях используются вложенные операторы цикла?

2 Как оформляются вложенные операторы цикла?

3 Нарисуйте алгоритмическую схему выполнения вложенных операторов цикла.

4 Вложенный цикл образован двумя операторами цикла с параметром. Что является телом?

5 Может ли внешний оператор вложенного цикла:

а) не выполниться ни разу;

б) выполняться бесконечное число раз (или до того момента, когда пользователь прервет его выполнение)?

16 Самостоятельная работа № 16. Разработка программы с использованием циклов While

Цель работы.

Научиться применять операторы цикла с предусловием и постусловием.

Методические указания.

Циклы данного вида используются, когда заранее неизвестно, сколько раз будет выполняться тело цикла.

Циклы с предусловием (DoWhile … Loop, DoUntil … Loop) представлены в таблице 16.1, а операторы циклов с постусловием (Do … LoopWhile, Do … LoopUntil) – в таблице 16.2.

Таблица 16.1 – Циклы с предусловием

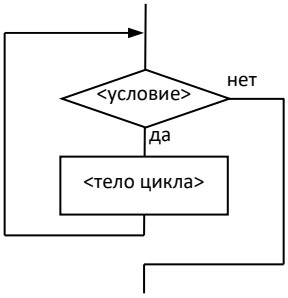
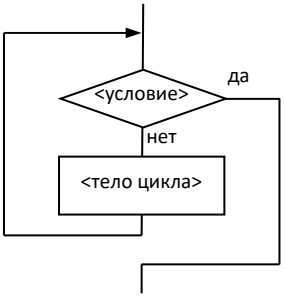
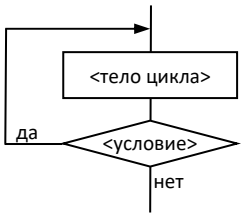
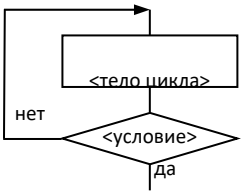
Синтаксис	**DoWhile**<*условие*> <*тело цикла*> [**ExitDo**] … ***Loop***	**DoUnlil**<*условие*> <*тело цикла*> [**ExitDo**] … ***Loop***
Порядок выполнения	**<***Тело цикла***>** будет выполняться в том случае, когда <*условие*> имеет значение Истина (TRUE) (цикл продолжается при истинном значении <*условия***>**). Если <*условие*> ложно (FALSE), то выполняются операторы, стоящие за циклом. В первом случае есть возможность досрочного выхода из цикла (это реализовано через **ExitDo)**	**<***Тело цикла***>** выполняется до тех пор, пока <*условие*> не примет значение Истина (цикл продолжается при ложном значении <*условия*>). Есть возможность досрочного выхода из цикла (это реализовано через **ExitDo**)
Изображение в блок-схемах		

Таблица 16.2 – Циклы с постусловием

Синтаксис	**Do** <*телоцикла*> [**Exit Do**] … **Loop While**<*условие*>	**Do** <*тело цикла*> [**Exit Do**] … **Loop Until <***условие*>
Порядок выполнения	<*Тело цикла*> будет выполняться в том случае, когда <*условие*> имеет значение Истина (цикл продолжается при истинном значении <*условия*>). Если <*условие*> ложно, то выполняются операторы, стоящие за циклом. Предоставлена возможность досрочного выхода из цикла (это реализовано через **ExitDo**)	**<***Тело цикла*> выполняется до тех пор, пока <*условие*> не примет значение Истина (цикл продолжается при ложном значении <*условия*>). Есть возможность досрочного выхода из цикла (это реализовано через **ExitDo**)
Изображение в блок-схемах		

Отличие циклов с предусловием от циклов с постусловием заключается в том, что тело цикла первых может не выполниться ни разу, в то время как тело цикла с постусловием всегда выполнится хотя бы один раз.

Пример − Организовать ввод последовательности целых чисел, пока их сумма не превысит целого числа m. Вывести количество введенных чисел. Текст программы с пояснениями представлен в таблице 16.3.

Таблица 16.3 – Программа по вводу последовательности целых чисел

Текст программы	Описание
Public Sub prog1()	–
Dim x As Integer, m As Integer	–
Dim s As Integer, i As Integer	–
m=InputBox(«Введите число m»)	Ввод предельного числа.
i = 1	Номер вводимого числа последовательности.
s =InputBox(«Введите 1 число»)	Ввод первого числа последовательности.
Do While s <= m	Цикл с предусловием: тело цикла выполняется, пока условие s<=m имеет значение Истина (TRUE).
i = i + 1	Тело цикла: увеличение номера на 1.
x=InputBox(«Введите» &i& «число»)	Ввод очередного (*i*-го) значения.
s = s + x	Добавление введенного значения к предыдущему значению суммы.
Loop	Конец тела цикла.
MsgBox («Количествочисел» &i)	Вывод значения переменной *i*
EndSub	

Задание

1 В цикле реализовать запрос пароля: «Введите пароль:».

2 Сравнить введенный пользователем пароль с заданным.

3 Цикл должен выполняться до тех пор, пока пароль не совпадет.

4 Если пароль совпадает, цикл завершается и программа выдает сообщение: «Пароль принят!».

Контрольные вопросы

1 В каких случаях используются операторы цикла с условием?

2 В чём основное отличие между циклами с предусловием и с постусловием?

3 В каких случаях целесообразно использовать циклы с предусловием, циклы с постусловием и циклы по счётчику?

4 Может ли тело оператора цикла с предусловием:

а) не выполниться ни разу;

б) выполняться бесконечное число раз (или до тех пор, когда пользователь прервет его выполнение)?

17 Самостоятельная работа № 17. Использование подпрограмм, возвращающих значения (функции)

Цель работы.

Изучить функции и их применение.

Методические указания.

В программировании сложный код программы разбивают на подпрограммы. *Подпрограмма* – это группа операторов, выполняющих законченное действие. *Основная программа* – программа, реализующая основной алгоритм решения задачи и содержащая в себе обращения к подпрограммам (вызов подпрограмм). В точке вызова функции выполнение программы переходит к подпрограмме и, выполнив все действия подпрограммы, возвращается в основную программу. В подпрограмме могут быть объявлены собственные переменные, а также параметры подпрограммы для обмена значений с основной программой. В VBA существуют два типа подпрограмм: подпрограммы-функции и подпрограммы-процедуры. Функция, в отличие от процедуры, возвращает значение и может входить в состав выражений. Сравнительная характеристика процедур и функций представлена в таблице 17.1.

<*Список параметров*> отличается от <*списка аргументов*> тем, что первый указывается при описании подпрограммы, второй – при ее вызове в основной программе.

<*Список параметров*> позволяет передать в подпрограмму требуемые значения из вызывающей программы и имеет следующий синтаксис:

[ByRef|ByVal] *<имя параметра1>* ***[As****<Тип>****], [ByRef|ByVal]*** *<имя параметра2>* ***[As****<Тип>****], [ByRef|ByVal]*** *<имя параметра3>* ***[As****<Тип>****]****, ...*

<Тип> позволяет явно задать тип передаваемых значений. Если тип опущен, то по умолчанию принимает значение Variant.

Таблица 17.1 – Сравнительная характеристика процедур и функций

Тип подпрограммы	Описание	Вызов в основной программе
Подпрограмма-процедура	**Sub***<имя процедуры>***(***[<список параметров>]***)** *<операторы>* **[ExitSub]** *<операторы>* **EndSub**	*<имя процедуры><список аргументов>*; **Call***<имя процедуры>*[**(***<список аргументов>***)**]
Подпрограмма-функция	**Function<***имя функции>* *[(<список параметров>])* *[As<тип функции>]* *<операторы>* **[ExitFunction]** *<операторы>* **<***имя функции>* = **<***выражение>* **EndFunction**	<имя функции> (*<список аргументов>*)

Ключевое слово ***ByVal*** (передача по значению) используется в тех случаях, когда желают, чтобы изменение параметров внутри процедуры не приводило к изменению соответствующих аргументов процедуры в основной программе. Использование ***ByRef*** (передача по ссылке) или, если ключевое слово опущено, означает, что при изменении параметров внутри процедуры происходит изменение соответствующих параметров в основной программе.

<Список аргументов> перечисляется через запятую. Количество и типы параметров и аргументов должны соответствовать. Аргументы, соответствующие параметрам с ключевым словом ***ByRef*** (или по умолчанию), должны быть переменными.

Для выхода из подпрограммы и возврата в основную программу, опуская оставшиеся операторы, используется **ExitSub** (в процедурах) и **ExitFunction** (в функциях).

Пример 1 − Вычислить сумму членов ряда $\sum_{i=1}^{n} \frac{1}{i!}$, где $i!$ – факториал числа i (произведение натуральных чисел от 1 до i). Текст программы с пояснениями представлен в таблице 17.2. Схема алгоритма программы prog3 и подпрограммы Faktor приведена на рисунке 17.1.

Таблица 17.2 – Программа вычисления суммы членов ряда

Текст программы	Описание
Public Sub prog3() *Dim i As Integer, n As Integer* *Dim s As Double* *n = CInt(InputBox(«Введите n»))* *For i = 1 To n* *s = s + 1 / faktor(i)* *Nexti* *MsgBox s* *End Sub*	– – – – – При вычислении искомой суммы производится вызов функции factor и передается ее аргумент *i*
Public Function faktor(x As Integer) As Long *faktor = 1* *For i = 1 To x* *faktor = faktor * i* *Next i* *EndFunction*	При описании функции типу ее результата присваивается тип длинный целый (Long), т. к., например, 10!=40320, что выходит за диапазон типа Integer. При выполнении функции результат присваивается ее имени

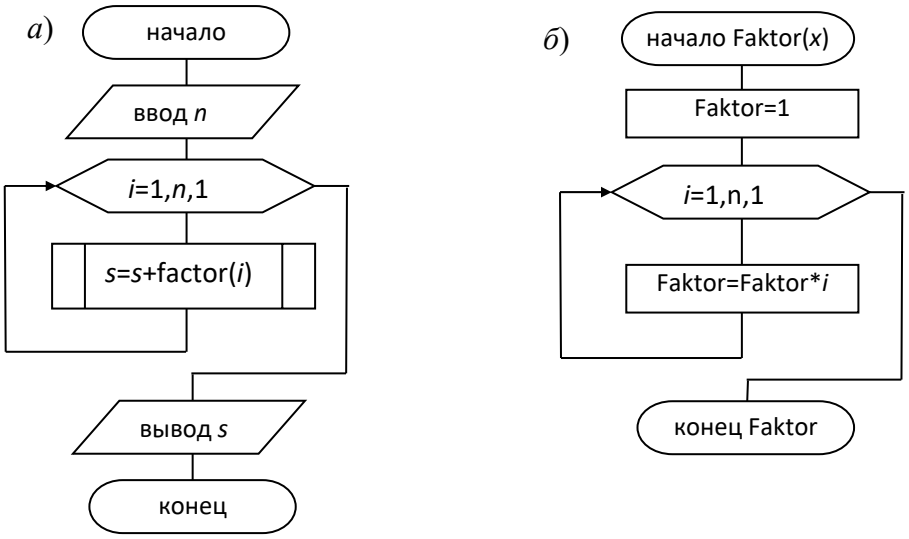


Рисунок 17.1 – Схема алгоритма программы prog3 (*а*) и подпрограммы Faktor (*б*)

Пример 2 − Построить график функции $y = \frac{\sin x}{x^2 + 1}$ на отрезке [a;b] с шагом 0,1. Построение графика организовать процедурой. Текст программы с пояснениями представлен в таблице 17.3. Блок-схема реализации вычисления приведена на рисунке 17.2.

Таблица 17.3 – Программа построения графика функции

Текст программы	Описание
Public Sub prog4() *Dim a As Double b As Double, h As Double* *Worksheets(1).Range(«A:B»).Select* *Selection.Clear* *a = CDbl (InputBox(«Введите b»))* *b = CDbl (InputBox(«Введите b»))* *j = 1* *For i = 1 To b Step 0.1* *Worksheets(1).Range(«A»& j) = i* *Worksheets(1).Range(«B»& j) = Sin(i) / (i ^ 2 + 1)* *j = j + 1* *Next i* *график* *End Sub*	– – Выделение двух столбцов (А и В) первого листа Очистка от данных выделенного диапазона *j* используется для задания номера строки при выводе данных В цикле *For…Next* выводятся на лист Excel данные для построения диаграммы Вызывается процедура − график
Sub график() *n=Application.CountA(Worksheets (1).Range(«A:A»))* *Range(«A1:B»&CStr(n)).Select* *Charts.Add* *ActiveChart.ChartType = xlXYScatterSmoothNoMarkers* *ActiveChart.SetSourceData* *Source:=Sheets(«Лист1»).Range(«A1:B»&CStr(n)),* *PlotBy:= xlColumns* *EndSub*	При создании процедуры графика вначале был создан макрос в Excel, а затем макрос отредактирован. В частности, первая команда в данной процедуре определяет количество заполненных строк на листе в столбце А и это значение присваивается переменной *n*

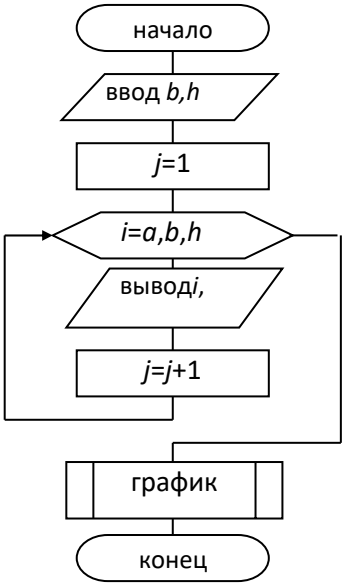


Рисунок 17.2 − Схема алгоритма программы prog4

Задание

1 Разработать функцию, осуществляющую расчет стоимости товара с НДС. В качестве аргументов данная функция должна принимать стоимость товара

без НДС и ставку НДС. Возвращаемое функцией значение – стоимость товара с НДС.

Схема алгоритма макроса не изображается.

2 Создать новую процедуру и задать в ней вызов разработанной функции с целью расчета стоимости товара с НДС, указав при вызове необходимые параметры (стоимость товара без НДС и ставку НДС). Полученную стоимость вывести на экран с помощью сообщения:

«*Стоимость товара с НДС:* ____».

Контрольные вопросы

1 Понятие и синтаксис функций (понятие, объявление, определение, вызов).

2 Чем отличаются локальные переменные от глобальных?

3 В какой последовательности должны располагаться имена параметров в операторах вызова функции?

4 Как объявить тип данных для аргументов функции?

5 Как организовать преждевременный выход из функции?

18 Самостоятельная работа № 18. Использование подпрограмм, не возвращающих значения

Цель работы.

Изучить процедуры и их применение.

Задание

Используя методический материал, изложенный в самостоятельной работе № 17, создать новую процедуру и задать в ней вызов любой из процедур, разработанных в рамках самостоятельных работ № 9–16.

Контрольные вопросы

1 В чем различие между функцией и процедурой?

2 Понятие и синтаксис процедур (понятие, объявление, определение, вызов).

3 Могут ли в одной программе процедура и функция иметь одно и то же имя?

19 Самостоятельные работы № 19–20. Использование подпрограмм, возвращающих значения через параметры

Цель работы.

Изучить процедуры и их применение.

Задание

Используя методический материал, изложенный в самостоятельной работе № 17, создать новую процедуру, с помощью которой вычисляются параметры геометрической фигуры (например, длина окружности и площадь) по заданному значению радиуса круга. Вычисленные значения передаются в головную программу, где используются для дальнейших вычислений и вывода результата.

Контрольные вопросы

1 В чем различие между возвращаемыми и невозвращаемыми параметрами, как они объявляются?

2 Можно ли изменять порядок следования аргументов процедуры при ее вызове?

3 Можно ли изменять количество аргументов процедуры при ее вызове?

20 Самостоятельная работа № 21. Использование структурированных типов данных

Цель работы.

Ознакомиться с возможностями создания пользовательских типов данных и их использования в программах.

Методические указания.

VBA позволяет создавать специальные, или пользовательские, типы данных, которые называются структурированными типами данных. Определенный пользователем тип данных может облегчить управление некоторыми типами данных. Например, если приложение обрабатывает сведения о студенте, то можно создать пользовательский тип данных с названием StudentInfo.

```
Type StudentInfo
Family As String
Name As String
GroupeAs String
Curs As Integer
EndType
```

Пользовательские типы данных определяются в верхней части модуля перед началом процедур или в отдельном модуле. Если пользовательский тип данных уже создан, для объявления переменной этого типа применяется оператор Dim. Обычно пользовательский тип данных определяется для массивов.

DimStudents (1 To 100) AsStudentInfo

Все 100 элементов этого массива состоят из четырех компонентов (как указано в пользовательском типе данных – StudentInfo).

Сослаться на конкретный компонент элемента можно с помощью составного имени следующим образом:

Students(1).Family = "Petrov"
Students(1).Name = "Ivan"
Students(1).Group = “ASU-161”
Students(1).Curs = 3

Задание

Разработать программу для работы с графическими объектами – треугольниками, состоящими из трёх линий, каждая из которых имеет координаты точки начала и точки окончания, также величину своей длины. Программа должна обеспечить задание пространственного положения треугольников и вычисление их параметров: площади и периметра.

Контрольные вопросы

1 Где объявляются пользовательские типы данных? Можно ли его объявить в теле процедуры или функции как локальный?

2 Как задается тип отдельного компонента структуры?

3 Как получить доступ к отдельному компоненту структуры?

21 Самостоятельная работа № 22. Разработка программ с использованием пользовательских форм с простейшими элементами управления

Цель работы.

Ознакомиться с возможностями создания интерфейса в VBA, некоторыми его элементами, их свойствами и методами.

Методические указания.

По своей сути форма (или пользовательская форма) представляет собой диалоговое окно, в котором можно размещать различные элементы управления. В приложении может быть как одна, так и несколько форм. Новая форма добавляется в проект выбором команды **Вставка (Insert) → UserForm**.

В VBA имеется обширный набор встроенных элементов управления.

Используя этот набор и редактор форм, нетрудно создать любой пользовательский интерфейс, который будет удовлетворять всем требованиям, предъявляемым к интерфейсу в среде Windows. Элементы управления являются объектами. Как любые объекты, они обладают свойствами, методами и событиями. Элементы управления создаются при помощи Панели элементов, которая отображается на экране либо выбором команды **Вид (View) →Панель элементов (Toolbox)**, либо нажатием кнопки панели инструментов **Standard.** На этой панели представлены кнопки, позволяющие конструировать элементы управления. Для создания элементов управления служат все кнопки панели инструментов, за исключением кнопки **Выбор объекта .** Щелкнув по кнопке **Выбор объекта**, можно выбрать уже созданный в форме элемент управления для последующего его редактирования (изменения размеров или редактирования).

В таблице 21.1 представлен список основных элементов управления и соответствующих кнопок панели элементов.

Таблица 21.1 – Список основных элементов управления

Элемент управления	Имя	Кнопка для создания	Элемент управления	Имя	Кнопка для создания
Поле	TextBox		Переключатель	OptionButton	
Надпись	Label		Флажок	CheckBox	
Кнопка	CommandButton		Выключатель	ToggleButton	
Список	ListBox		Рамка	Frame	
Поле со списком	ComboBox		Рисунок	Image	
Полоса прокрутки	ScrolBar		Набор страниц	MultiPage	
Счетчик	SpinButton		Набор вкладок	TabStrip	

Для размещения элемента управления на лист или в форму необходимо нажать соответствующую кнопку на панели элементов и с помощью мыши перетащить рамку элемента управления в нужное место. После этого элемент управления можно перемещать, изменять его размеры, копировать в буфер обмена, вставлять из буфера обмена и удалять из формы.

В таблице 21.2 представлены основные общие свойства элементов управления.

В таблице 21.3 представлены наиболее часто используемые свойства элементов управления.

Таблица 21.2 – Основные общие свойства элементов управления

Свойство	Описание
Caption	Надпись, отображаемая при элементе управления
AutoSize	Допустимые значения: True (устанавливает режим автоматического изменения размеров элемента управления так, чтобы на нем полностью помещался текст, присвоенный свойству Caption) и False (в противном случае)
Visible	Допустимые значения: True (элемент управления отображается во время выполнения программы) и False (в противном случае)
Enabled	Допустимые значения: True (пользователь вручную может управлять элементом управления) и False (в противном случае)
Height и Width	Устанавливают геометрические размеры объекта (высоту и ширину)
Left и Top	Устанавливают координаты верхнего левого угла элемента управления, определяющие его местоположение в форме
ControlTipText	Устанавливает текст в окне всплывающей подсказки, связанной с элементом управления. В следующем примере элементу управления CommandButton назначен текст всплывающей подсказки «Это кнопка»: CommandButton1.ControlTipText = «Это кнопка»
BackColor, ForeColor и BorderColor	Устанавливают цвет заднего и переднего плана элемента управления, также его границы
BackStyle	Устанавливает тип заднего фона
BorderStyle	Устанавливает тип границы. Допустимые значения: fmBorderStyleSingle (граница в виде контура); fmBorderStyleNone (граница невидима)
SpecialEffect	Устанавливает тип границы. Отличается от свойства BorderStyle тем, что позволяет установить несколько типов, но одного цвета. BorderStyle позволяет установить только один тип, но различных цветов
Picture (создание картинки)	Внедряет картинку на элемент управления

Таблица 21.3 – Свойства элементов управления

Свойство	Описание
TextBox (поле) используется для ввода текста пользователем или для вывода из него результатов расчетов программ	
Text	Возвращает текст, содержащийся в поле
Multiline	Допустимые значения: True (устанавливает многострочный режим ввода текста в поле) и False (однострочный режим)
WordWrap	Допустимые значения: True (устанавливает режим автоматического переноса) и False (в противном случае)
Label(надпись) используется для отображения надписей, например, заголовков элементов управления, не имеющих свойства Caption	
Caption	Возвращает текст, отображаемый в надписи
Multiline	Допустимые значения: True (устанавливает многострочный режим ввода) и False (однострочный режим)
WordWrap	Допустимые значения: True (устанавливает режим автоматического переноса) и False (в противном случае)

Окончание таблицы 21.3

Свойство	Описание
CommandButton (кнопка) используется для инициирования выполнения некоторых действий, вызываемых нажатием кнопки, например, запуск программы или остановка ее выполнения, печать и т. д.	
Caption	Возвращает текст, отображаемый на кнопке
Default	Задаёт кнопку по умолчанию, т. е. устанавливает ту кнопку, для которой действия, связанные с ней, будут выполняться при нажатии клавиши <Enter>
Cancel	Допустимые значения: True (устанавливаются отменяющие функции для кнопки, т. е. нажатие клавиши <Esc>приводит к тем же результатам, что и нажатие кнопки) и False (в противном случае)
Accelerator	Назначает клавишу, при нажатии на которую одновременно с клавишей <Alt> происходит запуск действий, связанных с кнопкой. Например, CommandButton1.Accelerator=«C»

Пример – В качестве примера работы с формой сконструируем простое приложение, вычисляющее значение функции, например, Cos(x).

Перейдем в VBA и, выполнив команду **Insert (Вставка) →UserForm**, добавим в проект форму. Расположим на форме следующие элементы управления: Label, TextBox, CommandButton (рисунок 22.1).

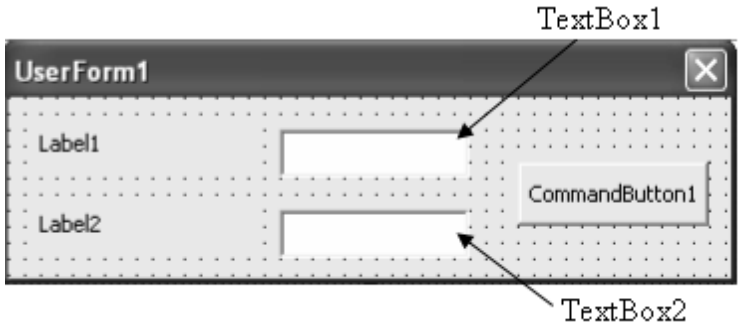


Рисунок 21.1 – Вид формы в режиме конструктора

В таблице 21.4 приведено описание создаваемой формы.

Форма создана, функция каждого элемента управления известна. Для написания кода программы, связанного с пользовательской формой, достаточно дважды щелкнуть, например, кнопку CommandButton1. Откроется редактор кода на листе модуля UserForm1. Более того, он откроется на том месте, где программируются действия, связанные с элементом управления, который был дважды нажат. Если код еще не набран, то при открытии редактора кода появятся инструкции заголовка и окончания процедуры, которая будет связана с элементом управления. Редактор кода представлен в таблице 21.5.

После конструирования формы и написания кода в модуле формы выберем команду **Run→RunSub/UserForm**, либо нажмем клавишу **<F5>,** либо кнопку панели инструментов Standard, и форма отобразится поверх активного рабочего листа Excel. Введем значение аргумента и нажмем кнопку *OK*. Вид полученной пользовательской формы представлен на рисунке 21.2.

Таблица 21.4 – Описание создаваемой формы

Элемент управления	Предназначение
CommandButton1 (кнопка)	При нажатии на кнопку запускается процедура обработки события (PrivateSub CommandButton1_Click()), которая считывает значение аргумента из поля TextBox1. Проверяется, введено ли в это поле число. Если введено не число, то на экране отображается соответствующее сообщение, прерывается выполнение процедуры, и фокус (курсор) устанавливается на поле TextBox1, предлагая исправить вводимые данные. Если введено число, то находится значение функции при введенном значении аргумента, результат выводится в TextBox2
Label1 (надпись)	Пояснительная надпись для поля ввода
TextBox1 (поле)	Поле для ввода пользователем значения аргумента
TextBox2 (поле)	В это поле будет выводиться значение функции. Поле сделаем недоступным для пользователя, т. е. пользователь не сможет ни ввести, ни скорректировать данные в этом поле

Таблица 21.5 – Пример программы с применением элементов управления

Текст программы	Описание
Private Sub CommandButton1_Click() *If Not IsNumeric(TextBox1.Text) Then* *MsgBox «Аргумент должен быть числом», _vbExclamation* *TextBox1.SetFocus* *Exit Sub* *End If* *x = CDbl(TextBox1.Text)* *y = Cos(x)* *TextBox2.Text = CStr(y)* *End Sub*	 Проверка является ли введённое значение числом. Вывод окна сообщения. Фокус (курсор) устанавливается на поле TextBox1. Досрочный выход из процедуры. При считывании числа из поля ввода при помощи функции CDbl строковый тип, возвращаемый свойством Text, преобразуется в числовой. Чтобы вывести результат в поле, переводим число в строковый формат при помощи функции CStr
Private Sub UserForm_Initialize() *UserForm1.Caption = «Значение функции Cos(x)»* *Label1.Caption = «Аргумент»* *Label2.Caption = «Значение функции»* *CommandButton1.Caption = «OK»* *TextBox2.Enabled = False* *EndSub*	Процедура UserForm_Initialize конструирует форму до ее загрузки. Инструкция устанавливает текст, отображаемый в строке заголовка формы3 *Label1.Caption = «Аргумент»* *Label2.Caption = «Значение функции»* *CommandButton1.Caption = «OK»* *TextBox2.Enabled = False* *EndSub*

После конструирования формы и написания кода в модуле формы выберем команду **Run→RunSub/UserForm**, либо нажмем клавишу **<F5>,** либо кнопку панели инструментов Standard, и форма отобразится поверх активного рабочего

листа Excel. Введем значение аргумента и нажмем кнопку *OK*. Вид полученной пользовательской формы представлен на рисунке 21.2.

Рисунок 21.2 – Вид пользовательской формы

Задание

Выполнить выше приведенные примеры.

Контрольные вопросы

1 Как создать графический интерфейс своего приложения с помощью VBA?
2 Назовите этапы создания форм.
3 Что такое элемент управления?
4 Какое назначение элементов управления Label, TextBox и CommandButton?

Список литературы

1 **Гвоздева, В. А**. Информатика, автоматизированные технологии и системы: учебник / В. А. Гвоздева. – Москва: ФОРУМ; ИНФРА-М, 2021. – 542 с.

2 **Скитер, Н. Н**. Информационные технологии: учебное пособие / Н. Н. Скитер, А. В. Костикова. – Волгоград: ВолгГТУ, 2019. – 96 с.