Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

А. П. ЛАЩЕНКО, С. А. БОРИСЕВИЧ, С. А. ОСОКО

КОМПЬЮТЕРНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

В 2-х частях

Часть 1

**Лабораторный практикум для студентов специальностей
1-25 01 07 «Экономика и управление на предприятии»,
1-26 02 02 «Менеджмент», 1-26 02 03 «Маркетинг»**

Минск 2018

УДК 004.9(076.5)
ББК 32.973я73
Л32

Рассмотрен и рекомендован к изданию редакционно-издательским советом Белорусского государственного технологического университета

Рецензенты:
заведующая кафедрой информационных технологий
УО «Белорусский государственный экономический университет»
кандидат технических наук, доцент *М. Н. Садовская*;
доцент кафедры информационных систем и технологий
УО «Белорусский государственный технологический университет»
кандидат технических наук, доцент *Н. Н. Пустовалова*

Лащенко, А. П.

Л32 Компьютерные информационные технологии. В 2 ч. Ч. 1 : лабораторный практикум для студентов специальностей 1-25 01 07 «Экономика и управление на предприятии», 1-26 02 02 «Менеджмент», 1-26 02 03 «Маркетинг» / А. П. Лащенко, С. А. Борисевич, С. А. Осоко. – Минск : БГТУ, 2018. – 119 с.

Издание содержит основные понятия, методические указания для проведения лабораторных работ и подготовки студентов к экзамену по дисциплине «Компьютерные информационные технологии». Практикум посвящен базовым вопросам компьютерной обработки информации и включает шесть самостоятельных разделов. Все разделы сопровождаются и поддерживаются конкретными практическими примерами, которые облегчают усвоение материала студентами.

УДК 004.9(076.5)
ББК 32.973я73

ПРЕДИСЛОВИЕ

Цель обучения студентов основам компьютерных сетей – обеспечить знание теоретических и практических основ в организации и функционировании компьютерных сетей, умение применять в профессиональной деятельности распределенные данные, прикладные программы и ресурсы сетей.

В настоящее время персональные компьютеры в автономном режиме практически не используются, их, как правило, объединяют в компьютерные сети. Подробнее о них можно узнать в приложении А.

Одной из самых востребованных программ, установленных на компьютере, является текстовый редактор. Эта программа, которая позволяет производить набор текста, его редактирование, а также визуальное оформление. Обзор наиболее популярных текстовых редакторов представлен в приложении Б.

Не менее востребованной программой является электронная таблица (табличный процессор). Эта компьютерная программа позволяет проводить вычисления с данными, представленными в виде двумерных массивов. Краткая инструкция по работе с электронной таблицей Microsoft Excel приводится в приложении В.

Возможности, предоставляемые электронными таблицами, слабо подходят для выполнения серьезных конструкторских расчетов. Для решения таких задач используют математические пакеты. Общие сведения о наиболее известных в настоящее время пакетах **Maple**, **MathCad**, **Mathematica** и **MatLab** отражены в приложении Г.

Разработка программ в системе Windows состоит из двух последовательных этапов:

- сначала на экране размещают элементы управления для организации необходимого графического интерфейса прикладной программы на экране компьютера;
- затем для определенных элементов управления интерфейса разрабатывается программный код, по которому будет выполняться работа.

В данном практикуме рассматривается программирование на языке Visual Basic компании Microsoft. Основные сведения об этом языке приведены в приложении Д.

КОМПЬЮТЕРНЫЕ СЕТИ

Лабораторная работа № 1
РАБОТА В СЕТИ

Цель работы: научиться работать в сети.

Методические указания

Создайте документ в приложении Word. Сохраните созданный файл на сервере. Отправьте этот файл в папку **Мои документы** и отредактируйте его с дальнейшим сохранением.

1. Подключите ПК к серверу.

Указание. Используйте диалоговое окно (рис. 1).

Рис. 1. Окно для безопасности ввода пароля

2. Определите дисковое пространство.

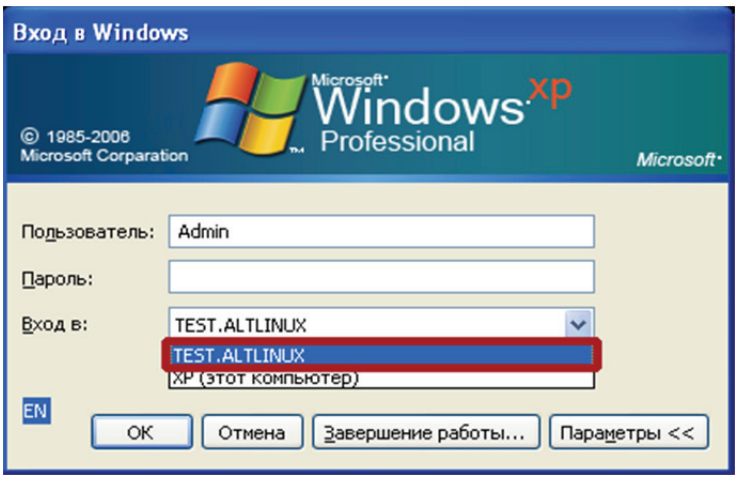


Рис. 2. Окно для входа в дисковое пространство сервера

Указание. Используйте диалоговое окно (рис. 2). Для каждого преподавателя отведено на сервере дисковое пространство, доступ к нему определяется заполнением строки **Пользователь** (регистр заполнения не важен) и строки **Пароль** (для этой строки строгое соблюдения написания символов).

3. В папке **Мои документы** создайте свою папку, соблюдая соглашение 8.3 (имена папок и файлов не кириллица и не более восьми символов в имени). Например, IVS (Иванов Василий Сергеевич).

4. Запустите программу MS Word. Настройте рабочее окно процессора. Создайте информацию, содержащую два абзаца. Один из абзацев состоит из двух строк.

5. Сохраните этот файл в вашей папке.

6. Активизируйте дисковое пространство преподавателя и вашей группы. Скопируйте вашу папку на это дисковое пространство сервера.

7. Завершите сеанс работы в сети.

Указание. Кнопка **Пуск**. Завершение работы сеанса ***student_ZZZ***.

8. Подключите ПК к серверу.

9. Определите дисковое пространство. Отправьте ваш файл в папку **Мои документы** и отредактируйте его с дальнейшим сохранением.

Примечание. Открытие и редактирование файлов на сервере **запрещено**, так как приводит к образованию мусора на сервере (файлы с расширением *.tmp).

ТЕКСТОВЫЕ РЕДАКТОРЫ

Лабораторная работа № 2
РАБОТА В MICROSOFT WORD.
СОЗДАНИЕ И ФОРМАТИРОВАНИЕ ДОКУМЕНТОВ

Цель работы: научиться создавать документ в Word, форматировать абзацы и шрифт.

Методические указания

Создайте в MS Word документ, представленный на рис. 3.

УО «Белорусский государственный технологический университет»
Дисциплина **«КОМПЬЮТЕРНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»**
1 курс, спец.: 1-25 01 07 «Экономика и управление на предприятии»; 1-26 02 02 «Менеджмент»; 1-26 02 03 «Маркетинг»

Экзаменационный билет № 10
Зимняя экзаменационная сессия 2017-2018 учебный год

1. Компьютерная графика. Виды компьютерной графики.
2. MS Excel. Ячейки и их адресация.
3. MathCAD. Индексные переменные. Назначение.
4. Задача.

Заведующий кафедрой Преподаватель

Рис. 3. Образец задания

1. Перейдите на закладку **Разметка страницы** и выберите в группе **Параметры страницы** команду **Поля** (рис. 4). Установите параметры страницы: левое поле – 2,5 см; правое поле – 2,5 см; верхнее поле – 2 см; нижнее поле – 3 см.

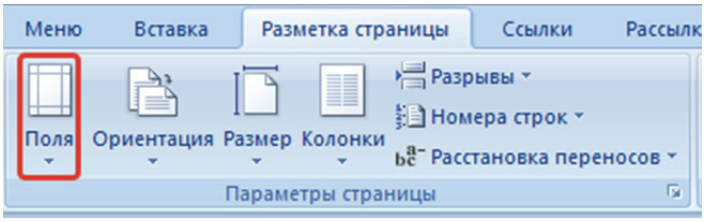


Рис. 4. Установление полей

2. Перейдите на закладку **Главная**, выберите в группе **Абзац** команду **Абзац** (рис. 5). Задайте параметры первого абзаца: Выравнивание – По центру; Отступ: Слева и Справа – 0 см; первая строка – нет; Интервал: Перед и После – 0 пт, междустрочный – Одинарный.

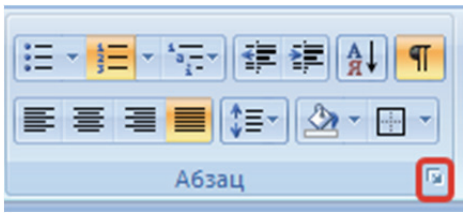


Рис. 5. Параметры абзаца

3. Установите параметры второго абзаца: Выравнивание – По центру; Отступ: Слева и Справа – 0 см; первая строка – нет; Интервал: Перед и После – 12 пт, междустрочный – Точно – 14 пт.

4. Задайте следующие отступы и интервалы в нумерованном списке: Выравнивание – По ширине; Отступ: Слева и Справа – 0 см; первая строка – Отступ – 1,25 см; Интервал: Перед и После – 0 пт, междустрочный – Одинарный.

5. Отформатируйте следующий за списком абзац по параметрам: Выравнивание – По ширине; Отступ: Слева и Справа – 0 см; первая строка – Отступ – 1,25 см; Интервал: Перед – 12 пт, После – 0 пт, междустрочный – Множитель – 2,5.

6. Форматирование последнего абзаца должно соответствовать следующим параметрам: Выравнивание – По центру; Отступ: Слева и Справа – 0 см; первая строка – нет; Интервал: Перед и После – 0 пт, междустрочный – Множитель – 2.

7. Установите курсор перед словом «Дисциплина». Выполните разрыв строки. Для этого нажмите комбинацию клавиш **Shift + Enter**.

8. Поставьте курсор перед фразой «1 курс». Сделайте разрыв строки (см. выше).

9. Установите курсор перед фразой «1-26 02 02». Выполните разрыв строки (см. выше).

10. Разместите курсор после слова «Дисциплина». Выделите текст до конца строки и сделайте все буквы прописными.

Для этого на закладке **Главная** в группе **Шрифт** нажмите левой кнопкой мыши по обозначенной на рис. 6 пиктограмме. Для выделенного текста выберите полужирный шрифт: на закладке **Главная** в группе **Шрифт** выберите команду **Полужирный** (рис. 7) или нажмите комбинацию клавиш **Ctrl + B.**

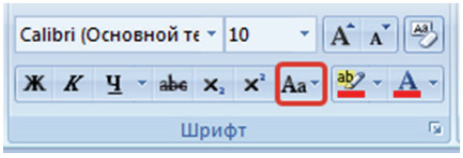


Рис. 6. Изменение регистра шрифта

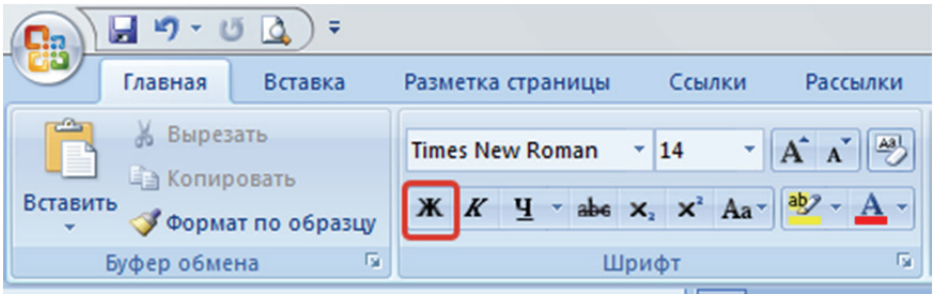


Рис. 7. Изменение толщины шрифта

11. Установите гарнитуру шрифта текста в документе Times New Roman и размер шрифта в двух первых строках первого абзаца 13 пт, в следующих строках – 12 пт, остального текста – 14 пт.

12. Сохраните файл в вашей папке с именем ***word_1.docx***.

Лабораторная работа № 3
РАБОТА В MICROSOFT WORD.
РАБОТА С ТАБЛИЦАМ

Цель работы: получить навыки работы с таблицами.

Методические указания

Создайте таблицу следующего вида (рис. 8).

Таблица 1

Ведомость				
№ п/п	***ФИО***	***Зарплата***	***Премия***	***Итог***
1	Иванов И. И.	300	30	**330**
2	Петров П. П.	400	40	**440**
3	Сидоров С. С.	350	35	**385**
4	Мороз И. М.	500	50	**550**
Всего		**1550**	**155**	**1705**

Рис. 8. Таблица

***Указания*:**

- чтобы вставить таблицу, на вкладке **Вставка** нажмите на кнопку **Таблица** и выберите команду **Вставить таблицу…** (рис. 9). В появившемся окне (рис. 10) задайте количество строк и столбцов в таблице и нажмите на кнопку **ОК**;
- для того чтобы объединить ячейки, выделите их и на вкладке **Макет** выберите в группе **Объединить** команду **Объединить ячейки** (рис. 11);
- текст заголовков столбцов таблицы наберите шрифтом Times New Roman курсивного полужирного начертания размером 16 пт; остальной текст таблицы – 14 пт.
- данные в таблицу о ФИО и размере зарплаты запишите свои. Премия должна составлять 10% от зарплаты. Чтобы вычислить размер премии перейдите в искомую ячейку. Затем на вкладке **Макет** в группе **Данные** выберите команду **Формула** (рис. 12). В появившемся окне (рис. 13) напишите формулу для расчета премии (=C3*0,1 и т. д.);
- значения столбца «Итог» вычисляются как сумма ячеек столбцов «Зарплата» и «Премия» выбранной строки. В окне **Формула** (рис. 13) введите формулу =SUM(A1;B1);

• в строке «Всего» значения вычисляются аналогичным образом;

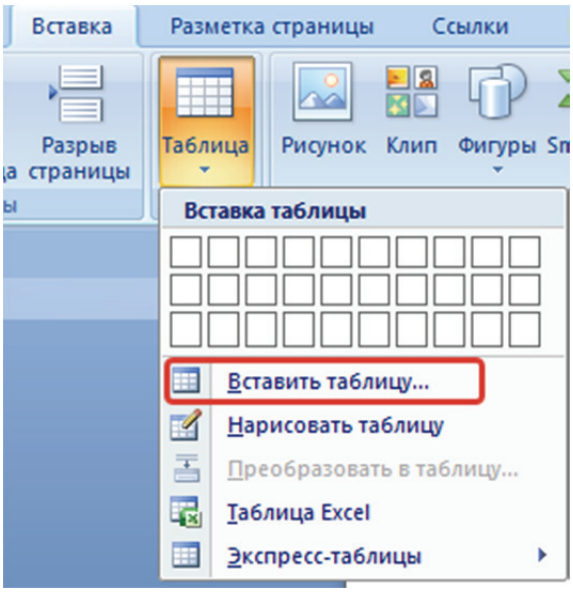


Рис. 9. Вставка таблицы

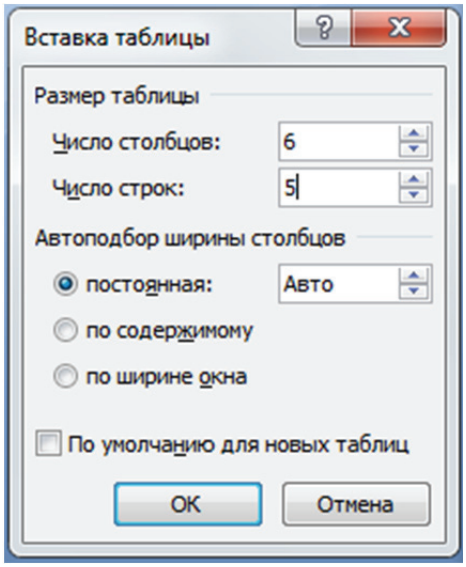


Рис. 10. Параметры таблицы

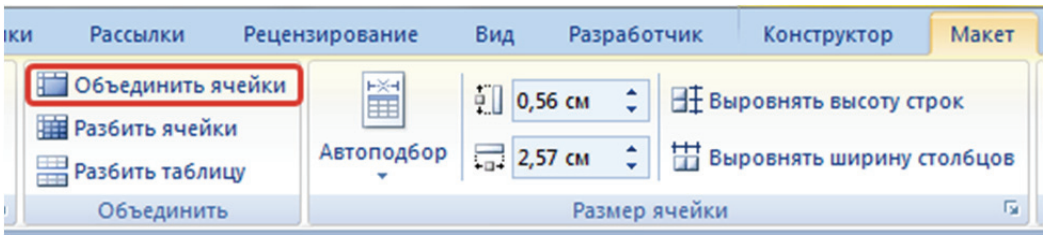


Рис. 11. Команда **Объединить ячейки**

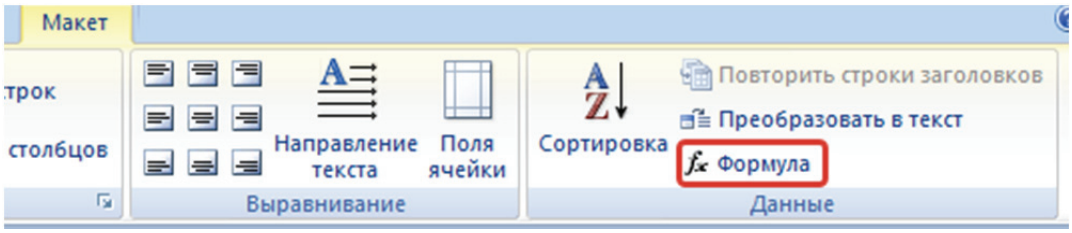


Рис. 12. Команда **Формула**

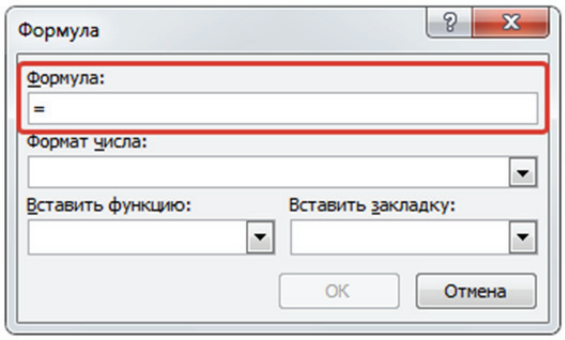


Рис. 13. Окно **Формула**

• для того, чтобы вставить название таблицы на вкладке **Макет** в группе **Таблица** выберите команду **Выделить** (рис. 14) и в открывшемся списке выберите команду **Выделить таблицу** (рис. 15). Затем на клавиатуре нажмите кнопку **Свойства** или правую кнопку мыши и в появившемся меню (рис. 16) выберите команду **Вставить название**. В окне **Название (**рис. 17) заполните необходимые поля и нажмите на кнопку **ОК;**

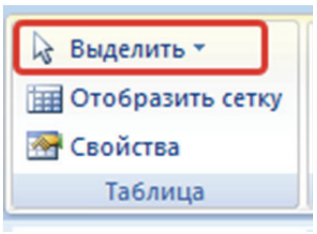


Рис. 14. Команда **Выделить**

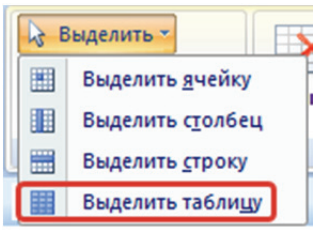


Рис. 15. Команда **Выделить таблицу**

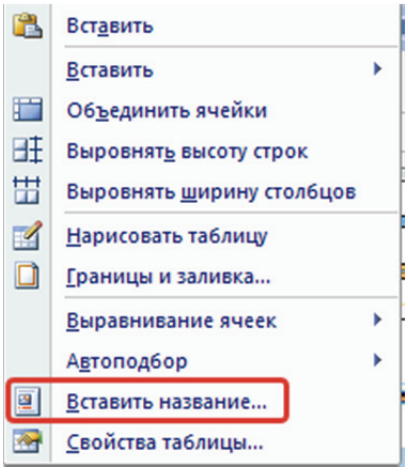


Рис. 16. Команда **Вставить название**

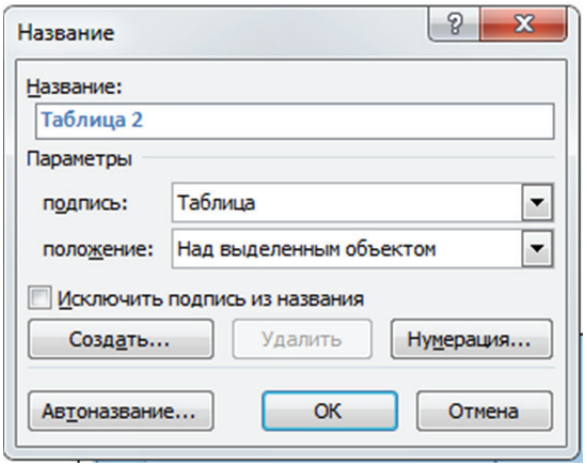


Рис. 17. Окно **Название**

• на вкладке **Конструктор** в группе **Стили таблиц** с помощью команды **Границы** (рис. 18) создайте необходимые границы в таблице (рис. 19);

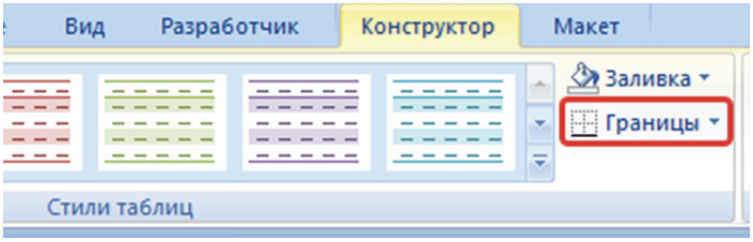


Рис. 18. Команда **Границы**

• создание наружной и внутренней границ таблицы разной ширины начинается с выделения таблицы. Затем выполните команду **Границы** (см. выше). В появившемся окне **Границы и заливка** (рис. 19) на вкладке **Граница** в списке **Тип** выберите **Сетка**. В списке **Тип** выберите рисунок линии границы. В поле **Ширина** установите толщину линии наружной границы и нажмите кнопку **ОК;**

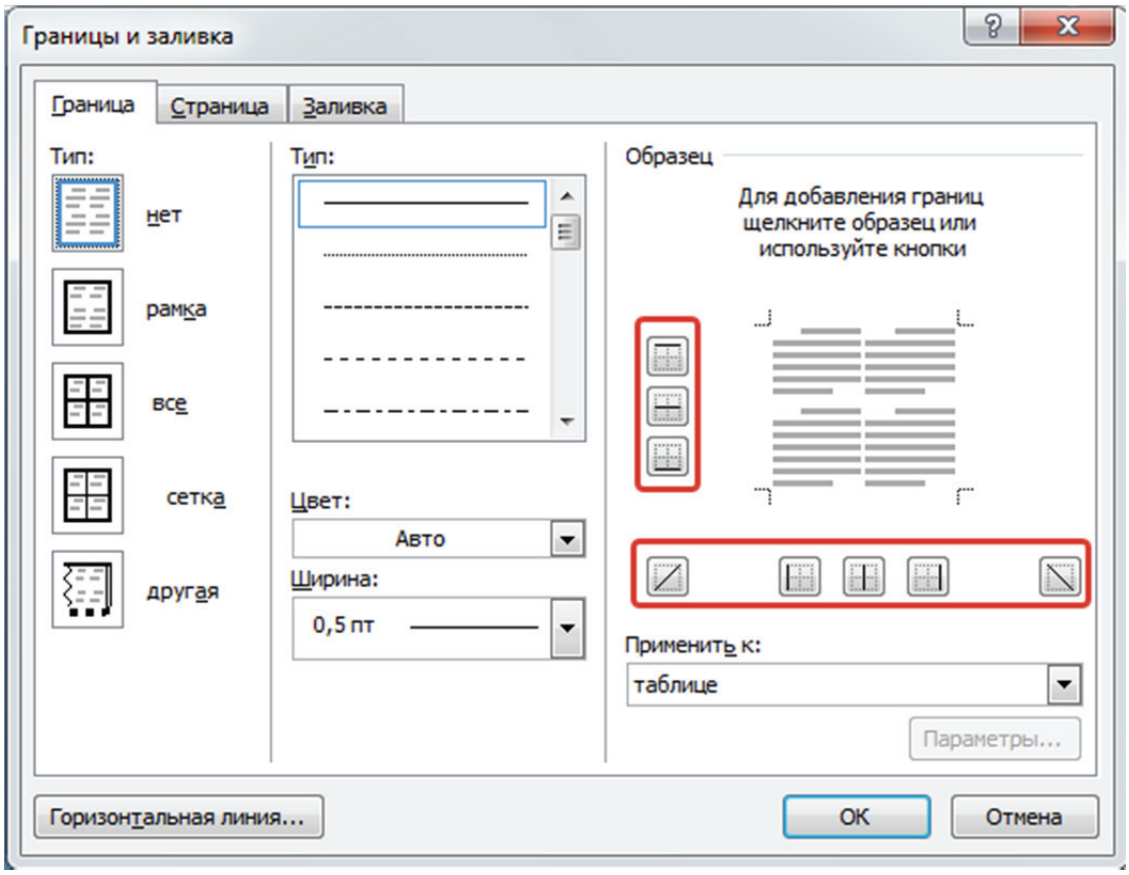


Рис. 19. Окно **Границы и заливка**

• если необходимо создать границу между соседними ячейками, отличающуюся от других границ, выделите эти ячейки, после чего вызовите на экран окно **Границы и заливка** (см. выше). В поле **Тип** выберите тип линии границы и нажмите на кнопки, соответствующие нужной границе, в разделе **Образец**. Сделанные изменения будут сохранены после нажатия кнопки **ОК**.

2. Сохранить файл в вашей папке с именем ***word_2.docx***.

Лабораторная работа № 4
РАБОТА В MICROSOFT WORD. РАБОТА С ОБЪЕКТАМИ

Цель работы: получить навыки работы с объектом Microsoft Equation 3.0 и иллюстрациями «Фигуры», «Клип»; работы с колонтитулами.

Методические указания

1. Создайте таблицу следующего вида (рис. 20).

$$s = \sqrt{x + \sin(x)} + \sum_{i=1}^{6} x_i^2 + \frac{1 + x^2}{1 - x} \qquad (1)$$

Рис. 20. Таблица

***Указания*:**

- для создания формулы используется объект Microsoft Equation 3.0. Он вставляется с помощью команды **Объект** расположенной на вкладке **Вставка** в группе **Текст** (рис. 21);

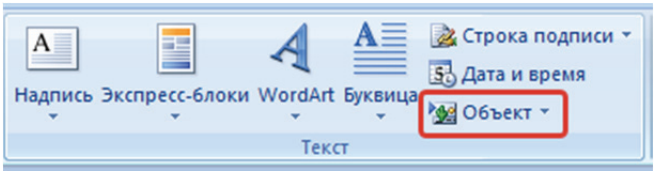


Рис. 21. Команда **Объект**

- в появившемся на экране окне на вкладке **Создание** в списке **Тип объекта** выберите запись «Microsoft Equstion 3.0» (рис. 22) и нажмите кнопку **ОК**. Используя клавиатуру, мышь и панель **Формула** (рис. 23) наберите формулу. Для завершения редактирования формулы нажмите на клавиатуре клавишу **ESC** или кликните мышкой вне границ формулы;
- задание положения содержимого ячеек осуществляется с помощью группы команд, расположенных на вкладке **Макет** в группе **Выравнивание** (рис. 24);
- скройте границы таблицы при печати.

2. Создайте объект WordArt следующего вида (рис. 25).

***Указания*:**

- для того чтобы вставить объект WordArt в документ Word, на вкладке **Вставка** в группе **Текст** нажмите кнопку **Объект WordArt** (рис. 26), а затем выберите требуемый стиль WordArt (рис. 27);

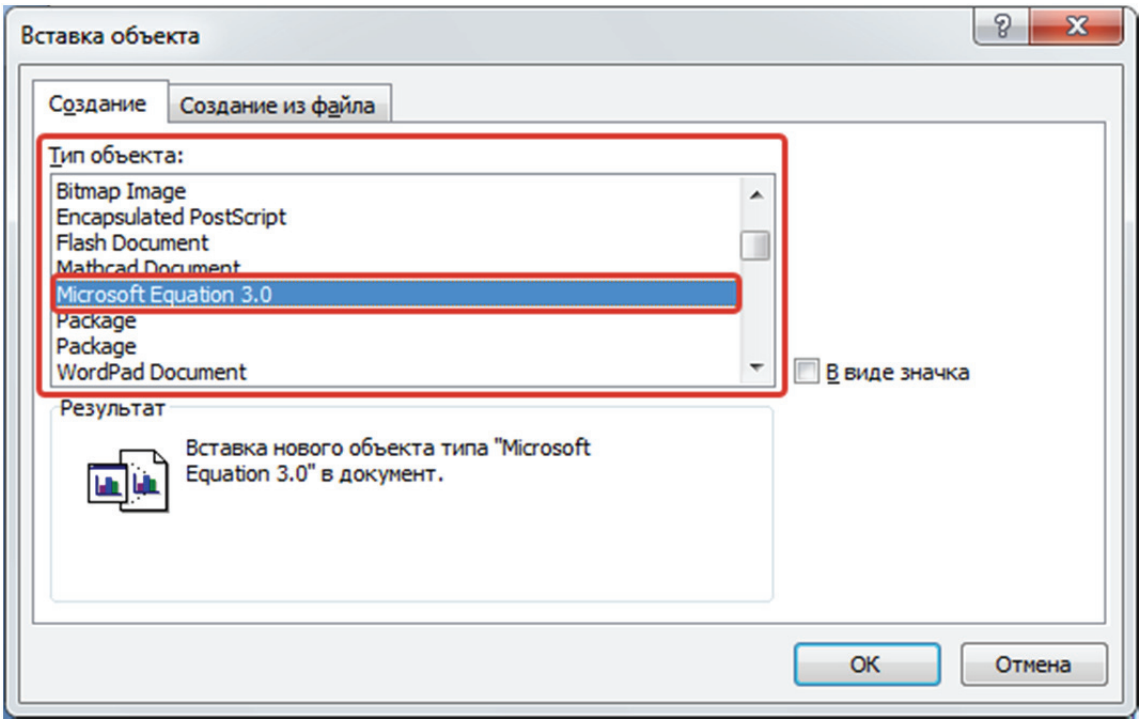


Рис. 22. Окно **Вставка объекта**

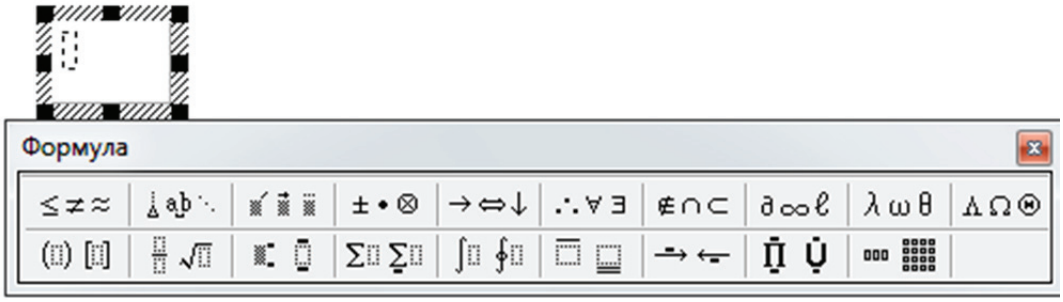


Рис. 23. Панель **Формула**

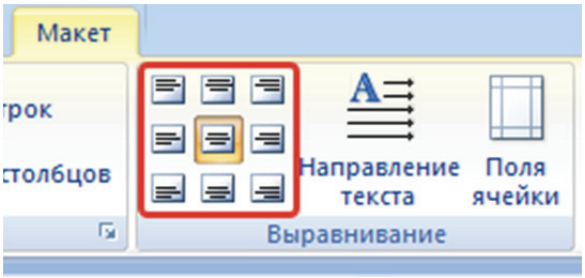


Рис. 24. Группа команд **Выравнивание**

Рис. 25. Объект WordArt

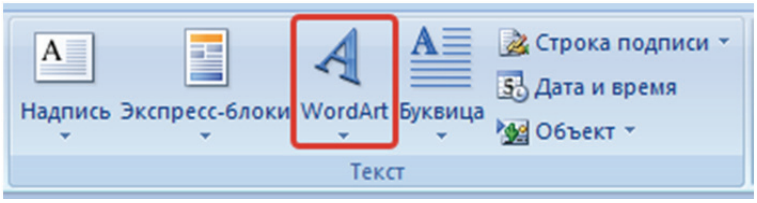


Рис. 26. Кнопка **Объект WordArt**

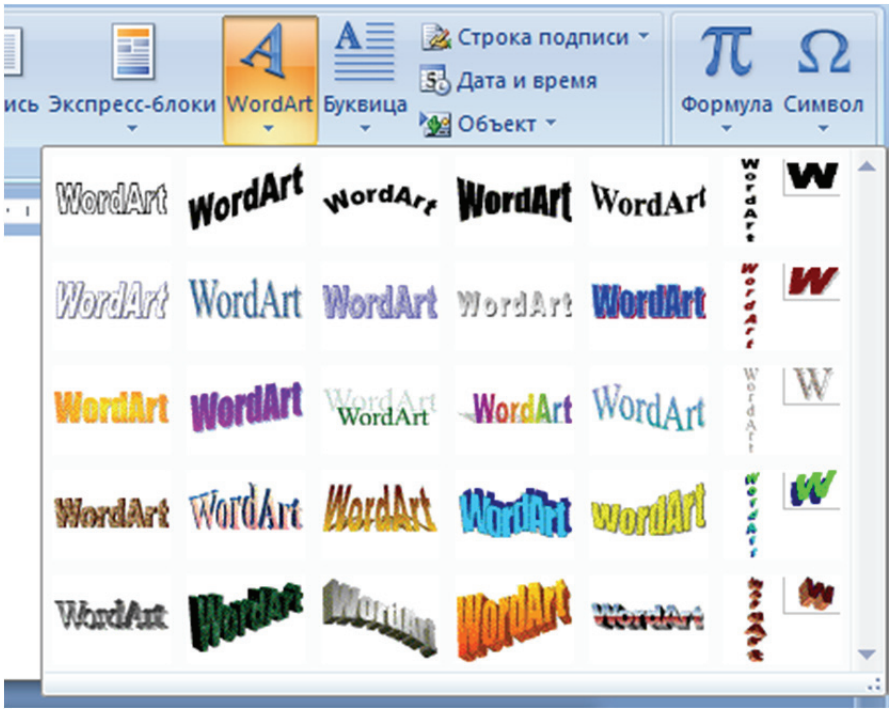


Рис. 27. Список встроенных стилей объекта WordArt

• в появившемся окне **Изменение текста WordArt** (рис. 28) в поле **Текст** введите текст, выберите шрифт, его размер и модификацию начертания;

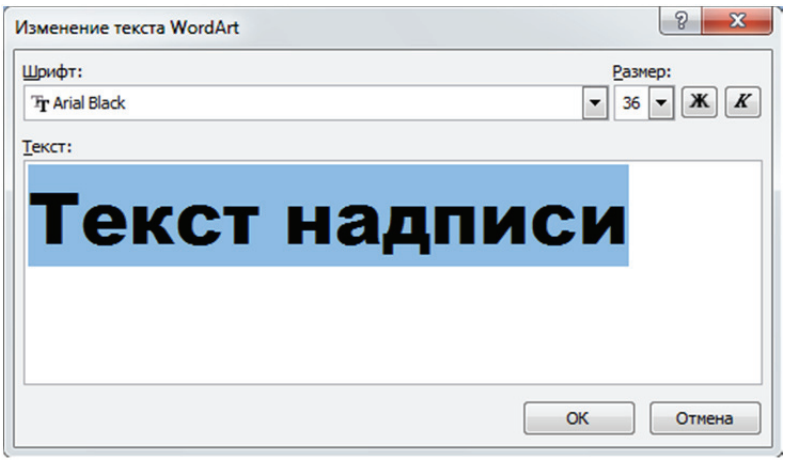


Рис. 28. Окно **Изменение текста WordArt**

• для изменения цвета букв на вкладке **Формат** в группе **Стили WordArt** выберите команду **Заливка фигуры** (рис. 29);

• чтобы создать тень от текста на вкладке **Формат** в группе **Эффекты тени** выберите команду **Эффекты тени** (рис. 30).

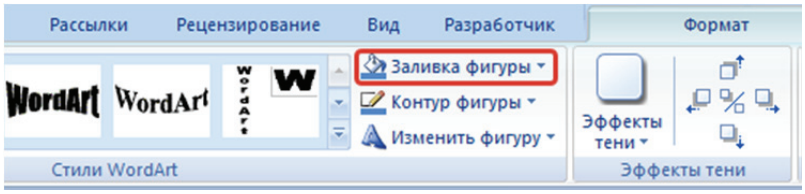


Рис. 29. Команда **Заливка фигуры**

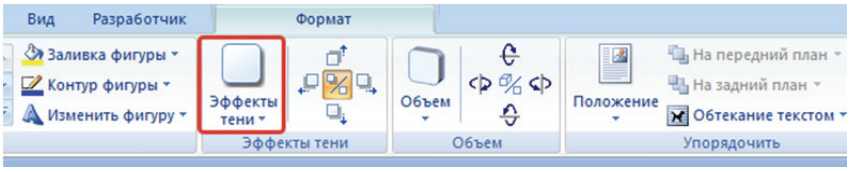


Рис. 30. Команда **Эффекты тени**

3. Создайте таблицу с фигурами следующего вида (рис. 31).

***Указания*:**

• вставьте таблицу с тремя строками и двумя столбцами;

• чтобы вставить круг в документ Word на вкладке **Вставка** в группе **Иллюстрации** нажмите кнопку **Фигуры** (рис. 32), а затем в открывшемся списке в группе **Основные фигуры** выберите команду **Овал** (рис. 33). Если нажать на клавиатуре клавишу **Ctrl** при рисовании фигуры **Овал**, будет нарисован круг;

• заливка фигуры осуществляется с помощью команды **Заливка фигуры**, расположенной на вкладке **Формат** в группе **Стили фигур** (рис. 34). В раскрывшемся списке выберите нужный стиль заливки;

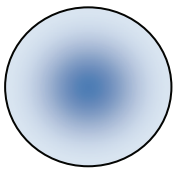


Рис. 1 – Круг с градиентом

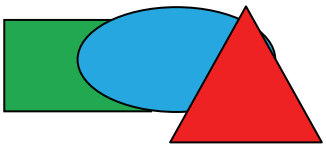


Рис. 2 – Прямоугольник, овал, треугольник

Рис. 3 – Зайчик

Рис. 31. Таблица с графическими объектами

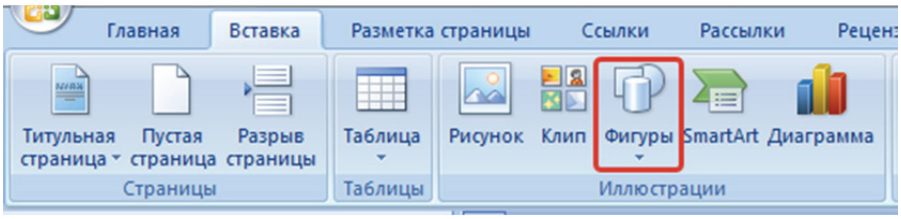


Рис. 32. Команда **Фигуры**

Рис. 33. Список **Основные фигуры**

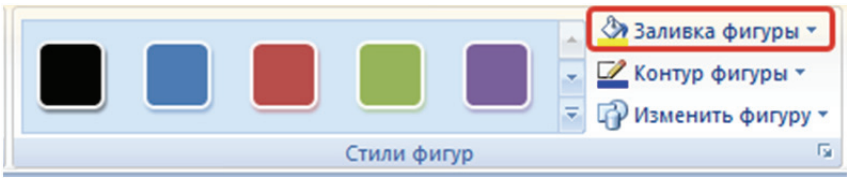


Рис. 34. Команда **Заливка фигуры**

• для того чтобы объединить нескольких фигур в одну (сгруппировать), выделите одну фигуру, кликнув по ней мышкой, затем нажмите на клавиатуре клавишу **Shift** и, не отпуская ее, кликните по другим фигурам, которые необходимо сгруппировать. После того как все фигуры будут выделены, на вкладке **Формат** в группе **Упорядочить** выберите команду **Группировать** (рис. 35). Если необходимо разгруппировать фигуры, выберите в команде **Группировать** подкоманду **Разгруппировать;**

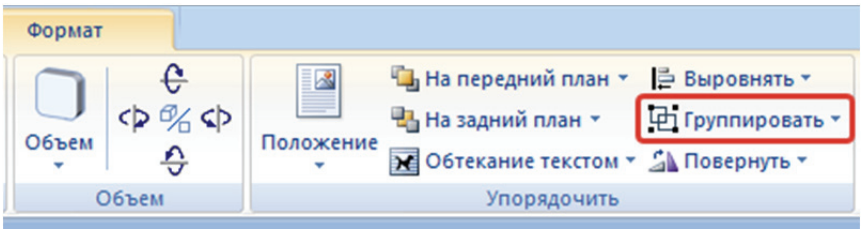


Рис. 35. Команда **Группировать**

• вставка картинки зайчика выполняется с помощью команды **Клип** расположенной на вкладке **Вставка** в группе **Иллюстрации**. В открывшемся окне (рис. 36) в поле **Искать** наберите текст, описывающий изображение, которое хотите найти, и нажмите на кнопку **Начать**. Внизу окна будут выведены картинки отвечающие критериям поиска. Кликнув по картинке, вы вставите ее в документ.

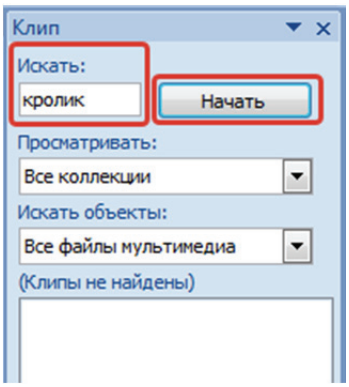


Рис. 36. Окно **Клип**

4. Создайте нижний колонтитул (рис. 37).

Рис. 37. Нижний колонтитул

***Указания*:**

• вставьте пустой нижний колонтитул с помощью команды **Нижний колонтитул,** расположенной в группе **Колонтитулы** вкладки **Вставка** (рис. 38);

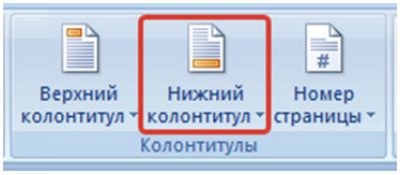


Рис. 38. Команда **Нижний колонтитул**

• запишите свое ФИО. Нажмите на клавиатуре клавишу **Tab;**

• на вкладке **Вставка** в группе **Текст** выберите команду **Экспресс-блоки** (рис. 39), затем в меню команду **Поле** (рис. 40). В окне **Поле** (рис. 41) в списке **Поля** выберите **Page**, в списке **Формат** задайте стиль нумерации страниц и нажмите на кнопку **ОК;**

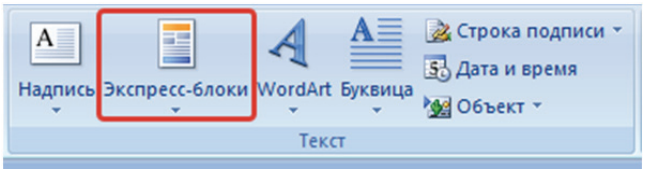


Рис. 39. Команда **Экспресс-блоки**

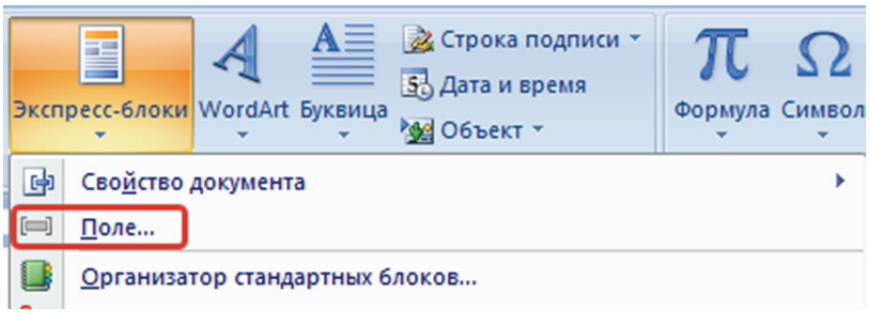


Рис. 40. Команда **Поле**

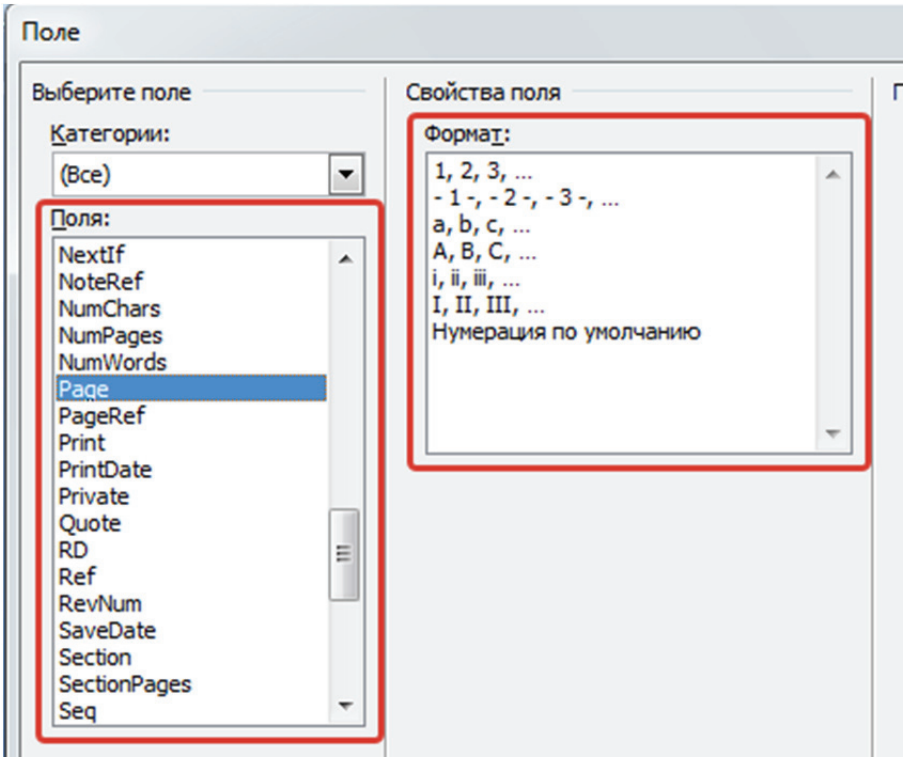


Рис. 41. Окно **Поле**

• для завершения работы с колонтитулом на вкладке **Конструктор** кликните по команде **Закрыть окно колонтитулов** (рис. 42).

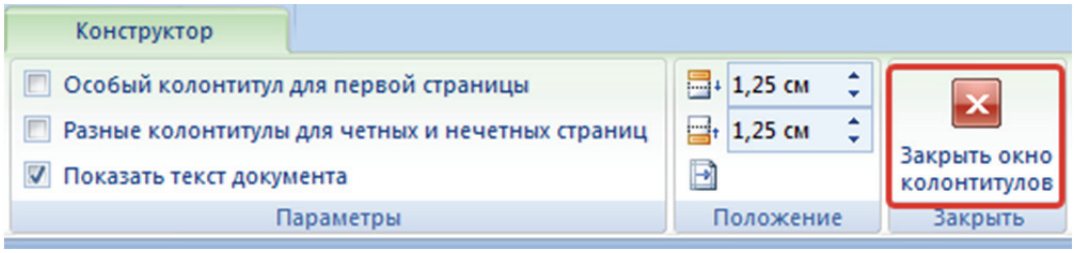


Рис. 42. Команда **Закрыть окно колонтитулов**

5. Сохраните файл в вашей папке с именем ***word_3.docx***.

ЭЛЕКТРОННЫЕ ТАБЛИЦЫ

Лабораторная работа № 5
РАБОТА В EXCEL

Цель работы: научиться создавать и форматировать таблицы в Excel. Изучить принципы адресации ячеек в Excel.

Методические указания

Создайте книгу Excel, которая позволяет автоматизировать принятие решений о поступлении абитуриентов на основании полученных ими оценок. С некоторыми возможностями форматирования ознакомьтесь в приложении В.

1. Запустите программу MS Excel и выберите иконку **Пустая книга**. Создайте таблицу следующего вида (рис. 43).

	A	B	C	D	E	F	G
1	Фамилия	Адрес	Дата рождения	Пол	Стаж	Оценки	
2						Физика	Математика
3							

Рис. 43. Заголовок таблицы

2. Отформатируйте и заполните таблицу собственными данными по примеру на рис. 44.

	A	B	C	D	E	F
1	Фамилия	Адрес	Дата рождения	Пол	Оценки	
2					Физика	Математика
3	Павлов А.П.	Минск	12.01.2000	муж.	8	6
4	Петров И.О.	Брест	23.07.2000	муж.	9	7
5	Иванова А.С.	Витебск	14.03.2000	жен.	4	5
6	Сидоров И.Т.	Гомель	06.07.2000	муж.	8	6
7	Кузнецов Ч.И.	Барановичи	16.01.2000	муж.	6	7
8	Яковлев И.Д.	Логойск	30.05.2000	муж.	5	9
9	Ус Т.И.	Гродно	12.04.2000	жен.	7	8

Рис. 44. Рабочая таблица

Указания:

- для объединения ячеек выделите их и на вкладке **Главная** в группе команд **Выравнивание** нажмите кнопку **Объединить и поместить в центре**. Для переноса текста нажать кнопку **Перенести текст**;

• для автоматического изменения ширины столбца в соответствии с содержимым ячеек выделите таблицу, на вкладке **Главная** в группе команд **Ячейки** нажмите кнопку **Формат** и в разделе **Размер ячейки** выберите пункт **Автоподбор ширины столбца**.

Примечание. Чтобы быстро подобрать ширину всех столбцов листа, нажмите кнопку **Выделить все** (рис. 45) и дважды щелкните любую границу между заголовками двух столбцов;

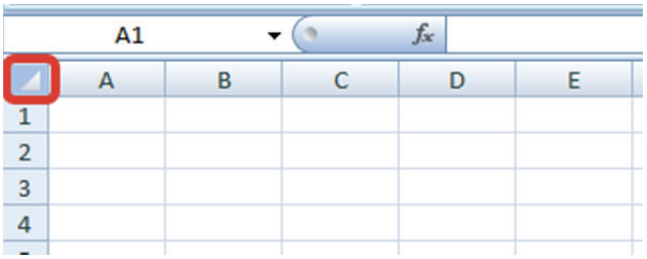


Рис. 45. Кнопка **Выделить все**

• отформатируйте заголовок таблицы следующим образом:
  - ✓ разместите данные по центру столбцов;
  - ✓ установите «полужирный» шрифт;
  - ✓ установите любой фон заливки ячеек;

• задайте обрамление ячеек, щелкнув по кнопке  **Все границы** на вкладке **Главная** в группе команд **Шрифт**;

• для ячеек столбца **Дата рождения** измените формат отображения значений на **Краткий формат даты**. Для этого на вкладке **Главная** в группе команд **Число** измените формат с **Общий** на **Краткий формат даты**.

3. Создайте на втором листе, используя ссылки на ячейки Листа 1 для столбцов A, B и C, таблицу вида, представленного на рис. 46. Информацию в столбец D ввести самостоятельно с клавиатуры.

	A	B	C	D
1	Фамилия	Адрес	Дата рождения	Форма обучения
2	Павлов А.П.	Минск	12.01.2000	бюджет
3	Петров И.О.	Брест	23.07.2000	платная
4	Иванова А.С.	Витебск	14.03.2000	бюджет
5	Сидоров И.Т.	Гомель	06.07.2000	платная
6	Кузнецов Ч.И	Барановичи	16.01.2000	бюджет
7	Яковлев И.Д	Логойск	30.05.2000	платная
8	Ус Т.И.	Гродно	12.04.2000	бюджет

Рис. 46. Таблица на Листе 2

Указания:

• для создания ссылки на ячейку Листа 1 установите курсор в нужную ячейку Листа 2, например А2, в строке формул наберите следующую информацию: =Лист1!А4 и нажмите клавишу **Enter**.

Примечание. Чтобы быстро набрать формулу в строке формул запишите знак «=» и мышкой кликните на нужном листе и затем на нужной ячейке, в этом случае формула появится автоматически;

• скопируйте формулу на весь диапазон ячеек. Для этого необходимо установить курсор в нижний правый угол ячейки А2, когда курсор изменит свой вид на «+», зажав левую кнопку мыши, протащите его вниз, затем вправо;

• заполните столбец **Форма обучения** данными самостоятельно.

4. Измените имена листов 1 и 2 на ***Абитуриент*** и ***Форма обучения***.

Указание. Щелкните правой кнопкой мыши по названию листа и в выпадающем меню выберите кнопку **Переименовать**.

5. Сохраните созданную книгу под именем ***Abiturient*** в своей папке, заранее созданной на диске D, для дальнейшего сохранения на сервер.

6. Создайте новую книгу Excel, в нее поместите представленную на рис. 47 таблицу, используя ссылки на соответствующие ячейки книги ***Abiturient***.

	A	B	C	D
1	Фамилия	Адрес	Дата рождения	Форма обучения
2				
3				
4	Павлов А.П.	Минск	12.01.2000	бюджет
5	Петров И.О.	Брест	23.07.2000	платная
6	Иванова А.С.	Витебск	14.03.2000	бюджет
7	Сидоров И.Т.	Гомель	06.07.2000	платная
8	Кузнецов Ч.И.	Барановичи	16.01.2000	бюджет
9	Яковлев И.Д.	Логойск	30.05.2000	платная
10	Ус Т.И.	Гродно	12.04.2000	бюджет

Рис. 47. Лист «Список_Группы»

Указания:

• введите заголовки столбцов таблицы с клавиатуры;

• для создания ссылки на ячейки листа ***Абитуриент*** книги ***Abiturient*** установите курсор в нужную ячейку листа, например А4, в строке формул наберите формулу =[Abiturient.xlsx]Лист1!А4 и нажмите клавишу **Enter**. Для переноса других данных достаточно скопировать формулу на весь диапазон ячеек. Аналогично можно перенести данные с листа ***Форма обучения*** книги ***Abiturient***.

Примечание. Чтобы быстро набрать формулу, в строке формул запишите знак «=» и мышкой кликните на нужной книге (эта книга должна быть открыта), на нужном листе и затем на нужной ячейке, в этом случае формула появится автоматически.

7. В столбце данных D, для студентов платной формы обучения, вставьте примечание – «Оплачено» или «Не оплачено».

Указание. Для создания примечания щелкните правой кнопкой мыши по нужной ячейке и из выпадающего меню выберите команду **Вставить примечание**.

8. Измените имя листа на ***Список группы***.

9. Сохраните книгу под именем ***Spisok.xlsx*** в своей папке для дальнейшего сохранения на сервер.

Лабораторная работа № 6
РАБОТА В EXCEL. ВСТРОЕННЫЕ ФУНКЦИИ. ПОСТРОЕНИЕ ГРАФИКОВ

Цель работы: научиться пользоваться встроенными арифметическими и логическими функциями Excel, создавать и форматировать диаграммы и гистограммы.

Методические указания

Создайте книгу Excel, которая позволяет автоматизировать принятие решений о поступлении абитуриентов на основании полученных ими оценок.

1. Откройте ранее созданную книгу ***Abiturient.xlsx*** и добавьте в таблицу столбцы с заголовками «Сумма баллов» и «Анализ». Отформатируйте эти заголовки так же, как и остальные, используя кнопку **Формат по образцу**.

Указание. Для копирования формата необходимо щелкнуть по ячейке, из которой нужно скопировать форматирование, и нажать на вкладке **Главная** из группы команд **Буфер обмена** кнопку **Формат по образцу**, затем щелкнуть на ячейке, к которой это форматирование необходимо применить.

2. В ячейках столбца «Сумма баллов» введите формулу, обеспечивающую подсчет суммарного балла каждого студента.

Указания:

- выделите первую ячейку, в которой будет находиться формула. На вкладке **Формулы** в группе команд **Библиотека функций** разверните выпадающее меню кнопки **Автосумма** и выберите из списка кнопку **Сумма**. Выделите мышью ячейки, в которых находятся нужные цифры, и нажмите клавишу **Enter**.

Примечание. Если необходимо выделить смежные ячейки, то для этого достаточно провести по ним курсором при нажатой левой клавише мыши. В этом случае в формуле появится ссылка на диапазон ячеек, например E3:F3. Если необходимо выделить несмежные ячейки, то их нужно выделять левой клавишей мыши, удерживая клавишу **Ctrl**. В этом случае в формуле появятся ссылки на отдельные ячейки, разделенные точкой с запятой, например E3;F3;

- скопируйте полученную формулу во все ячейки столбца «Сумма баллов».

3. В ячейках столбца «Анализ» введите формулу, обеспечивающую анализ информации, следующим образом:

а) если абитуриент получил оценку меньше 4 по любому предмету, то в ячейке должно появиться слово «Завалил»;

б) если сумма баллов студента меньше 16, то в ячейке должно появиться слово «Не прошел»;

в) если сумма баллов студента больше либо равна 16, то в ячейке должно появиться слово «Поступил».

Указание. Для анализа информации необходимо использовать встроенные логические функции **ЕСЛИ** и **ИЛИ**. Выделите первую ячейку, в которой будет находиться формула. Введите в ячейку следующую формулу и нажмите ENTER:

=ЕСЛИ(ИЛИ(E3<4;F3<4);"Завалил";ЕСЛИ(G3>=16;"Поступил";
"Не прошел"))

4. Отсортируйте таблицу по фамилиям в алфавитном порядке.

Указание. Выделите столбец **Фамилия**, затем на вкладке **Главная** в группе команд **Редактирование** разверните меню кнопки **Сортировка и фильтр** и нажмите кнопку **Сортировка от А до Я**. В появившемся окне предупреждения примите команду **Автоматически расширить выделенный диапазон**.

5. Отсортируйте данные столбца «Анализ» так, чтобы сначала располагались поступившие студенты, затем не прошедшие по конкурсу, а затем не сдавшие экзамен.

Указание. Выделите столбец «Анализ», затем на вкладке **Главная** в группе команд **Редактирование** разверните меню кнопки **Сортировка и фильтр** и нажмите кнопку **Настраиваемая сортировка**. В появившемся окне предупреждения примите команду **Автоматически расширить выделенный диапазон**. В открывшемся диалоговом окне выбрать следующие параметры: для поля **Столбец** выбрать – **Анализ**, для поля **Сортировка** выбрать – **Значения**, для поля **Порядок** выбрать – **Настраиваемый список** и в поле **Элементы списка** через запятую записать: «Поступил», «Не прошел», «Завалил», затем нажать **ОК**.

6. Получите на текущем листе итоговые значения:

- всего поступивших;
- сколько поступило;
- сколько не прошло по конкурсу;
- сколько не сдало экзамен.

Указания:

- введите в ячейки L5:L8 следующую текстовую информацию: L5 – «Всего поступавших», L6 – «Поступило», L7 – «Не поступило», L8 – «Завалило»;

• для подсчета значений необходимо использовать функции **СЧЕТ** и **СЧЕТЕСЛИ**;

• в ячейку M5 введите формулу: **=СЧЕТ(G3:G9)**;

• в ячейку M6 введите формулу: **=СЧЕТЕСЛИ(H3:H9; "Поступил")**;

• в ячейку M7 введите формулу: **=СЧЕТЕСЛИ(H3:H9; "Не прошел")**;

• в ячейку M8 введите формулу: **=СЧЕТЕСЛИ(H3:H9; "Завалил")**.

Примечание. Если формулы введены верно, то получится результат, представленный на рис. 48.

Всего поступавших	7
Поступило	2
Не поступило	3
Завалило	2

Рис. 48. Результаты расчета

7. По полученным в пятом задании данным постройте круговую диаграмму на Листе 3.

Указания:

• выделите в полученной в пятом задании таблице данные о результатах поступления (включая ячейки с текстом), затем на вкладке **Вставка** в группе команд **Диаграммы** разверните меню кнопки **Вставить круговую диаграмму** и выберите тип диаграммы **Объемная круговая**;

• переместите диаграмму на Лист 3. Для этого выделите диаграмму, в появившейся вкладке **Работа с диаграммами** на вкладке **Конструктор** нажмите кнопку **Переместить диаграмму** и следуйте указаниям диалогового окна (эту же команду можно выбрать из контекстного меню);

• отформатируйте диаграмму: измените название диаграммы на ***Итоги поступления***, добавьте подписи данных на диаграмму. Результат представлен на рис. 49.

Рис. 49. Диаграмма «Итоги поступления»

8. Используя данные столбца «Фамилия» и «Сумма баллов», постройте на Листе 3 объемную гистограмму результатов поступления (рис. 50).

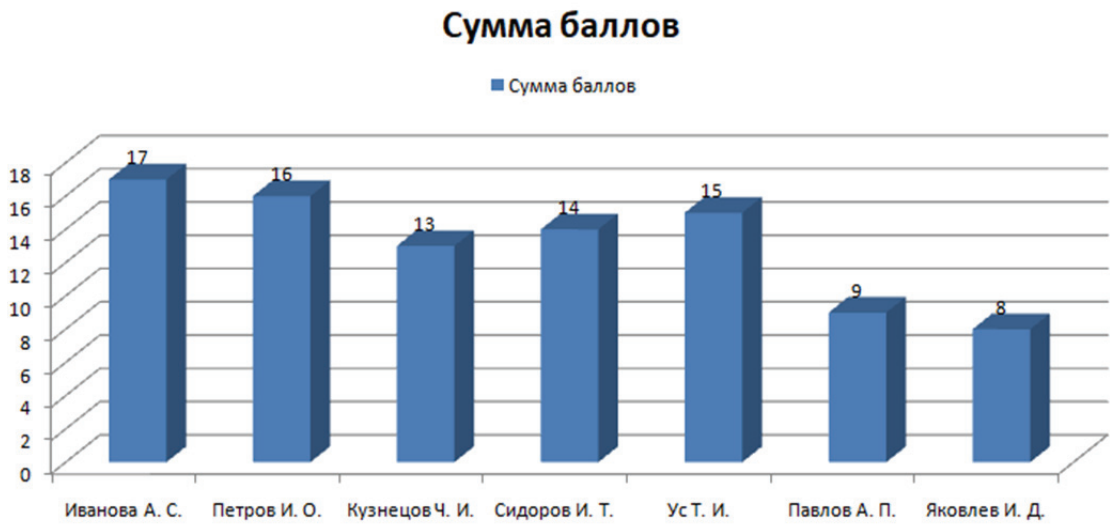


Рис. 50. Диаграмма «Результаты поступления»

Указания:

• выделите столбец «Фамилия», затем удерживая нажатой клавишу **Ctrl** выделите столбец «Сумма баллов». На вкладке **Вставка** в группе команд **Диаграммы** разверните меню кнопки **Вставить гистограмму** и выберите тип гистограммы **Объемная гистограмма с группировкой**;

• отформатируйте диаграмму: вставьте легенду, добавьте под данных на диаграмму. Результат представлен на рис. 50.

Лабораторная работа № 7
РАБОТА В EXCEL. СВЯЗЫВАНИЕ ДОКУМЕНТОВ

Цель работы: научиться совместному использованию программ MS Word и MS Excel.

Методические указания

1. Создайте макет отчета компании в программе MS Word (рис. 47). Для разметки страницы используйте таблицу с невидимыми границами. Вставьте логотип с названием компании. Используйте стиль **WordArt** для названия компании. Сохраните созданный документ под именем ***Отчет компании***.

Рис. 51. Макет отчета компании

Указания:

- вставьте таблицу, которая содержит четыре строки и два столбца. Объедините ячейки там, где это необходимо. Сделайте нужные записи в ячейках таблицы. Используя клавишу **Ctrl**, выделите ячейки с текстом, затем в появившейся вкладке **Работа с таблицами** на вкладке **Макет** в группе команд **Выравнивание** нажмите кнопку **Выровнять по центру**;
- напишите название компании, используя надпись WordArt. Для этого на вкладке **Вставка** в группе команд **Текст** нажмите кнопку **WordArt**. Щелкните по надписи и на появившейся вкладке **Средства рисования** выберите вкладку **Формат**. Используя появившуюся панель инструментов, отформатируйте надпись;
- найдите рисунок логотипа компании и вставьте его в нужную ячейку таблицы, используя кнопку **Рисунки** из группы команд **Иллюстрации** на вкладке **Вставка**. Щелкните по рисунку и в появившемся меню **Работа с рисунками** на вкладке **Формат** в группе команд **Упорядочение** разверните меню кнопки **Положение** и выберите положе-

ние рисунка **В тексте**. При необходимости обрежьте белые поля рисунка, используя кнопку **Обрезка** из группы команд **Размер**. Округлите значение размеров логотипа до целых чисел, при этом разверните панель группы команд **Размер** и убедитесь, что установлена галочка на пункте меню **Сохранять пропорции**.

2. Создайте макет таблицы с отчетными данными компании в программе MS Excel (рис. 52).

	Месяц	1 квартал	2 квартал	3 квартал	4 квартал	Всего
2						
3	Число продаж	4258	5782	3256	7852	
4	Выручка от реализации					
5	Затраты на сбыт					
6	Валовая прибыль					
7						
8	Зарплата персоналу	650	700	500	800	
9	Затраты на рекламу	1000	1000	1000	1000	
10	Косвенные затраты					
11	Суммарные затраты					
12						
13	Произ. Прибыль					
14	Норма прибыли					
15						
16	Цена изделия	40				
17	Затраты на изделие	25				

Рис. 52. Таблица с отчетными данными в программе MS Excel

Указание. Сформируйте макет требуемого вида, используя кнопку **Границы** на вкладке **Главная** в группе команд **Шрифт**. Введите в строки «Число продаж», «Зарплата персоналу», «Затраты на рекламу», «Цена изделия», «Затраты на изделие» числовые данные произвольным образом.

3. Вычислите оставшиеся позиции в таблице следуя указаниям. Результат представлен на рис. 53.

	A	B	C	D	E	F
1	Месяц	1 квартал	2 квартал	3 квартал	4 квартал	Всего
2						
3	Число продаж	4258	5782	3256	7852	21148
4	Выручка от реализации	170320	231280	130240	314080	845920
5	Затраты на сбыт	106450	144550	81400	196300	528700
6	Валовая прибыль	63870	86730	48840	117780	317220
7						
8	Зарплата персоналу	650	700	500	800	2650
9	Затраты на рекламу	1000	1000	1000	1000	4000
10	Косвенные затраты	25548	34692	19536	47112	126888
11	Суммарные затраты	27198	36392	21036	48912	133538
12						
13	Произ. Прибыль	36672	50338	27804	68868	183682
14	Норма прибыли	0,22	0,22	0,21	0,22	0,22
15						
16	Цена изделия	40				
17	Затраты на изделие	25				

Рис. 49. Таблица с отчетными данными

Указания:

• **выручка от реализации** находится как произведение числа продаж за квартал на цену изделия. Формула для первого квартала следующая: =B3*B16. Затем эту формулу скопируйте для других кварталов.

Примечание. Ссылка на ячейку B3 *относительная*, т. е. при копировании формулы вправо ссылка будет изменяться на значения C3, D3, E3. Ссылку на ячейку с ценой товара сделайте *абсолютной,* такая ссылка не изменяется при копировании формулы. Чтобы сделать ссылку абсолютной поставьте знаки $ перед буквой и перед цифрой ссылки или нажмите клавишу **F4** на клавиатуре;

• «Затраты на сбыт» находятся путем произведения числа продаж и затрат на изделие. Формула для первого квартала будет иметь вид =B3*B17. Ссылка на ячейку B3 *относительная*, ссылка на ячейку с затратами на производство одного изделия – *абсолютная*. Затем эту формулу скопируйте для других кварталов;

• «Валовая прибыль» находится как разность выручки от реализации и затрат на сбыт. Для первого квартала формула имеет вид =B4 – B5. Затем эту формулу скопируйте для других кварталов;

• «Косвенные затраты» в фонд предприятия примите равными 15% от выручки от реализации. Формула для первого квартала имеет вид =B4*15%. Затем эту формулу скопируйте для других кварталов;

• «Суммарные затраты» являются суммой затрат на персонал, затрат на рекламу и косвенных затрат. Формула для первого квартала имеет вид =СУММ(B8:B10). Затем эту формулу скопируйте для других кварталов;

• «Производственная прибыль» равна валовой прибыли за вычетом суммарных затрат. Формула для первого квартала имеет вид = B6 – B11). Затем эту формулу скопируйте для других кварталов;

• «Норма прибыли» равняется отношению прибыли к выручке от реализации. Формула для первого квартала имеет вид = B13/B4. Затем эту формулу скопируйте для других кварталов. Для значений нормы прибыли установите точность равной двум знакам после запятой, используя кнопки **Уменьшить разрядность** и **Увеличить разрядность** в группе команд **Число** на вкладке **Главная**;

• для столбца таблицы «Всего» вычислите сумму значений для всех кварталов. Для строки «Число продаж» формула имеет вид =СУММ(B3:E3). Затем эту формулу скопируйте для других позиций таблицы;

• для позиции «Норма прибыли» в столбце «Всего» вычислите среднее значение по кварталам. Формула для расчета имеет вид = СРЗНАЧ(B14:E14).

Сохраните книгу Excel под именем ***Расчет*** в свою папку на рабочем компьютере.

4. Создайте гистограмму на отдельном листе, используя данные из строк «Число продаж», «Выручка от реализации», «Затраты на сбыт» и «Валовая прибыль». Результат представлен на рис. 54. Сохраните лист в новую книгу (назовите ее ***График***).

Указания:

- выделите ячейки A3:E6, затем на вкладке **Вставка** в группе команд **Диаграммы** разверните меню кнопки **Вставить гистограмму** и выберите тип гистограммы **Объемная гистограмма с группировкой**. В открывшемся вкладке **Работа с диаграммами** на вкладке **Конструктор** в группе команд **Данные** нажмите кнопку **Выбрать данные**. В открывшемся диалоговом окне на правой панели **Подписи горизонтальной оси** нажмите кнопку **Изменить**. Выделите ячейки B1:E1, которые содержат названия кварталов, и нажмите кнопку **ОК**. Отформатируйте гистограмму: измените название гистограммы на ***Годовой отчет***, разместите легенду справа;
- переместите гистограмму на Лист 2. Затем щелкните правой кнопкой мыши по имени листа и в появившемся контекстном меню выберите команду **Переместить или скопировать**. В выпадающем меню выбрать кнопку **Новая книга** и нажать кнопку **ОК**. Сохраните новую книгу под именем ***График***.

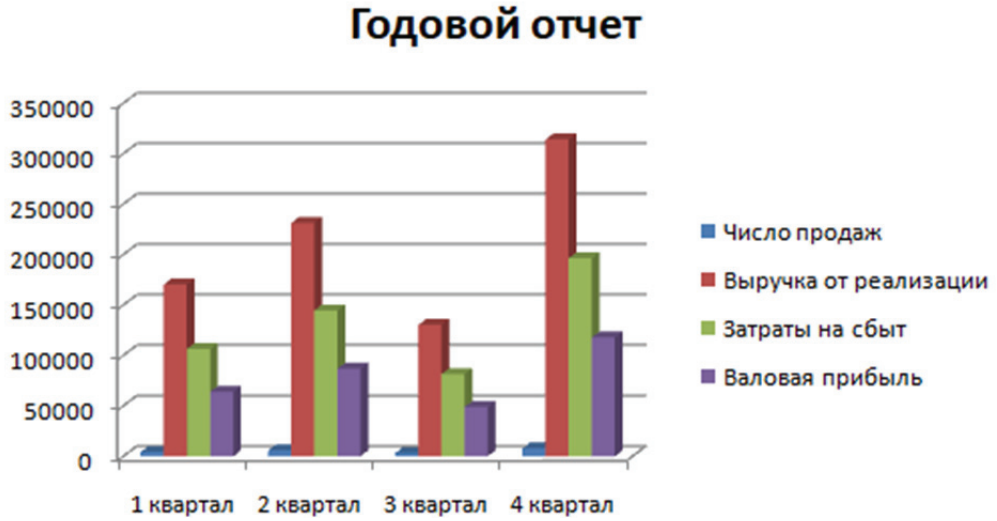


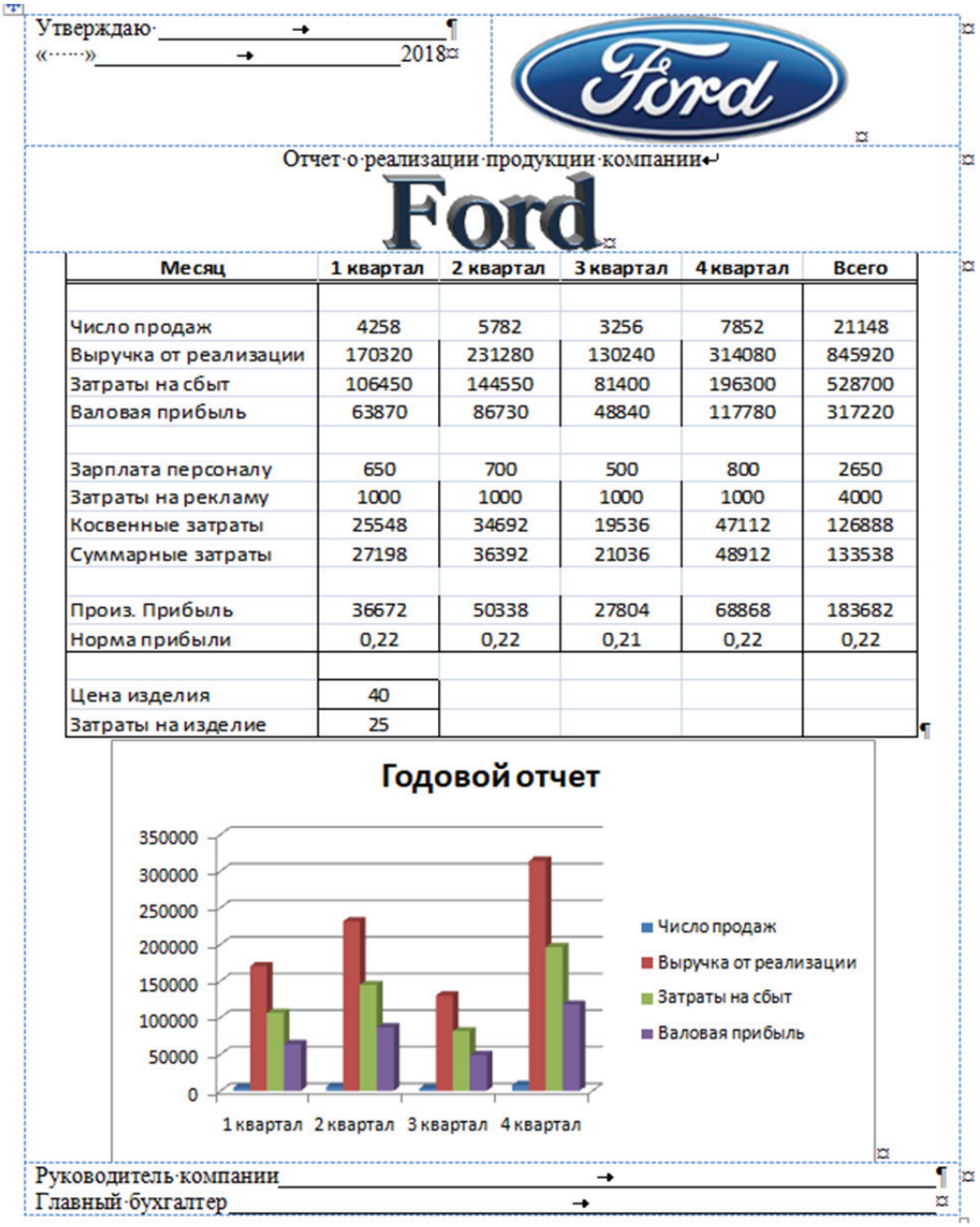
Рис. 54. Гистограмма для отчета

5. Вставьте в текстовый документ MS Word ***Отчет компании*** объекты из книг MS Excel ***Расчет*** и ***График***. Сохраните документ под именем ***Отчет***. Отчет в готовом виде представлен на рис. 55.

Указания:

• откройте текстовый документ ***Отчет компании***. Поставьте курсор в документе после названия компании и на вкладке главного меню **Вставка** в группе команд **Текст** нажмите кнопку **Объект**. Выберите вкладку **Создание из файла** и нажмите кнопку **Обзор**. Выберите файл MS Excel ***Расчет***, установите флажок на команде **Связь с файлом** и нажмите кнопку **ОК;**

• поставьте курсор после таблицы и проделайте аналогичные действия, для того чтобы вставить график из книги ***График***.



Месяц	1 квартал	2 квартал	3 квартал	4 квартал	Всего
Число продаж	4258	5782	3256	7852	21148
Выручка от реализации	170320	231280	130240	314080	845920
Затраты на сбыт	106450	144550	81400	196300	528700
Валовая прибыль	63870	86730	48840	117780	317220
Зарплата персоналу	650	700	500	800	2650
Затраты на рекламу	1000	1000	1000	1000	4000
Косвенные затраты	25548	34692	19536	47112	126888
Суммарные затраты	27198	36392	21036	48912	133538
Произ. Прибыль	36672	50338	27804	68868	183682
Норма прибыли	0,22	0,22	0,21	0,22	0,22
Цена изделия	40				
Затраты на изделие	25				

Рис. 55. Отчет в окончательном виде

Лабораторная работа № 8
РАБОТА В EXCEL.
ВНЕДРЕНИЕ ДОКУМЕНТОВ

Цель работы: научиться совместному использованию программ MS Word и MS Excel.

Методические указания

1. Создайте макет счета на приобретение товаров в программе MS Word (рис. 56). Для получения требуемого вида счета используйте фигуру **Прямоугольник**. Сохраните созданный документ под именем ***Счет***.

Указания:

- создайте новый документ MS Word. На вкладке главного меню **Вставка** в группе команд **Иллюстрации** разверните меню кнопки **Фигуры**. Выберите фигуру **Прямоугольник** и обрисуйте контуры будущего счета;
- щелкните по фигуре и в появившейся вкладке **Средства рисования** выберите вкладку **Формат**, используя ленту инструментов, отформатируйте фигуру следующим образом: уберите заливку фигуры, добавьте тень;
- щелкните по контуру фигуры правой кнопкой мыши и из выпадающего меню выберите команду **Добавить текст**. На ленте инструментов вкладки **Средства рисования** в группе команд **Текст** разверните меню кнопки **Выровнять текст** и выберите команду **Сверху**. Наберите номер, дату счета и данные для подписи внизу счета. Реквизиты поставщика и получателя оформите следующим способом: щелкните по фигуре и в появившейся вкладке **Средства рисования** выберите вкладку **Формат**, в открывшейся ленте инструментов в группе команд **Вставка фигур** нажмите кнопку **Добавление надписи**, введите данные поставщика, аналогично добавьте реквизиты получателя.

2. Внедрите в макет счета лист MS Excel, в котором введите исходные данные и произведите необходимые расчеты, следуя указаниям. Пример листа показан на рис. 57.

Указания:

- поставьте курсор в макет счета после реквизитов предприятий. На вкладке главного меню **Вставка** в группе команд **Текст** нажмите кнопку **Объект**. Выберите вкладку **Создание** и выберите тип объекта **Лист Microsoft Excel** и нажмите кнопку **ОК**. В появившуюся таблицу MS Excel введите исходные данные, которые необходимо придумать самостоятельно. Пример приведен на рис. 57;

СЧЕТ №213 от 21.09.2018

ПОСТАВЩИК: ООО «Гепард»	**ПОЛУЧАТЕЛЬ:** ЗАО «Приятель»
Банк: «Приорбанк»	Банк: «Белбизнесбанк»
Счет №211357645	Счет №567547328
Адрес: г. Минск, пр. Скорины, 21	Адрес: г. Минск, пр. Свердлова, 15
Тел./факс 245-64-56	Тел./факс 238-72-18

Счет действителен в течение 3 дней

Руководитель _____ **(Иванов А. П.)**

Главный бухгалтер _____ **(Петрова Т. В.)**

Рис. 56. Макет счета

	A	B	C	D	E	F
1	**Наименование**	**Количество (кг)**	**Учетная цена (руб.)**	**Наценка**	**Отпускная цена**	**Скидка**
2	Бананы	10	1,6	20%	1,92	3%
3	Апельсины	8	2,5	15%	2,875	1%
4	Ананасы	6	3,8	8%	4,104	
5						
6	**Итого:**	**66,82**				
7	**Скидка:**	**0,81**				
8	**Итого к оплате:**	**66,02**				

Рис. 57. Пример исходных данных

• для расчета **отпускной цены** необходимо учетную цену увеличить на величину наценки. Формула для первой позиции товара имеет вид =C2+C2*D2. Далее формулу копируйте для остальных позиций товара;

• в строке «Итого» находится общая стоимость всех позиций товара. При этом необходимо учесть, что в таблице, в столбце «Отпускная

цена» размещается цена за единицу товара, поэтому при вычислении общей стоимости товаров необходимо умножать количество товара на его отпускную цену. При расчете удобно воспользоваться встроенной функцией **СУММПРОИЗВ**, формула для расчета имеет следующий вид: =СУММПРОИЗВ(B2:B4;E2:E4). Сделайте разрядность полученного результата два знака после запятой;

- в строке «Скидка» находится общая сумма скидки для всех позиций. При расчете удобно воспользоваться встроенной функцией СУММПРОИЗВ, формула для вычислений будет выглядеть =СУММПРОИЗВ(B2:B4;E2:E4;F2:F4);
- в строке «Итого к оплате» находится общая стоимость всех позиций с учетом скидок, которая может быть найдена как разность результатов ячеек «Итого» и «Скидка». Результат выполненной работы представлен на рис. 58.

СЧЕТ №213 от 21.09.2018

ПОСТАВЩИК: ООО «Гепард»
Банк: «Приорбанк»
Счет №211357645
Адрес: г. Минск, пр. Скорины, 21
Тел./факс 245-64-56

ПОЛУЧАТЕЛЬ: ЗАО «Приятель»
Банк: «Белбизнесбанк»
Счет №567547328
Адрес: г. Минск, пр. Свердлова, 15
Тел./факс 238-72-18

Наименование	Количество (кг)	Учетная цена (руб.)	Наценка	Отпускная цена	Скидка
Бананы	10	1,6	20%	1,92	3%
Апельсины	8	2,5	15%	2,875	1%
Ананасы	6	3,8	8%	4,104	
Итого:	**66,82**				
Скидка:	**0,81**				
Итого к оплате:	**66,02**				

Счет действителен в течение 3 дней

Руководитель _____ **(Иванов А. П.)**

Главный бухгалтер _____ **(Петрова Т. В.)**

Рис. 58. Результат выполнения лабораторной работы

3. Сохраните документ в своей папке под именем ***Счет*** для дальнейшего сохранения на сервере.

МАТЕМАТИЧЕСКИЕ ПАКЕТЫ

Лабораторная работа № 9
РАБОТА В MATHCAD

Цель работы: познакомиться с интерфейсом математического пакета Mathcad. Освоить различие между текстовой и вычисляемой областями. Изучить основные типы данных, применяемых в документах Mathcad, и принципы их ввода-вывода. Выполнить простейшие вычисления.

Методические указания

Информация о существующих пакетах компьютерной алгебры представлена в приложении Г.

1. Запустите программу Mathcad. Перетаскивая панели инструментов и используя панель инструментов **Математика**, организуйте интерфейс программы так, как показано на рис. 59.

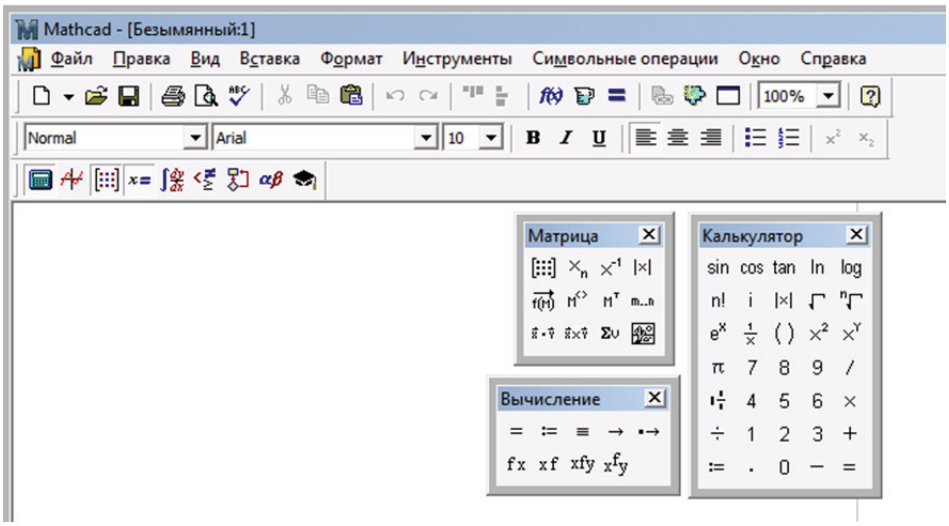


Рис. 59. Интерфейс программы Mathcad

Указания:

• для перетаскивания панели инструментов подведите курсор к левому краю панели и, зажав левую клавишу мыши, перетащите в нужное место. Чтобы убрать или добавить панель инструментов, воспользуйтесь командой **Панели инструментов** на вкладке главного меню **Вид**;

• для добавления дополнительных панелей инструментов **Калькулятор**, **Матрица** и **Вычисления** воспользуйтесь панелью инструментов **Математика** (рис. 60).

2. Вставьте текстовую область в документ, в которой сделайте запись: Лабораторная работа № 9. Отформатируйте запись: выберите гарнитуру Times New Roman, размер шрифта – 16 пт, выделите текстовый регион цветом. Результат представлен на рис. 61.

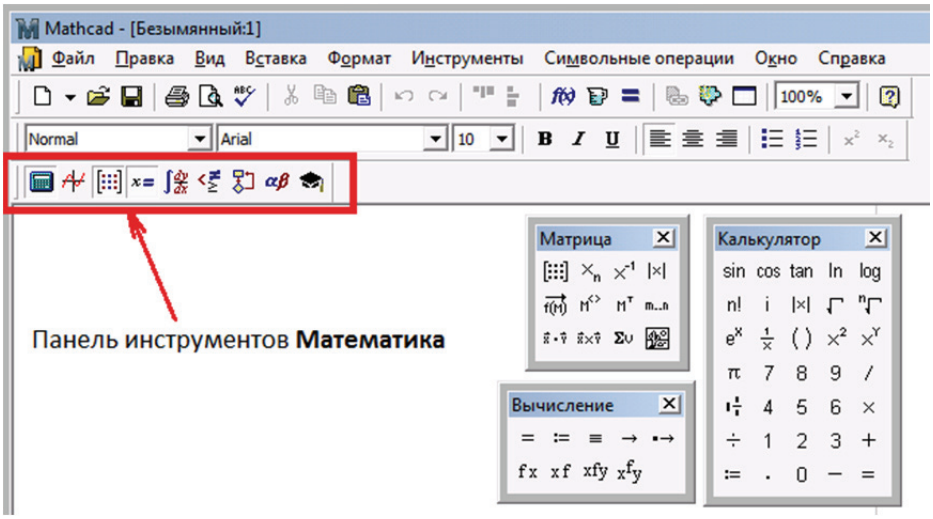


Рис. 60. Панель инструментов **Математика**

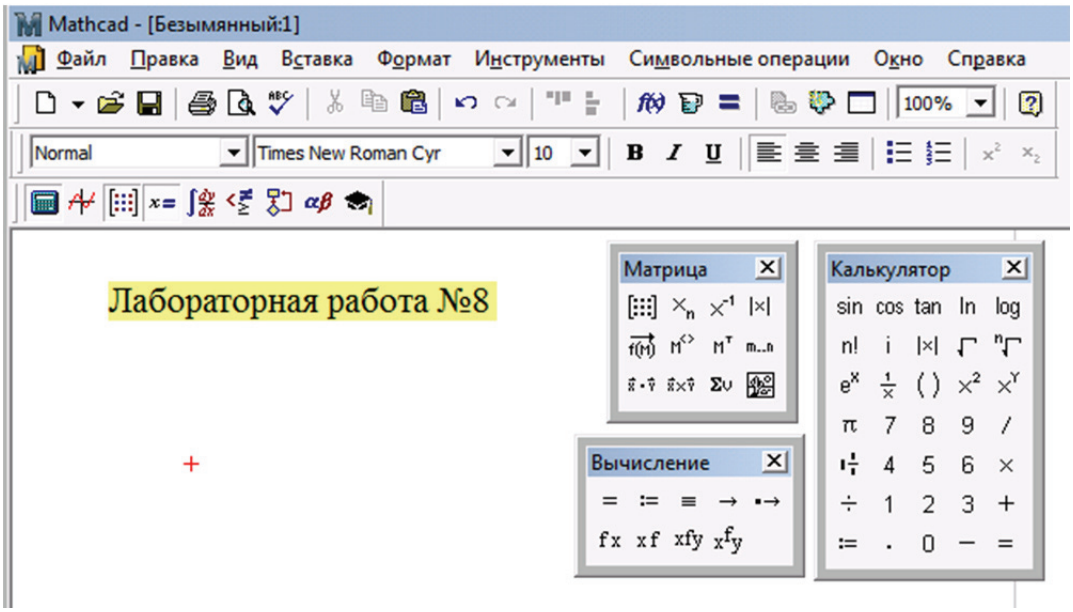


Рис. 61. Форматирование текстового региона

Указания:

• для вставки текстовой области воспользуйтесь командой **Регион текста** на вкладке главного меню **Вставка**. В появившейся области

впишите необходимый текст. Отформатируйте его, используя панель инструментов **Форматирование**;

• для выделения цветом текстового региона щелкните по нему правой кнопкой мыши и выберите команду **Свойства**. В появившемся диалоговом окне поставьте флажок на поле **Выделить регион**, выберите цвет и нажмите **ОК**. Обратите внимание на форму рамки вокруг текстового региона.

3. Присвойте переменным ***a*** и ***b*** произвольные значения и вычислите выражение для ***c***:

$$c = \frac{\sqrt{a^3 + b} + \cos(|b - 1|)}{\sqrt[4]{a^5 + 8b}}.$$

Указания:

• чтобы присвоить значение переменной, после имени переменной (***a*** или ***b***) поставьте знак присвоения «:=», а затем напишите значение переменной. Чтобы написать знак присвоения, нажмите соответствующую кнопку на панели инструментов **Калькулятор** либо сочетание клавиш **SHIFT+:**. При наборе формул необходимо помнить, что математические операции применяются к выделенной курсором области (подчеркивающая голубая линия курсора). Для выделения области воспользуйтесь клавишей **SPACE** (**Пробел**). Обратите внимание на форму рамки вокруг вычисляемого региона. Пример вычисления показан на рис. 62;

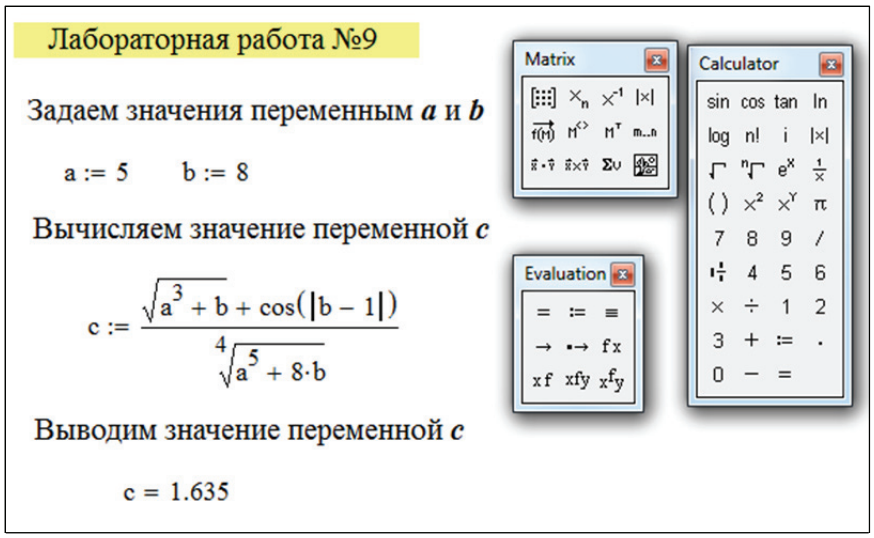


Рис. 62. Простейшие вычисления в Mathcad

• чтобы вывести значение переменной, после имени переменной ***c*** запишите знак численных вычислений «=», который выберите на панели

инструментов **Калькулятор** либо нажмите соответствующую кнопку на клавиатуре. При этом необходимо помнить, что программа считывает значения сверху вниз и слева направо, поэтому значения переменных ***a*** и ***b*** должны быть заданы выше формулы для ***c***.

4. Создайте вектор ***d*** – одномерный массив, содержащий один столбец из четырех элементов. Извлеките из массива второй и четвертый элементы, предварительно изменив нумерацию элементов так, чтобы она начиналась с единицы. Пример показан на рис. 63.

Указания:

• по умолчанию в Mathcad нумерация элементов массива начинается с нуля. Чтобы нумерация начиналась с единицы, присвойте встроенной переменной ORIGIN значение, равное единице: ORIGIN := 1. Для создания одномерного массива после имени массива (***d***) поставьте знак присвоения «:=», а затем на панели инструментов **Матрица** нажмите кнопку **Матрица или Вектор** ( ). В появившемся диалоговом окне выберите четыре строки и один столбец, затем заполните массив произвольными числами;

• чтобы вывести значения отдельных элементов массива, набрите имя массива (***d***) и на панели инструментов **Матрица** нажмите кнопку **Индекс** ( ) либо клавишу «[», введите номер нужного элемента массива и нажмите кнопку «=».

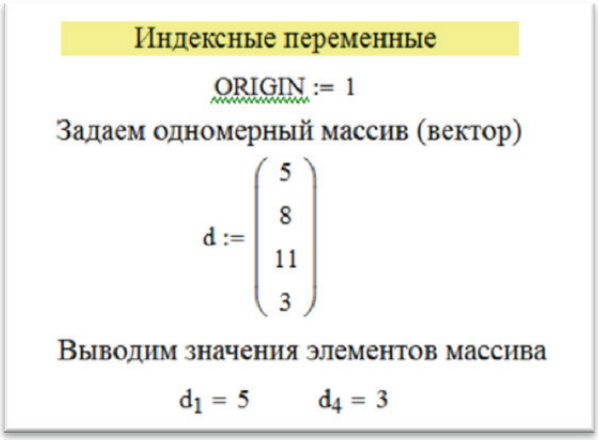


Рис. 63. Индексные переменные

5. Создайте вектор-строку с именем ***z***, состоящий из четырех элементов, извлеките из массива второй и четвертый элементы. Пример приведен на рис. 64.

Указания:

• для создания одномерного массива (вектор-строки) после имени массива (***z***) поставьте знак присвоения «:=», а затем на панели

инструментов **Матрица** нажмите кнопку **Матрица или Вектор** ( ). В появившемся диалоговом окне выберите одну строку и четыре столбца, затем заполните массив произвольными числами;

• чтобы вывести значения отдельных элементов вектор-строки, его предварительно необходимо транспонировать. Для этого поставьте курсор после массива и на панели инструментов **Матрица** нажмите кнопку **Транспонирование матрицы** ( ). После транспонирования наберите имя массива (**z**) и на панели инструментов **Матрица** нажмите кнопку **Индекс** ( ) либо клавишу «[», введите номер нужного элемента массива и нажмите кнопку «=».

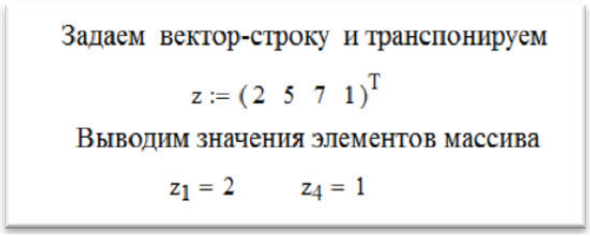


Рис. 64. Пример транспонирования матрицы

6. Создайте вектор (одномерный массив) с именем ***f***, состоящий из четырех элементов, задавая отдельные элементы вектора. Пример приведен на рис. 65.

Указания:

• чтобы задать отдельные значения элементов массива, наберите имя массива (***f***) и на панели инструментов **Матрица** нажмите кнопку **Индекс** ( ) либо клавишу «[», введите номер нужного элемента массива и поставьте знак присвоения «:=», затем введите значение элемента. При этом необходимо помнить, что размер массива определяется номером последнего элемента массива. Пропущенным элементам массива по умолчанию присваивается значение, равное нулю;

• чтобы вывести заданный массив, введите имя массива (***f***) и нажмите кнопку «=».

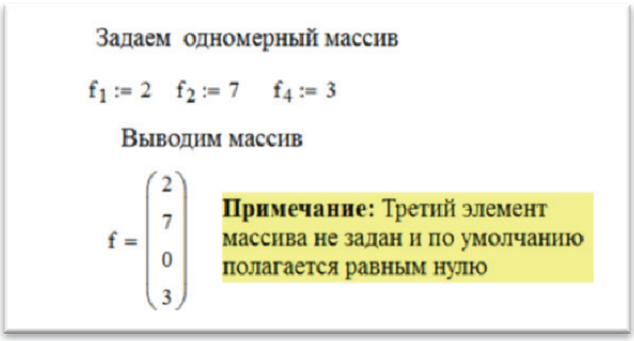


Рис. 65. Пример создания одномерного массива

7. Вычислите:

а) сумму элементов массивов ***d*** и ***z***;

б) сумму последних двух элементов массива ***z***;

в) сумму квадратов элементов массива ***d*** и элементов массива ***z***. Пример приведен на рис. 66.

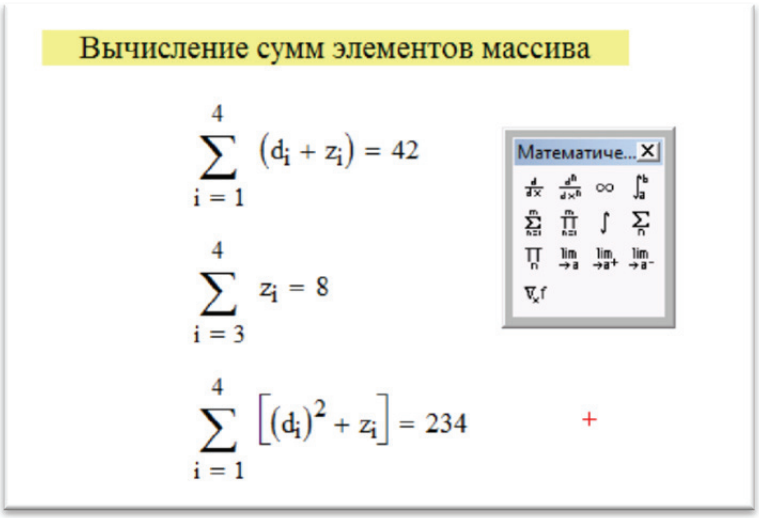


Рис. 66. Вычисление сумм элементов

Указания:

• чтобы вычислить сумму элементов массивов ***d*** и ***z***, на панели инструментов **Математический анализ** нажмите кнопку **Сумма** ( ) и введите данные следующим образом: $\sum_{i=1}^{4}(d_i + z_i)$, нажмите клавишу «=» (цифра снизу знака суммы определяет первый элемент массива, вверху – последний элемент массива, по ним осуществляется суммирование);

• сумму последних двух элементов массива ***z*** можно рассчитать по формуле $\sum_{i=3}^{4} z_i$;

• сумму квадратов элементов массива ***d*** и элементов массива ***z*** можно вычислить по формуле $\sum_{i=1}^{4}((d_i)^2 + z_i)$.

Лабораторная работа № 10
РАБОТА В MATHCAD. СИМВОЛЬНЫЙ ПРОЦЕССОР И ПОСТРОЕНИЕ ГРАФИКОВ

Цель работы: научиться создавать пользовательские функции и выполнять простейшие символьные вычисления. Научиться определять ранжированную переменную и пользоваться графопостроителем.

Методические указания

1. Создайте пользовательскую функцию $f(x) = x^3 + \sin(4x)$. Пример представлен на рис. 67.

Указание. При создании пользовательской функции в Mathcad необходимо помнить, что после имени функции (*f*) необходимо в скобках указать переменную от которой функция зависит, например $f(x)$. После имени функции необходимо поставить знак присвоить значение «:=» и сформулировать тело функции.

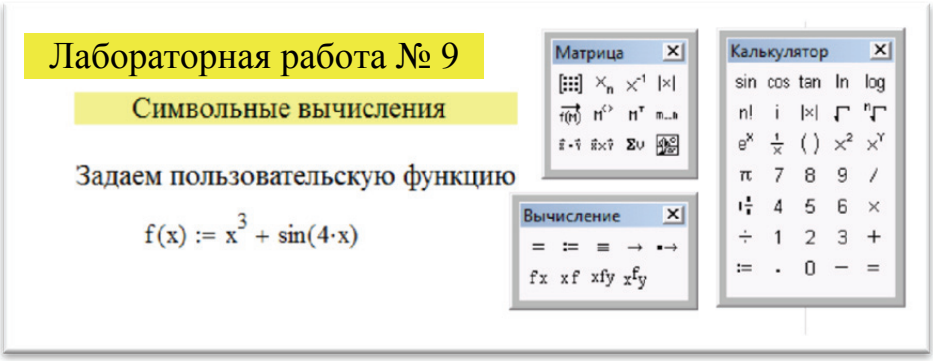


Рис. 67. Пользовательская функция

2. Вычислите в символьном виде первую и вторую производные от пользовательской функции. Пример приведен на рис. 68.

Указания:

- для вычисления производной в символьном виде на панели инструментов **Математический анализ** нажмите кнопку **Производная** ( ), введите имя функции, которую необходимо дифференцировать и переменную, по которой следует дифференцировать. После ввода на панели инструментов **Вычисление** нажмите кнопку **Вычислить аналитически** (→). Для нахождения второй производной на панели инструментов **Математический анализ** нажмите кнопку ***n*-я производная** ( ) и проведите аналогичные действия.

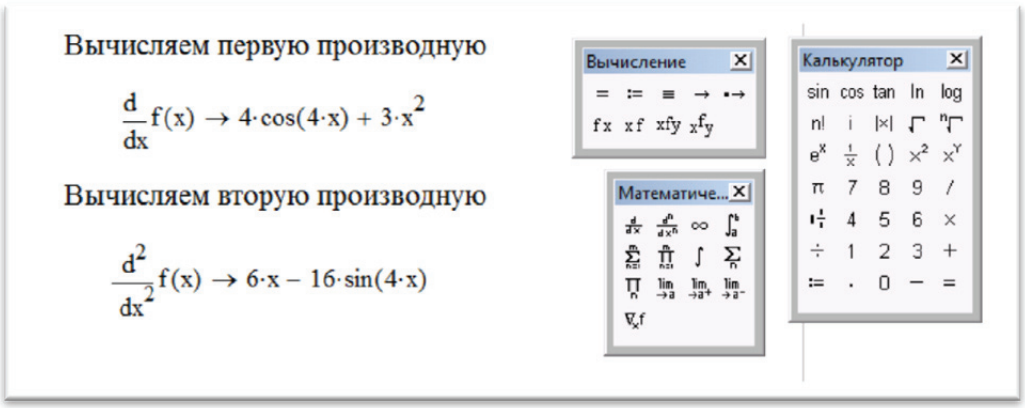


Рис. 68. Аналитическое вычисление производных

3. Рассчитайте аналитически пределы функций: $\lim\limits_{x\to 0}\dfrac{\sin(x)}{x}$, $\lim\limits_{x\to 1} f(x)$, $\lim\limits_{x\to\infty} f(x)$. Пример приведен на рис. 69.

Указание. Для аналитического вычисления функций на панели инструментов **Математический анализ** нажмите кнопку **Двусторонний предел** ($\lim\limits_{\to a}$) и заполните необходимыми данными. Знак бесконечности (∞) находится на панели инструментов **Математический анализ**.

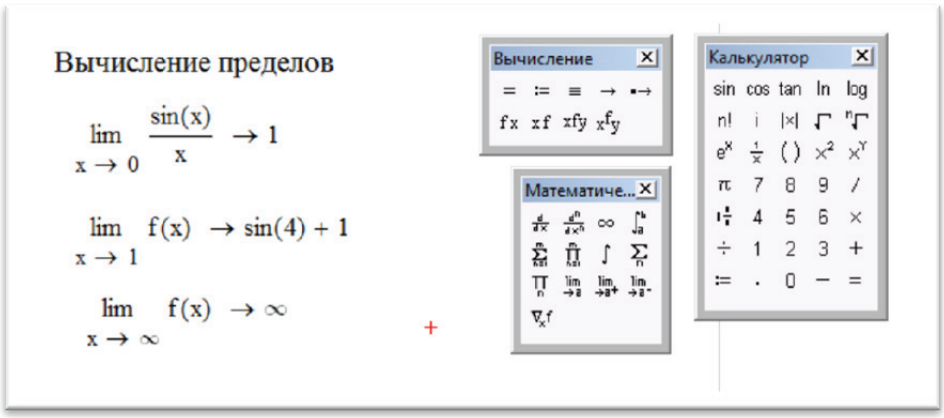


Рис. 69. Вычисление пределов функций

4. Задайте ранжированную переменную x в диапазоне от $-\pi$ до π с шагом 0,1 и постройте график функции $z(x) = x^2 + \cos(x)$. Пример приведен на рис. 70.

Указания:

• чтобы задать ранжированную переменную, после имени переменной x поставьте знак присвоить значение (:=) и на панели инструментов **Матрица** нажмите кнопку **Переменная-диапазон** (m..n),

укажите границы диапазона. Значение π является встроенной константой и этот символ можно набрать из панели инструментов **Греческие символы**. По умолчанию в Mathcad шаг ранжированной переменной равен единице. Чтобы определить другой шаг, после цифры, определяющей левую границу диапазона, поставьте запятую и укажите цифру, которая должна идти следующей в ряде значений ранжированной переменной;

• для построения графика функции на панели инструментов **График** нажмите кнопку **График X-Y** ( ). В ячейке, расположенной под осью абсцисс, укажите независимую переменную x, в ячейке рядом с осью ординат задайте функцию $z(x)$, график которой хотите построить. Если эта функция была определена заранее, то в ячейку достаточно ввести $z(x)$, в противном случае следует ввести изображаемую функцию в явном виде (например, cos(x)).

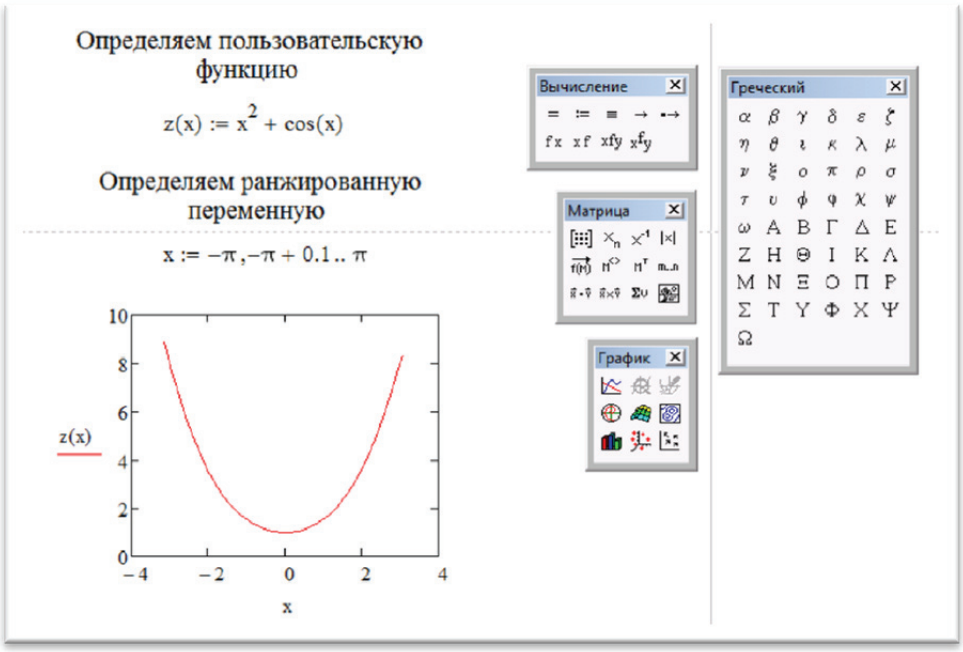


Рис. 70. Ранжированная переменная и построение графика

5. Добавьте на построенный график кривую производной от функции $z(x)$ и отформатируйте график следующим образом:

а) сделайте пересекающиеся оси на графике;

б) отобразите линии сетки на графике;

в) обе кривые сделайте сплошной линией, толщина которой два;

г) сделайте заголовок графика;

д) установите границы отображения графика по оси x: от –3 до 3, по оси y: от –6 до 8. Пример приведен на рис. 71.

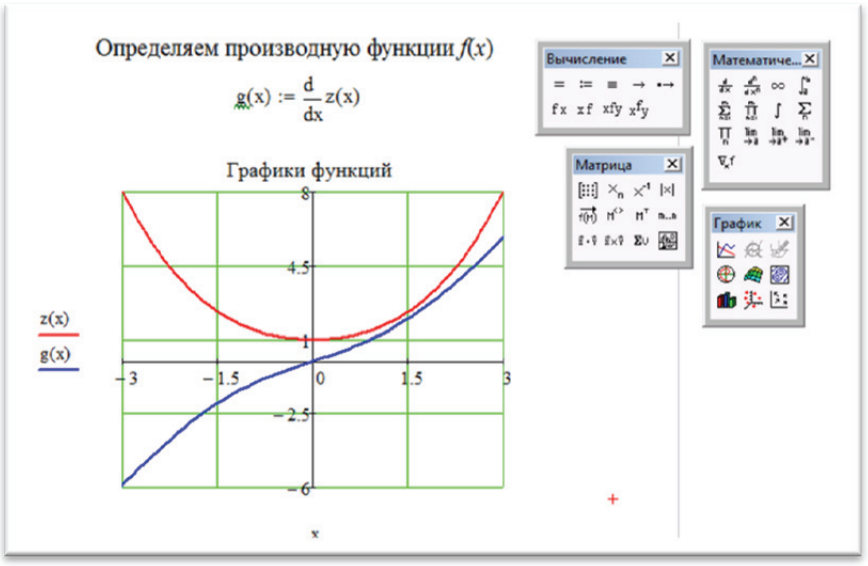


Рис. 71. Форматирование графика функций

Указания:

• присвойте имя производной от функции, например $g(x)$. Для добавления кривой на график на оси ординат после первой функции $z(x)$ поставьте запятую, в появившемся маркере впишите имя производной и щелкните на свободном месте рабочего листа или нажмите на клавиатуре кнопку **Enter**;

• для форматирования графика вызовите панель форматирования, дважды щелкнув на поле графика. Для форматирования графика следует воспользоваться вкладками **Оси X-Y**, **Трассировка**, **Подписи**.

Лабораторная работа № 11
РАБОТА В MATHCAD.
РЕШЕНИЕ УРАВНЕНИЙ И СИСТЕМ УРАВНЕНИЙ

Цель работы: научиться решать алгебраические уравнения и системы уравнений, используя встроенные функции Mathcad.

Методические указания

1. Решите квадратное уравнение $x^2 + 2x - 6 = 0$, используя встроенную функцию root. Проверьте решение, построив график функции. Пример представлен на рис. 72.

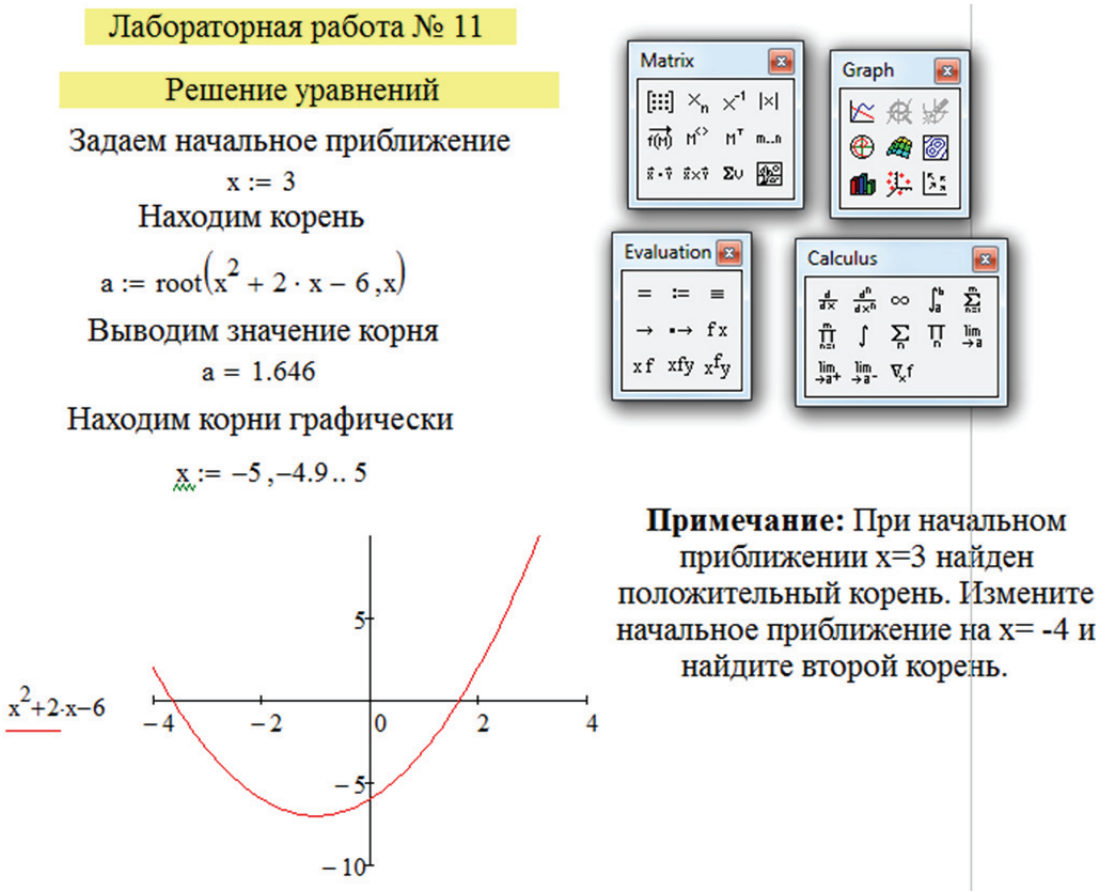


Рис. 72. Решение уравнения с помощью функции root

Указания:

• аргументами функции root являются выражение и переменная, входящая в выражение *root*(*f*(*x*), *x*). Функция ищет значение переменной *x*, при котором выражение *f*(*x*) обращается в ноль. Функция возвращает значение переменной, которое обращает выражение в ноль.

Определите начальное значение переменной x, например: $x = 3$. Важно понимать, что выбор начального приближения влияет на корень, возвращаемый MathCad (если выражение имеет несколько корней – возвращается ближайший к начальному приближению корень);

• определите переменную, например a, как корень уравнения и запишите: $a := root(x^2 + 2x - 6, x)$;

• чтобы вывести значение корня запишите: $a =$;

• постройте график функции $x^2 + 2x - 6$ и проверьте решение.

2. Найдите точки пересечения окружности $x^2 + y^2 = 6$ и прямой $x + y = 2$, используя блок **Given – Find**. Проверьте решение графически. Пример представлен на рис. 73.

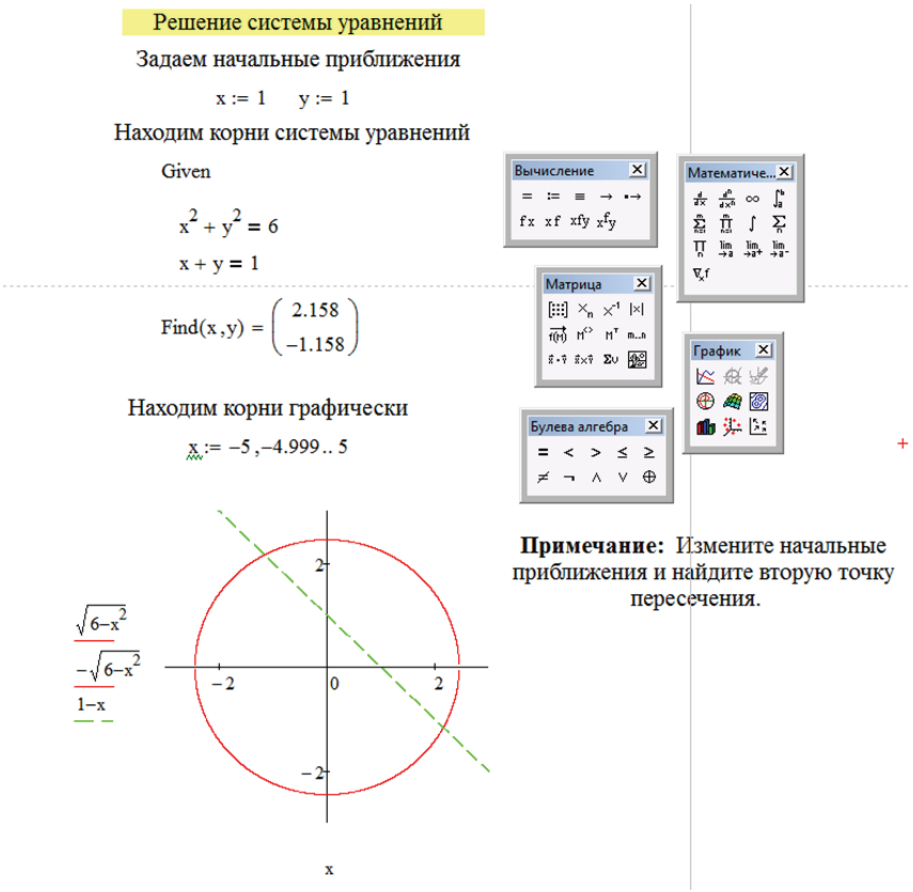


Рис. 73. Решение систем уравнений

Указания:

• для нахождения точек пересечения окружности и прямой совместно решите два уравнения. Задайте начальные приближения для всех неизвестных, входящих в систему уравнений, например: $x := 1$, $x := 1$;

- напечатайте ключевое слово **Given**. Оно указывает Mathcad, что далее следует система уравнений. При печати слова **Given** убедитесь, что при этом вы не находитесь в текстовой области;
- введите уравнения в любом порядке ниже ключевого слова **Given**. При этом удостоверьтесь, что между левыми и правыми частями уравнений стоит символ **логического равно** «=» из панели инструментов **Булева алгебра**;
- напечатайте функцию **Find** и в скобках укажите искомые неизвестные и нажмите равно «=»;
- проверьте решение графически, построив график двух функций.

ПРОГРАММЫ ДЛЯ СОЗДАНИЯ ПРЕЗЕНТАЦИЙ

Лабораторная работа № 12
РАБОТА В POWERPOINT

Цель работы: в PowerPoint создать презентацию, в которой будут отражены результаты, полученные в лабораторных работах № 4–10. Освоить работу с гиперссылками.

Методические указания

1. Создайте титульный слайд, который содержит ваше ФИО, номер группы и подгруппы. Пример приведен на рис. 74.

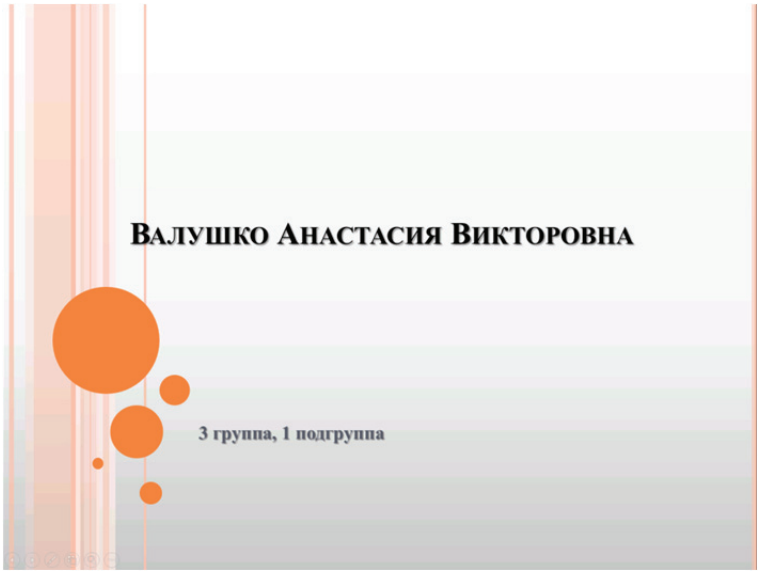


Рис. 74. Пример титульного листа презентации

2. Добавте слайд, содержащий гиперссылки на документы Word ***Отчет*** (лабораторная работа № 6) и ***Счет*** (лабораторная работа № 7). Создайте на слайде кнопки, обеспечивающие навигацию по презентации. Пример слайда продемонстрирован на рис. 75.

3. Разработайте слайд, который содержит гиперссылки на книги Excel ***Абитуриент*** (лабораторные работы № 4, 5) и ***Счет*** (лабораторная работа № 7). Создайте на слайде кнопки, обеспечивающие навигацию по презентации. Пример слайда приведен на рис. 76.

4. Добавьте слайд, содержащий гиперссылки на документы Mathcad (лабораторные работы № 8–10). Создайте на слайде кнопки, обеспечивающие навигацию по презентации. Пример слайда продемонстрирован на рис. 77.

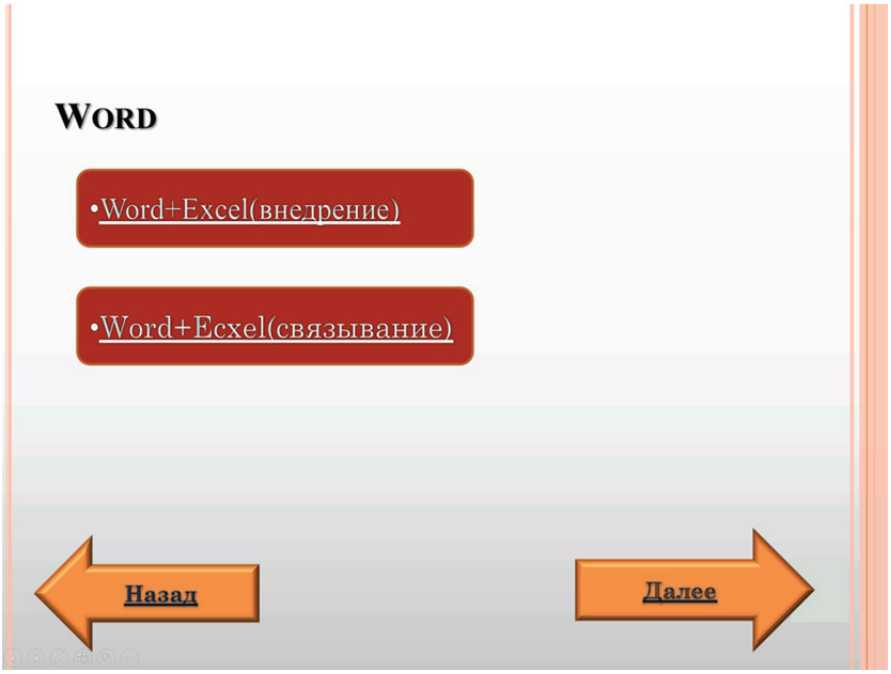


Рис. 75. Слайд презентации, посвященный технологиям связывания и внедрения в MS Word

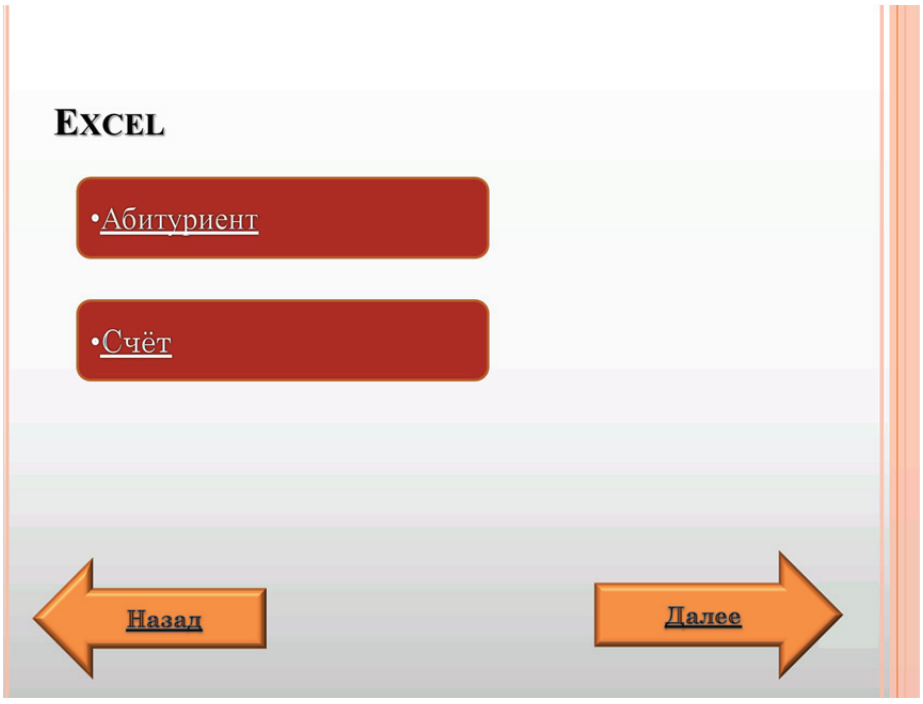


Рис. 76. Слайд презентации, посвященный расчетам в MS Word

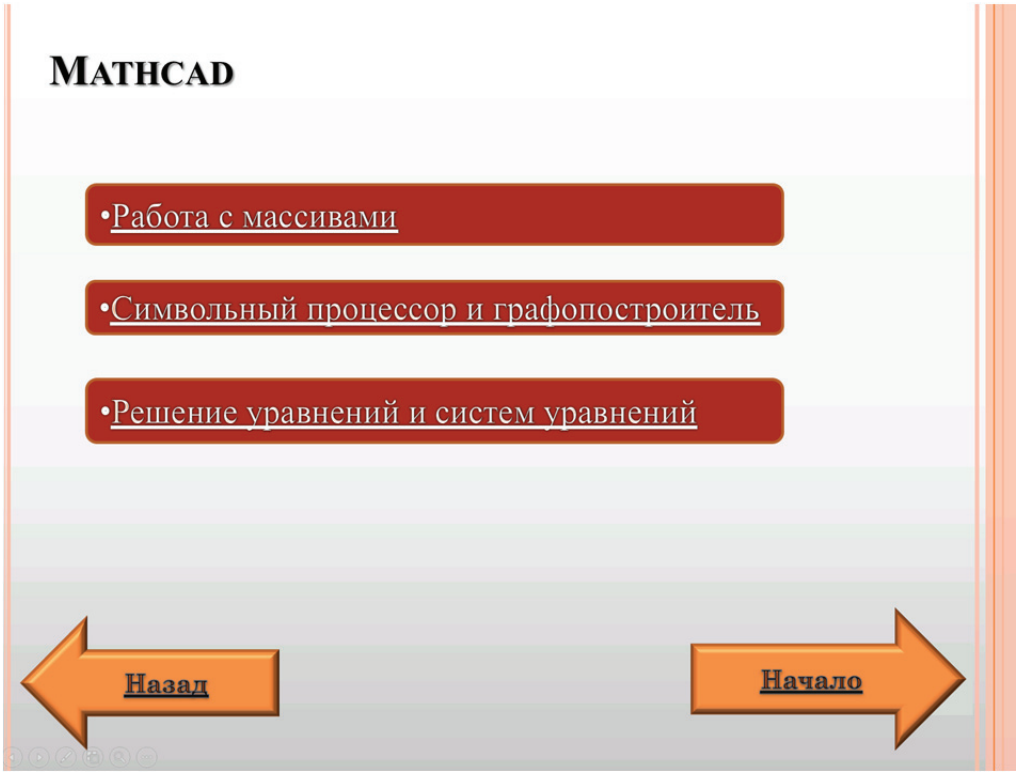


Рис. 77. Слайд презентации, посвященный выполненным работам в Mathcad

5. Создайте заключительный слайд и настройте показ презентации таким образом, чтобы переход с титульного слайда на второй происходил автоматически через 5 с, а остальные переходы осуществлялись по щелчку мыши.

СРЕДСТВА ДЛЯ РАЗРАБОТКИ ПРОГРАММ ПРИЛОЖЕНИЙ

Лабораторная работа № 13
РАБОТА В VISUAL BASIC

Цель работы: изучить среду программирования Visual Basic. Научиться создавать программы линейной структуры, используя программу Microsoft Visual Basic.

Методические указания

1. Запустите систему программирования VB и изучите основные компоненты ее среды на экране (вызовите при необходимости на экран недостающие основные окна среды, используя соответствующие команды пункта **View** из головного меню).

2. Измените с помощью мыши размеры и расположение окон среды VB на экране (примерно так, как показано в приложении Д на рис. Д.2).

3. Преобразуйте форму, активизируя мышью соответствующие свойства в окне **Properties** (рис. 78), заголовок формы (**Caption**), цвет фона (**BackColor**), тип шрифта (**Font**), тип границ (**BorderStyle**) и по желанию другие ее свойства (см. приложение Д).

4. Добавьте на форму командную кнопку (**CommandButton**) из окна **Toolbox** (рис. 79). Измените ее свойства: имя (**Name**), заголовок кнопки (**Caption**)**,** тип шрифта для него (**Font**), размеры (**Hieght** и **Width**), положение на форме (**Left** и **Top**).

5. Создайте на форме еще одну командную кнопку, задайте для нее новые значения соответствующих свойств.

6. Добавьте в форму окно рисунка **PictureBox** (графическое окно), в которое загрузите какую-либо картинку, активизировав свойство **Picture** (в окне **Properties**, рис. 78) и выбрав для этого в появившемся диалоговом окне соответствующий графический файл (например, рисунок, созданный ранее в программе **Paint)**.

7. Создайте на форме окно для ввода текста **TextBox** (текстовое окно, рис. 79) и задать его новые свойства (рис. 78): подберите цвет фона (**BackColor**), цвет шрифта (**ForeColor**), введите любой текст в свойстве **Text**, опробуйте различные шрифты, изменяя различные характеристики в диалоговом окне свойства **Font**, задайте возможность

ввода текста в несколько строк (**MultiLine**), установите линии прокрутки текста в окне (**ScrollBars**).

8. Рядом с текстовым окном сделайте надпись, разместив на форме объект **Label** (метка) (рис. 79) и задав соответствующее значение для его заголовка в свойстве **Caption** (рис. 78). Надпись заключите в рамку с помощью свойства **BorderStyle** (стиль границ), опробуйте свойства **Alignment** (выравнивание) и **AutoSize** (подгонка размера), измените цвета и шрифт для заголовка.

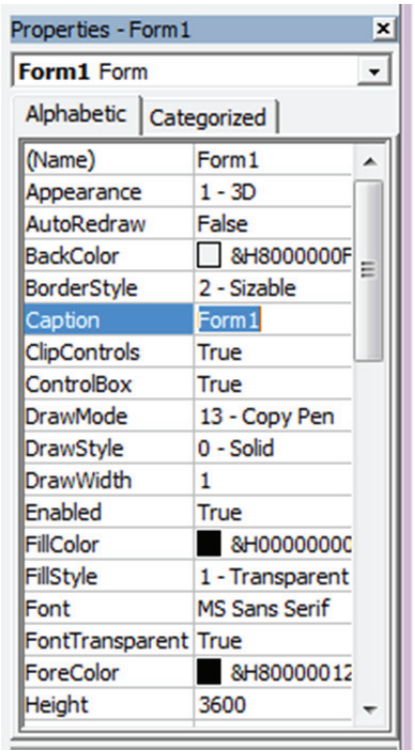


Рис. 78. Окно **Properties**

Рис. 79. Окно **Toolbox**

9. Запустите созданное приложение (пока только с интерфейсом, без программного кода) на выполнение, щелкнув кнопку **Start** на пиктографическом меню в главном окне среды VB (рис. 80), убедитесь, что командные кнопки «работают» (нажимаются), но не выполняют никаких действий.

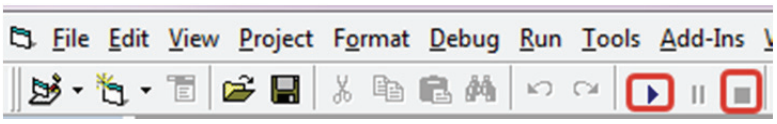


Рис. 80. Команды запуска и остановки программы

10. Закончите работу с приложением, выполнив команду **End** (рис. 80).

11. Перейдите к режиму написания программного кода (в окне **Project** нажмите кнопку **View Code**, рис. 81, или дважды кликните по элементу **CommandBotton** на форме) и выберите из списка объектов окно формы **Form**.

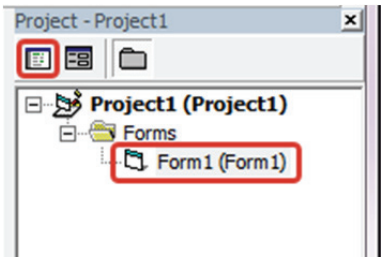


Рис. 81. Окно **Project**

12. Выберите из списка процедур обработки событий, связанных с формой, процедуру щелчка мышью **Click** (рис. 82).

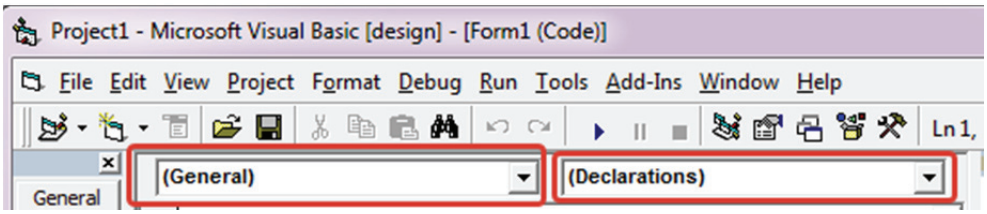


Рис. 82. Команды выбора объекта и обработки события

13. Внутри процедурных скобок данной процедуры напишите строку программы, выводящую на форму слово «Привет».

Print «Привет»

Если переход на кириллицу в тексте программы не происходит, установите соответствующий шрифт, выполнив следующую цепочку команд: **Tools/Options.../Editor Format/Font**, и выберите в списке шрифт, имя которого заканчивается символами **Cyr**:

Заметим, что объект форма **Form1** для воздействия на него методом **Print** можно не указывать – этот объект установлен в VB для методов и свойств на данной форме по умолчанию.

14. Запустите приложение на выполнение и кликните мышью в любом месте формы. В левом верхнем углу должно появиться слово «Привет». Завершите работу приложения нажатием на кнопку **End**.

15. Выберите для формы событие **Load** (загрузка формы в оперативную память после запуска приложения) и запишите для его процедуры программу, изменяющую при запуске приложения заголовок формы на слова «Первая программа».

Caption = «Первая программа»

16. Запустите приложение и проверьте работу этой процедуры.

17. Напишите для командной кнопки **Command1** программный код, устанавливающий по щелчку мыши размер букв, выводимых на форму, равный 24 пт (**Font.Size = 24**).

18. Запустите приложение на выполнение и пощелкайте по форме, командной кнопке – размер букв должен меняться.

19. Создайте для второй командной кнопки **Command2** процедуру, выводящую по щелчку мышью по ней в графическое окно слово «Студент» с размером букв 24 пт.

Picture1.Font.size=24
Picture1.Print «Студент»

20. Нарисуйте на форме новую командную кнопку, очищающую форму и графическое окно методом **Cls**, запустите приложение и проверьте работу этой кнопки.

21. Создайте для обработки графического окна две процедуры – одинарный щелчок по графическому окну должен сделать его невидимым (для графического окна задать свойство **Visible = False**), а двойной щелчок по форме – видимым (для графического окна задать свойство **Visible = True**).

22. Добавьте на форме текстовое окно и напишите для него процедуры, выводящие в окно при одинарном щелчке по окну фамилию, а при двойном щелчке – имя и отчество.

23. Сохраните в своей папке файлы формы и проекта приложения, а также его **exe**-файл.

24. Закройте редактор VB.

25. Найдите в программе **Проводник** исполнимый exe-файл разработанного приложения, запустите его на выполнение и проверьте работу всех процедур.

Лабораторная работа № 14
РАБОТА В VISUAL BASIC.
СОЗДАНИЕ ПРОГРАММЫ С ЛИНЕЙНОЙ СТРУКТУРОЙ

Цель работы: научиться создавать программы с линейной структурой, используя программу Microsoft Visual Basic.

Методические указания

1. Задайте для формы заголовок (**Caption**) «Вычисления по формулам» и размер шрифта выводимых на форму результатов, равный 20 пт (рис. 78, 79).

2. Создайте на форме командную кнопку с заголовком «Пример расчета» (см. лабораторную работу № 13, п. 4–8).

3. Наберите для созданной командной кнопки приведенный выше текст программы (см. рис. Д.19 на с. 108).

4. Запустите приложение, выполните программу щелчком мыши по кнопке **Пример расчета**. Для переменной ***a*** задайте значение 3,75, а для ***m*** – 10,46. Значение переменной ***w*** должно быть равно минус 60,98, а переменной ***z*** – 81,32.

5. Создайте на форме командную кнопку с заголовком «Очистка формы» и с программным кодом, состоящим из одного оператора **Cls.**

6. Запустите снова приложение, повторите расчет примера и нажатием на кнопку «Очистка формы» проверьте работу ее программы. Измените цвет фона формы, цвет и шрифты выводимой на форму информации.

7. Создайте на форме графическое окно и еще одну командную кнопку с заголовком «Расчет в окно».

8. Добавьте новую форму и в эту форму графическое окно.

9. Создайте программу при щелчке по кнопке **Расчет в окно** для вывода результатов расчета ***w*** в графическое окно на первой форме и ***z*** в графическое окно на второй форме. Чтобы не повторять с клавиатуры набор текста программы расчета, можно воспользоваться копированием через буфер обмена среды Windows.

10. Отформатируйте вывод результатов в окно с точностью до двух цифр после десятичной запятой, для чего используйте функцию **Format** в операторе **Print.**

11. Создайте командную кнопку очистки графических окон и формы с заголовком «Очистка окна» и проверьте ее работу.

12. Создайте откомпилированный вариант разработанного приложения и сохраните его.

13. Сохраните файлы формы и проекта в своей папке.

14. Выйдите из среды системы программирования VB.

Лабораторная работа № 15
РАБОТА В VISUAL BASIC. СОЗДАНИЕ ПРОГРАММЫ С ЦИКЛИЧЕСКОЙ СТРУКТУРОЙ

Цель работы: научиться создавать программы с циклической структурой, используя программу Microsoft Visual Basic.

Методические указания

Общие сведения по работе с циклами в Visual Basic приведены в приложении Д (см. с. 112).

1. Создайте на форме пользовательский интерфейс для проекта нового приложения, аналогичный представленному на рис. 83.

2. Для кнопок **Программа1**, **Программа2**, **Программа3** напишите программы вычисления таблицы умножения для числа ***n*** (***n*** – номер варианта) с использованием соответствующего оператора цикла.

3. Вывод результата осуществите в соответствующее графическое окно, расположив таблицу умножения по центру окна.

4. Создайте на форме командную кнопку с заголовком «Очистка» и с программным кодом, выполняющим очистку всех *Picturebox*.

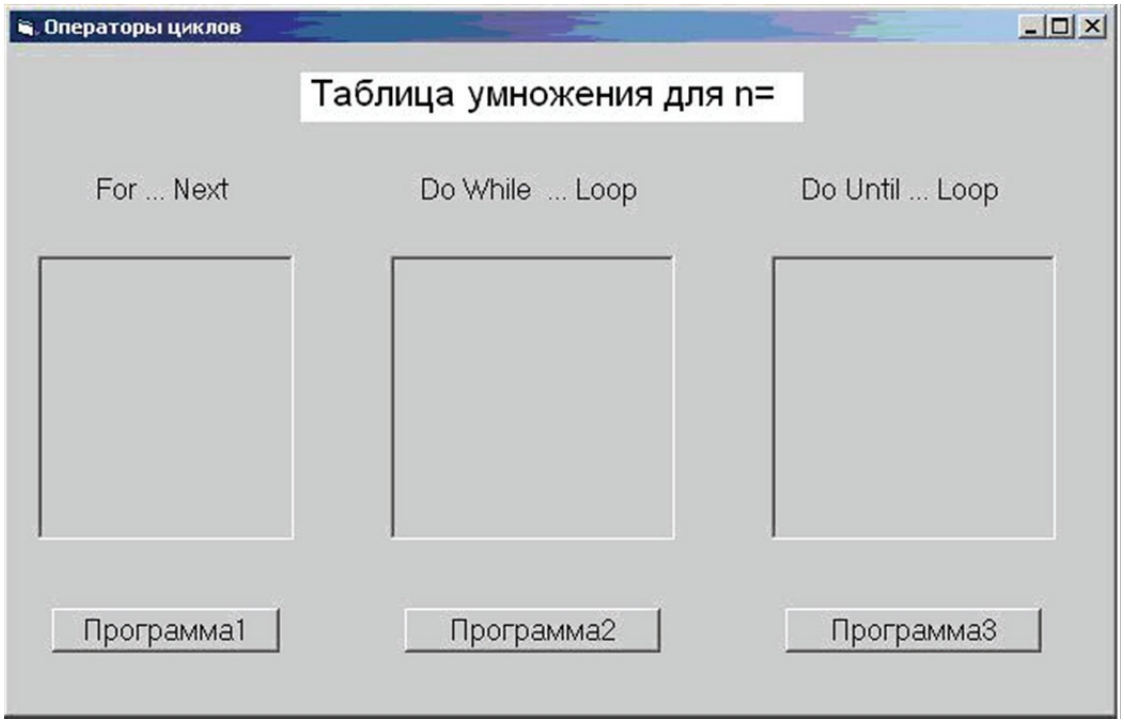


Рис. 83. Общий вид формы

Лабораторная работа № 16
РАБОТА В VISUAL BASIC. СОЗДАНИЕ ПРОГРАММЫ, ОБРАБАТЫВАЮЩЕЙ ДАННЫЕ, СОХРАНЕННЫЕ В ОДНОМЕРНЫХ МАССИВАХ

Цель работы: научиться создавать программы, обрабатывающие данные, сохраненные в одномерных массивах, используя программу Microsoft Visual Basic.

Методические указания

Общие сведения по работе с одномерными массивами в Visual Basic приведены в приложении Д (см. с. 114).

1. Создайте на форме пользовательский интерфейс, представленный на рис. 84.

2. Для командной кнопки **Сумма и произведение** создайте и выполните программу вычислений суммы и произведения элементов массива. Выведите значения элементов массива и результатов работы программы в графические окна.

3. Выполните вычисление суммы положительных и суммы отрицательных элементов массива. Выведите сумму положительных элементов в графическое окно на первой форме, отрицательных – на второй.

4. Для командной кнопки **Очистка** создайте программу очистки всех графических окон на формах и проверьте ее работу.

5. Напишите программу для кнопки **Моя задача**. Вывод выполните в соответствующие графические окна на первой и второй форме.

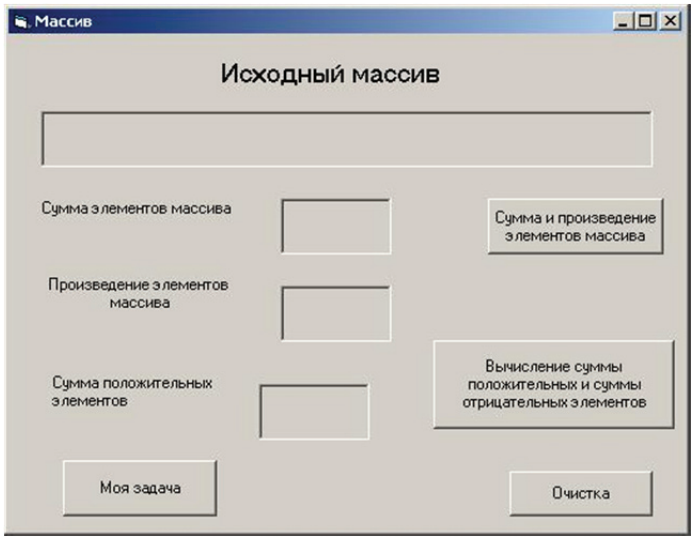


Рис. 84. Примерный интерфейс задачи

Приложение А

КОМПЬЮТЕРНЫЕ СЕТИ

При физическом соединении двух или более компьютеров образуется компьютерная сеть. В общем случае для создания компьютерных сетей необходимо специальное аппаратное (сетевое оборудование) и программное обеспечение (сетевые программные средства). Простейшее соединение двух компьютеров для обмена данными называется прямым соединением. Для создания прямого соединения компьютеров, работающих в операционной системе Windows, не требуется ни специального аппаратного, ни программного обеспечения. В этом случае аппаратными средствами являются стандартные порты ввода/вывода (последовательный или параллельный), а в качестве программного обеспечения используется стандартное средство, имеющееся в составе операционной системы (Пуск → Программы → Стандартные → Связь → Прямое соединение).

История появления и развития компьютерных сетей

Первую в мире локальную вычислительную сеть (ЛВС) создал в 1967 г. Дональд Дэвис (Donald Davies) в Национальной физической лаборатории Великобритании (British National Physics Laboratory). К началу 70-х гг. сеть работала с пиковой скоростью 0,25 Мбит/с, обслуживая около 200 пользователей.

Развитие компьютерных сетей происходило, в первую очередь, за счет развития двух более крупных направлений технологии – вычислительной техники и коммуникаций. Первые попытки создать возможность работы с вычислительной техникой нескольких пользователей заключались в загрузке в мейнфрейм (основной компьютер) нескольких готовых пакетов данных, которые были заранее подготовлены и нуждались в обработке.

Следующим прообразом компьютерных сетей стало создание отдельных терминалов, имеющих полноценные собственные устройства ввода/вывода и работающие напрямую с одним общим компьютером. Для самого пользователя работа за таким устройством была куда более удобной – он мог не замечать, что мощности компьютера параллельно используются еще несколькими людьми. Именно тогда стали появляться первые сети, чей принцип работы заключался лишь в банальном физическом удалении терминалов на определенные расстояния.

Как только начали появляться более компактные компьютеры – это произошло в 70-х гг., – позволить себе их установку могли все больше предприятий, поэтому необходимость использования какого-либо средства связи возрастала и тогда возникли первые приближенные к современным способы объединения компьютеров в сеть и потребность в монтаже компьютерных сетей.

APRANET и появление полноценных сетей

В 1969 г. произошло знаковое событие – Министерство обороны США приняло решение об объединении всех основных компьютерных узлов в общую сеть. Передача данных осуществлялась между ними по коммутируемому кабелю, а для ее осуществления были созданы специальные операционные системы и огромное количество сложных сопутствующих протоколов.

Впоследствии коммутируемые кабели телефонных сетей станут одним из основных способов передачи данных вплоть до середины 80-х гг.

При упоминании о компьютерных сетях сейчас имеют в виду две основных их разновидности. Под подключением WAN (Wide Area Network) подразумевают объединение удаленных физически друг от друга компьютеров, а также простой выход в Интернет, в то время как LAN – это закрытая сеть, объединяющая физически близкие компьютеры, и способная быть полностью изолированной от каких-либо других соединений.

Однако на ранних этапах развития компьютеров нужды в LAN-сетях не было – их заменяли стандартные комплексы из мейнфреймов и терминалов, хотя удаленная передача данных была крайне важным и приоритетным направлением исследований.

Важную роль в развитии сетей сыграло появление персональных компьютеров, унификация их комплектующих и программного обеспечения. Так начали появляться первые сетевые протоколы – это произошло в 80-х гг. К концу века однозначным лидером среди них стал протокол Ethernet, способный обеспечивать скорость передачи данных в первом поколении своего развития со скоростью 10 Мбит/с, а на данный момент поддерживающий скорость передачи, превышающую 1 Гбит/с.

Краткая история развития компьютерных сетей

1. 1950–1960 гг. – первые попытки объединения мейнфрейма с терминалами.
2. 1969 г. – появление APRANET и использование телефонных сетей для передачи данных.

3. 1970–1974 гг. – возникновение мини-компьютеров и создание вручную настраиваемых локальных сетей.

4. 1974 г. появление первой стандартизованной сетевой архитектуры IBM SNA, а также стандартизации X.25.

5. 1980–1985 гг. возникновение персональных компьютеров, появление Интернета в близком к современности виде. Использование стека TCP/IP на всех узлах. Возникновение стандартных технологий локальных сетевых протоколов Ethernet, FDDI, Token Ring.

6. 1986–1987 гг. – старт коммерческого использования Интернета.

7. 1991 г. появление протокола Web и первых интернет-сайтов.

8. 1995–2000 гг. развитие Web и массовая популяризация компьютеров.

9. 2000–2010 гг. – использование беспроводных сетей, снижение стоимости передачи единицы информации сразу в несколько тысяч раз.

Классификация компьютерных сетей

По территориальной распространенности:

- PAN (Personal Area Network) – персональная сеть, предназначенная для взаимодействия различных устройств, принадлежащих одному владельцу;
- LAN (Local Area Network) – локальные сети, имеющие замкнутую инфраструктуру до выхода на поставщиков услуг. Термин LAN может описывать и маленькую офисную сеть, и сеть уровня большого завода, занимающего несколько сотен гектаров. Зарубежные источники дают даже близкую оценку – около шести миль (10 км) в радиусе. Локальные сети являются сетями закрытого типа, доступ к ним разрешен только ограниченному кругу пользователей, для которых работа в такой сети непосредственно связана с их профессиональной деятельностью;
- CAN (Campus Area Network – кампусная сеть) – объединяет локальные сети близко расположенных зданий;
- MAN (Metropolitan Area Network) – городские сети между учреждениями в пределах одного или нескольких городов, связывающие много локальных вычислительных сетей;
- WAN (Wide Area Network) – глобальная сеть, покрывающая большие географические регионы, включающие в себя как локальные сети, так и прочие телекоммуникационные сети и устройства.

По типу среды передачи:

- проводные (телефонный провод, коаксиальный кабель, витая пара, волоконно-оптический кабель);

• беспроводные (передачей информации по радиоволнам в определенном частотном диапазоне).

По скорости передачи:

• низкоскоростные (до 10 Мбит/с);
• среднескоростные (до 100 Мбит/с);
• высокоскоростные (свыше 100 Мбит/с).

По сетевым операционным системам:

• на основе Windows;
• UNIX;
• NetWare;
• Cisco.

По типу сетевой топологии

• шина;
• звезда;
• кольцо;
• Fat Tree.

Топология соединений

Все компьютеры в локальной сети соединены линиями связи. Геометрическое расположение линий связи относительно узлов сети и физическое подключение узлов к сети называется физической топологией. В зависимости от топологии различают сети: шинной, кольцевой, звездной, иерархической и произвольной структуры.

Выделяют физическую и логическую топологию. Логическая и физическая топологии сети независимы друг от друга. Физическая топология – это геометрия построения сети, а логическая топология определяет направления потоков данных между узлами сети и способы передачи данных.

В настоящее время в локальных сетях используются следующие топологии:

• физическая шина (Bus);
• физическая звезда (Star);
• физическое кольцо (Ring);
• физическая звезда и логическое кольцо (Token Ring).

Шинная топология

Сети с шинной топологией (рис. А.1) используют линейный моноканал (коаксиальный кабель) передачи данных, на концах которого устанавливаются оконечные сопротивления (терминаторы). Каждый компьютер подключается к коаксиальному кабелю с помощью Т-

разъема (Т-коннектора). Данные от передающего узла сети передаются по шине в обе стороны, отражаясь от оконечных терминаторов. Терминаторы предотвращают отражение сигналов, т. е. используются для гашения сигналов, которые достигают концов канала передачи данных.

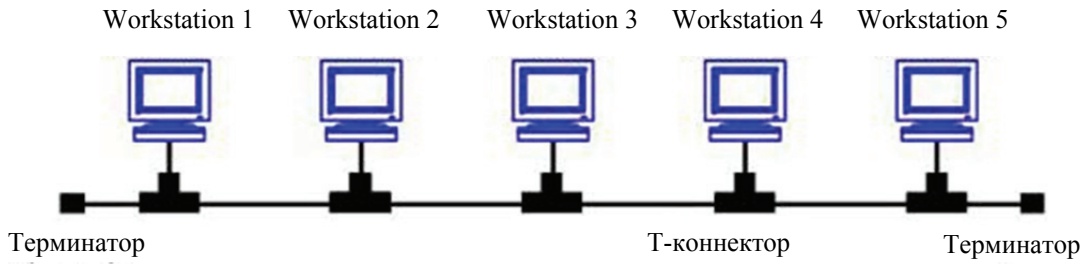


Рис. А.1. Сеть с шинной топологией

Таким образом информация поступает на все узлы, но принимается только тем узлом, которому она предназначается. В топологии типа «логическая шина» среда передачи данных используется совместно и одновременно всеми ПК сети, а сигналы от ПК распространяются одновременно во все направления по среде передачи. Так как передача сигналов в топологии типа «физическая шина» является широковещательной, т. е. сигналы распространяются одновременно во все направления, то логическая топология данной локальной сети является логической шиной.

Данная топология применяется в локальных сетях с архитектурой Ethernet (классы 10Base-5 и 10Base-2 для толстого и тонкого коаксиального кабеля соответственно).

Преимущества сетей шинной топологии:

- отказ одного из узлов не влияет на работу сети в целом;
- сеть легко настраивать и конфигурировать;
- сеть устойчива к неисправностям отдельных узлов.

Недостатки сетей шинной топологии:

а) разрыв кабеля может повлиять на работу всей сети;

б) ограниченная длина кабеля и количество рабочих станций;

в) трудно определить дефекты соединений.

Топология «звезда»

В сети, построенной по топологии типа «звезда» (рис. А.2), каждая рабочая станция подсоединяется кабелем (витой парой) к концентратору, или хабу (*hub*). Концентратор обеспечивает параллельное соединение ПК и таким образом все компьютеры, подключенные к сети, могут общаться друг с другом.

Данные от передающей станции сети передаются через хаб по всем линиям связи всем ПК. Информация поступает на все рабочие

станции, но принимается только теми станциями, которым она предназначается. Так как передача сигналов в топологии «физическая звезда» является широковещательной, т. е. сигналы от ПК распространяются одновременно по всем направлениям, то логическая топология данной локальной сети является логической шиной.

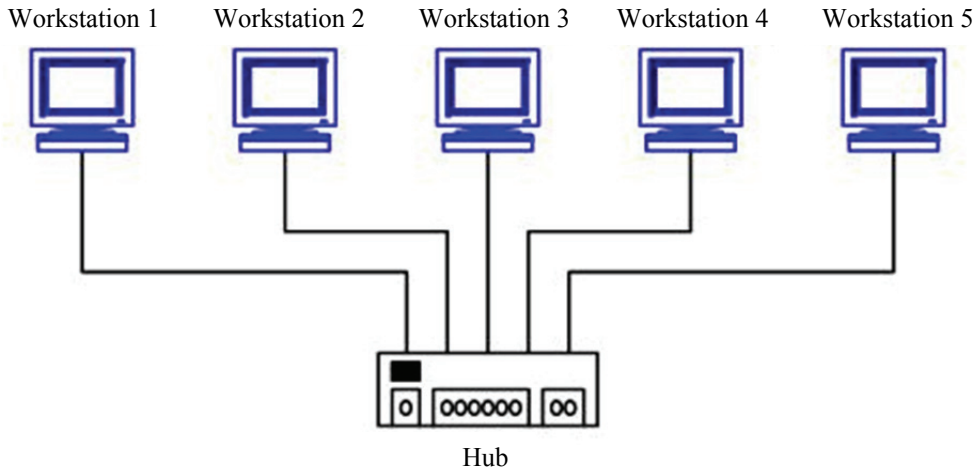


Рис. А.2. Сеть по топологии «звезда»

Данная топология применяется в локальных сетях с архитектурой 10Base-T Ethernet.

Преимущества сетей топологии «звезда»:

- легко подключить новый ПК;
- имеется возможность централизованного управления;
- сеть устойчива к неисправностям отдельных ПК и к разрывам соединения отдельных ПК.

Недостатки сетей топологии «звезда»:

а) отказ хаба влияет на работу всей сети;

б) большой расход кабеля.

Топология «кольцо»

В сети с топологией «кольцо» (рис. А.3) все узлы соединены каналами связи в неразрывное кольцо (необязательно окружность), по которому передаются данные. Выход одного ПК соединяется со входом другого ПК. Начав движение из одной точки, данные в конечном счете попадают на его начало. Данные в кольце всегда движутся в одном и том же направлении.

Принимающая рабочая станция распознает и получает только адресованное ей сообщение.

В сети с топологией типа «физическое кольцо» используется маркерный доступ, который предоставляет станции право на использование

кольца в определенном порядке. Логическая топология данной сети – логическое кольцо. Данную сеть очень легко создавать и настраивать.

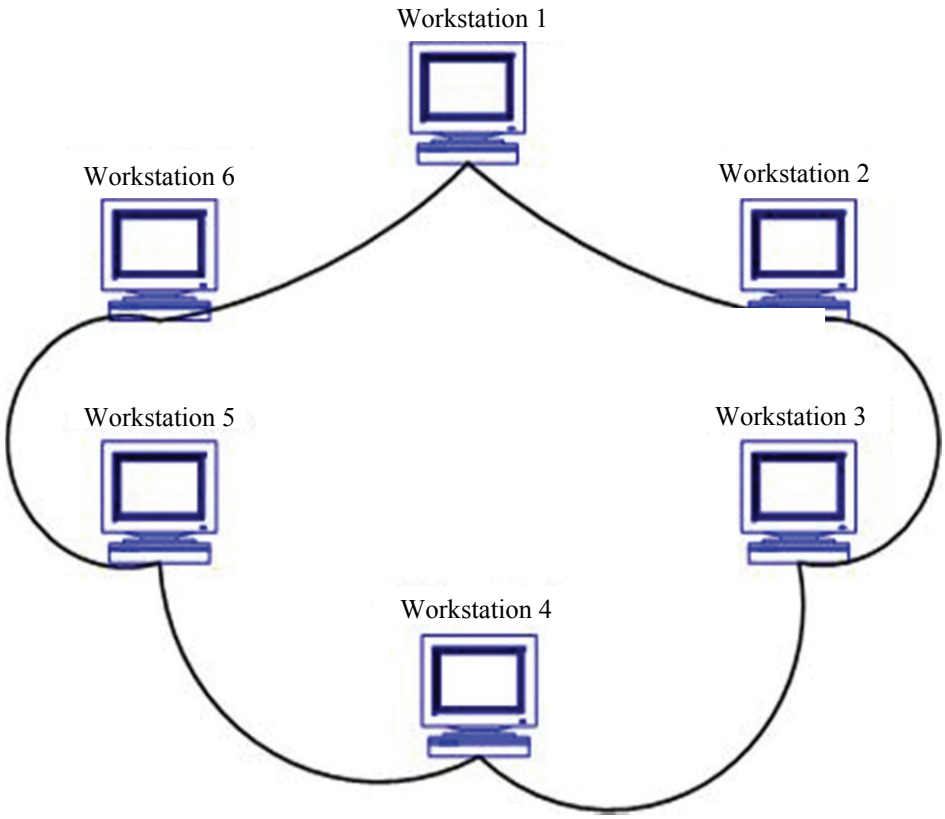


Рис. А.3. Сеть по топологии «кольцо»

Основным недостатком сетей топологии «кольцо» является то, что повреждение линии связи в одном месте или отказ ПК приводит к неработоспособности всей сети.

Как правило, в чистом виде топология «кольцо» не применяется из-за своей ненадежности, поэтому на практике используются различные модификации кольцевой топологии.

Топология «физическая звезда и логическое кольцо» (Token Ring)

Эта топология (рис. А.4) основана на топологии «физическое кольцо» с подключением типа «звезда». В данной топологии все рабочие станции подключаются к центральному концентратору (Token Ring), как в топологии «физическая звезда». Центральный концентратор – это интеллектуальное устройство, которое с помощью перемычек обеспечивает последовательное соединение выхода одной станции со входом другой станции.

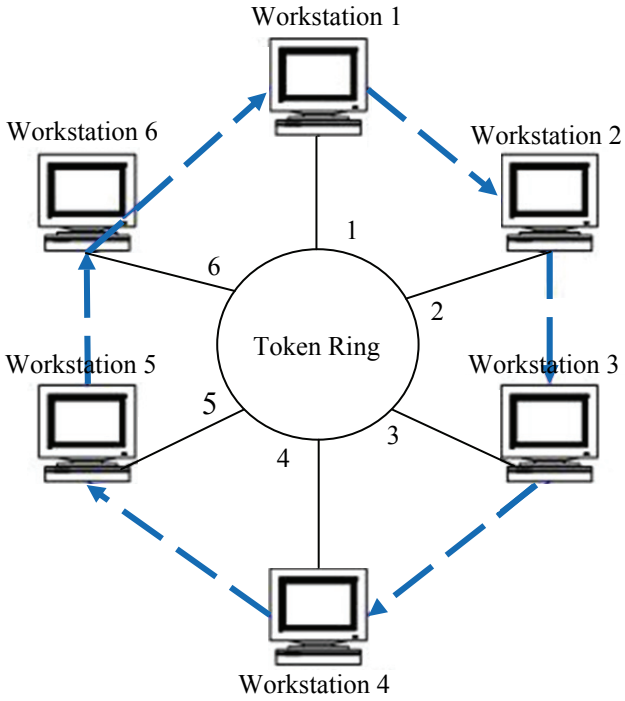


Рис. А.4. Сеть топологии Token Ring

Другими словами, с помощью концентратора каждая станция соединяется только с двумя другими станциями (предыдущей и последующей). Таким образом, рабочие станции связаны петлей кабеля, по которой пакеты данных передаются от одной станции к другой и каждая станция ретранслирует эти посланные пакеты. В каждой рабочей станции имеется для этого приемо-передающее устройство, которое позволяет управлять прохождением данных в сети. Физически такая сеть построена по типу топологии «звезда».

Концентратор создает первичное (основное) и резервное кольца. Если в основном кольце произойдет обрыв, то его можно обойти, воспользовавшись резервным кольцом, так как используется четырехжильный кабель. Отказ станции или обрыв линии связи рабочей станции не влечет за собой отказ сети как в топологии «кольцо», потому что концентратор отключет неисправную станцию и замкнет кольцо передачи данных.

В архитектуре Token Ring маркер передается от узла к узлу по логическому кольцу, созданному центральным концентратором. Такая маркерная передача осуществляется в фиксированном направлении (направление движения маркера и пакетов данных представлено на рис. А.4 стрелками синего цвета). Станция, обладающая маркером, может отправить данные другой станции.

Для передачи данных рабочие станции должны сначала дождаться прихода свободного маркера. В маркере содержится адрес станции, пославшей этот маркер, а также адрес той станции, которой он предназначается. После этого отправитель передает маркер следующей в сети станции для того, чтобы и та могла отправить свои данные.

Один из узлов сети (обычно для этого используется файл-сервер) создает маркер, который отправляется в кольцо сети. Такой узел выступает в качестве активного монитора, следящего за тем, чтобы маркер не был утерян или разрушен.

Intranet БГТУ

В УО «Белорусский государственный технологический университет» существует локальная компьютерная вычислительная сеть. Первоначально при создании ЛВС университета (1994 г.) преследовались две основные цели:

- сохранение студентом выполненной лабораторной работы без права несанкционированного доступа для ее изменения с дальнейшей ее защитой и защита файлов от компьютерных вирусов;
- независимость от рабочего места для дальнейшей работы со своей ранее созданной информацией.

В настоящее время ЛВС университета предусматривает к ранее реализованным задачам и решение следующих задач:

1) повышение продуктивности выполнения лабораторных работ студентами;

2) координация учебной и методической деятельности;

3) обеспечение эффективного использования программных и аппаратных средств;

4) автоматизации процесса контроля учебной деятельности;

5) возможность влиться в мировое информационное пространство;

6) повышение качества знаний студентов.

Каждый компьютерный класс университета (19 классов, 320 рабочих мест) имеет свою ЛВС, которая непосредственно может быть объединена с другим классом. Это позволяет студентам независимо на протяжении всего учебного процесса обучения использовать все свои разработки.

Компьютерная вычислительная сеть построена таким образом, что студент, зная доступ только к своей информации, не может без согласия преподавателя удалить ее. Кроме этого, у каждого преподавателя имеется отведенное дисковое пространство на сервере, прямой доступ к которому устанавливается администратором компьютерной сети

связанных учебных классов. Удобство использования ЛВС нашего университета заключается в том, что каждый студент, пропустивший занятия по каким-то причинам, может независимо от рабочего места в определенном учебном классе отработать лабораторную работу, предварительно согласовав задание с преподавателем, и соответствующим образом сохранить ее на отведенном дисковом пространстве сервера.

Сетевые компьютерные классы используются в университете на протяжении всего процесса обучения современным компьютерным технологиям и программным средствам, используемым в прикладных отраслях. Однако применение локальной сети при изучении дисциплин «Информатика», «Информатика и компьютерная графика», «Компьютерные информационные технологии», «Информационные технологии» студентами первых и вторых курсов университета является наиболее актуальным.

Это обусловлено тем, что многие лабораторные работы по одной теме студенты выполняют в несколько этапов и они рассчитаны не на одно учебное занятие. Это такие темы, как «Текстовый редактор Word», «Электронные таблицы Excel», «СУБД Access», «Создание Web-документов». Так, при изучении темы «СУБД Access» студенты должны разработать базу данных своей предметной области в несколько этапов (рис. А.5).

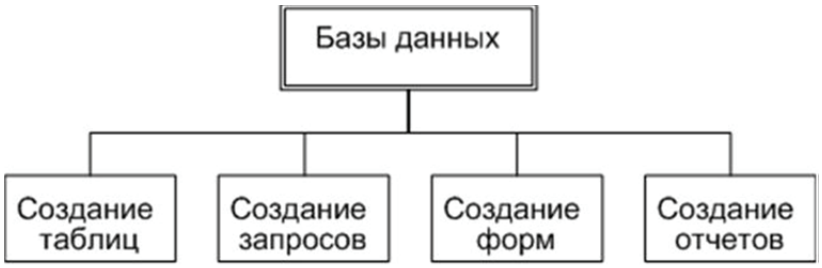


Рис. А.5. Разделы изучения студентами темы «СУБД Access»

На первом этапе (первая лабораторная работа по теме) студент должен разработать структуру своей базы данных, состоящей из взаимосвязанных таблиц. Затем, используя заполненные таблицы, применять ее для изучения следующих разделов:

- создание запросов (4 ч);
- создание форм (2 ч);
- создание отчетов (2 ч).

При использовании ЛВС университета проблема получения итогового результата поставленной десятичасовой лабораторной работы решается весьма успешно.

Для эффективного усвоения материала необходимо сначала внимательно проанализировать (возможно, и не один раз!) предыдущие результаты своих лабораторных работ, осмыслить и запомнить. Затем таким же образом воспользоваться рекомендациями и последовательно выполнить новое задание на компьютере университета, используя предыдущие свои разработки, сохраненные на соответствующем сервере учебного класса. Как правило, учебные занятия студентов разных факультетов (университет располагает семью факультетами) распределяются в соответствии с используемым математическим обеспечением и с применяемыми аппаратными средствами.

Помимо лабораторных работ студенты по дисциплине «Информатика» выполняют еще и курсовые работы. Курсовая работа обобщает полученные студентами теоретические знания и способствует применению их к решению конкретной инженерной задачи. При этом студент должен использовать полученные ранее знания в области программирования, а также использовать знание современных информационных технологий.

Курсовая работа является самостоятельной творческой работой студента, в которой он решает комплексную задачу в области использования современных аппаратных средств и программного обеспечения. При выполнении данной работы необходимо не только затратить большое количество времени, но и хранить большой объем информации, требуемый для выполнения курсовой работы. Кроме этого, студенту необходимо как можно более полно и достоверно использовать свои предыдущие разработки. Все это и позволяет сделать ЛВС университета.

Проблема поиска информации в наше время является одной из наиболее актуальных и часто решаемых при создании и реализации абсолютно любых проектов. Каждый студент регулярно сталкивается с необходимостью получения новых знаний, последней информации о той или иной научной разработке, новом способе решения каких-то старых задач и т. д. Способов пополнить свои знания и получить необходимую информацию множество: можно позвонить другу, сходить в библиотеку и др. Сегодня ко всем этим способам получения новых знаний присоединилась и компьютерная сеть.

Использование ЛВС играет огромную роль и при контроле знаний студентов. Преподаватель имеет возможность во время экзамена более полно и качественно оценить знания студента. Просмотрев любой раздел лабораторной или курсовой работы, преподаватель, как правило, имеет достоверную информацию о проделанной работе экзаменуемого студента и может правильно ее оценить.

Приложение Б

ОБЗОР ТЕКСТОВЫХ РЕДАКТОРОВ ДЛЯ КОМПЬЮТЕРА

Одной из самых востребованных программ, установленных на компьютере, является текстовый редактор независимо от установленной операционной системы. Это программа, которая позволяет производить набор текста, редактировать его, а также визуально оформлять. Рассмотрим наиболее популярные текстовые редакторы в порядке функциональности и популярности.

Блокнот (рис. Б.1). Это самый простой текстовый редактор в Windows. Он используется для создания и редактирования каких-либо заметок, небольших фраз и прочих пометок. Также в редакторе **Блокнот** удобно копировать пароли, ссылки и консольные команды. Он поставляется наряду со стандартным предустановленным пакетом программ операционной системы.

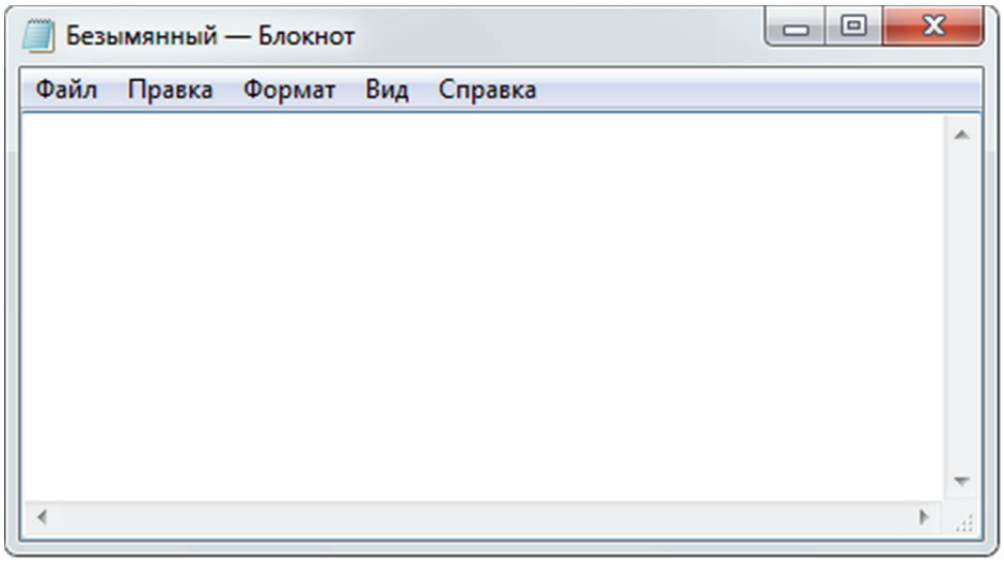


Рис. Б.1. Редактор **Блокнот**

В операционной системе Linux есть свой аналог блокнота – gedit. Программа по функциям и назначению абсолютно идентична с аналогом из Windows.

Преимущество данных текстовых редакторов в том, что они просты и компактны, это идеальный вариант для пометок и хранения

элементов кода. Недостатком данного редактора текста является его преимущество – излишняя простата, которая не дает возможность производить оформление теста.

Notepad++ (рис. Б.2). Когда возможностей, предоставляемых **Блокнотом**, становится мало, понадобится его расширенная версия – **Notepad++**, которая имеет большое количество функций, но при этом остается все тем же **Блокнотом.** Такая программа подойдет программистам, потому что умеет выделять цветом конструкции различных языков программирования. Загрузить **Блокнот** можно с официального сайта программы Notepad-plus-plus.org. Программа имеет русский интерфейс и распространяется бесплатно.

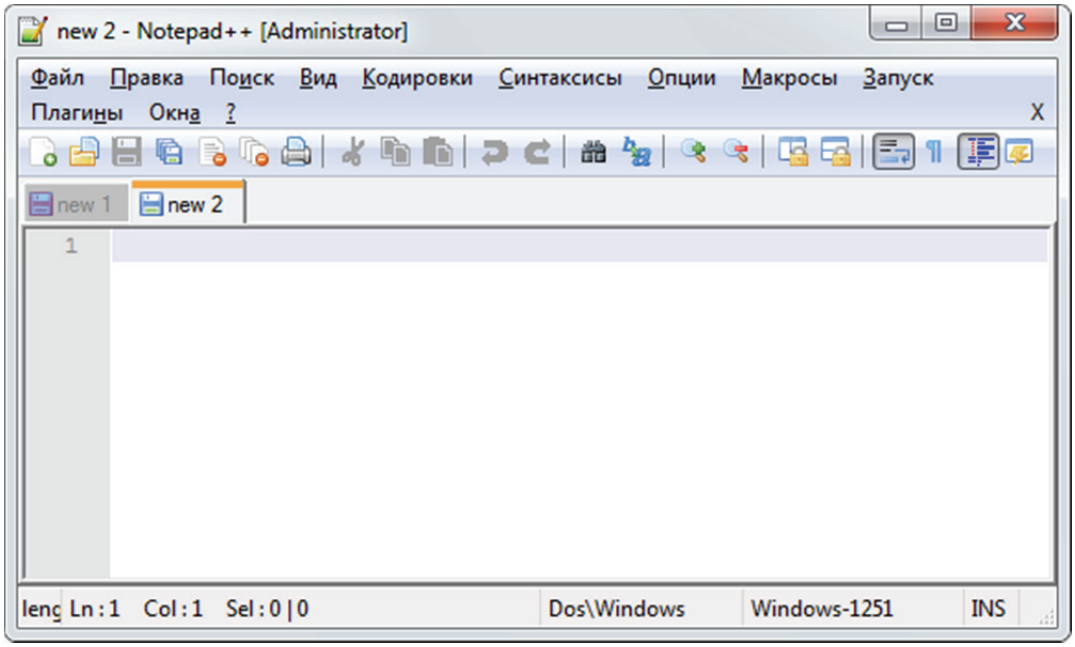


Рис. Б.2. Редактор **Notepad++**

WordPad (рис. Б.3). В отличие от ранее рассмотренных текстовых редакторов этот редактор позволяет выполнять несложное визуальное форматирование текста. Он обладает базовыми возможностями проверки орфографии, чего лишены рассмотренные ранее программы и он бесплатен.

LibreOffice Writer (рис. Б.4). Изначально данный текстовый редактор был создан для операционной системы Linux, сменив устоявшийся на то время OpenOfficeOrg, который существует и поныне. Затем была выпущена версия LibreOffice Writer под Windows.

По интерфейсу данный текстовый редактор напоминает Microsoft Word 2003. Редактор представляет функционал, аналогичный функционалу своего более знаменитого аналога, при этом обладает некоторыми

дополнительными возможностями (например, вычисления в тексте). Редактор постоянно развивается и в отличие от Microsoft Word является бесплатным.

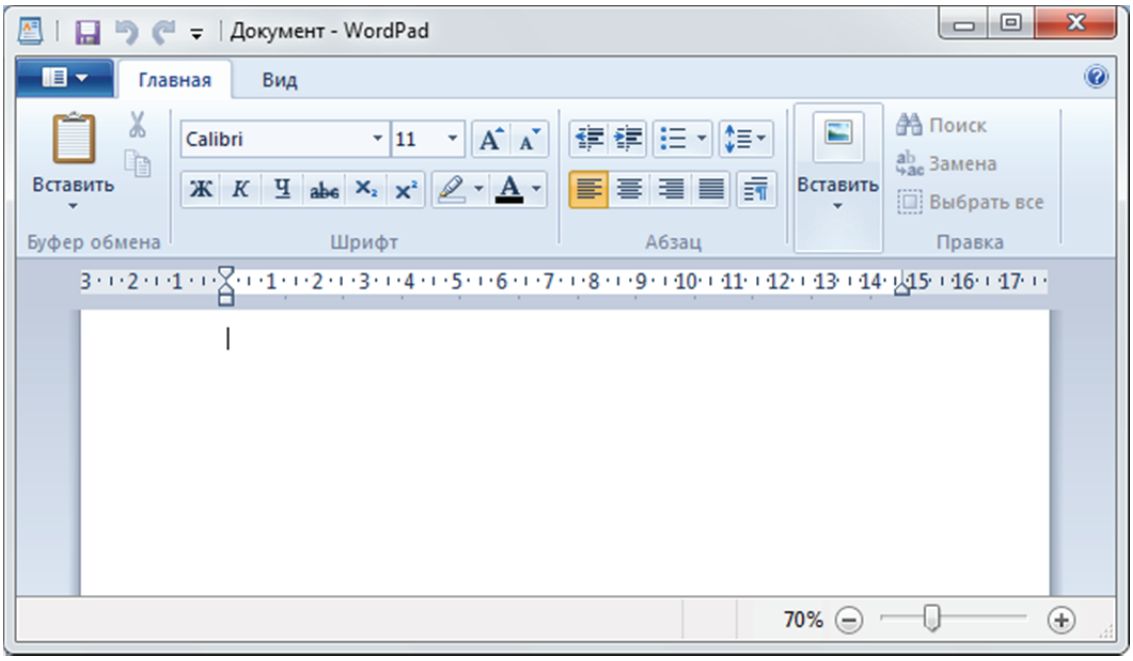


Рис. Б.3. Редактор **WordPad**

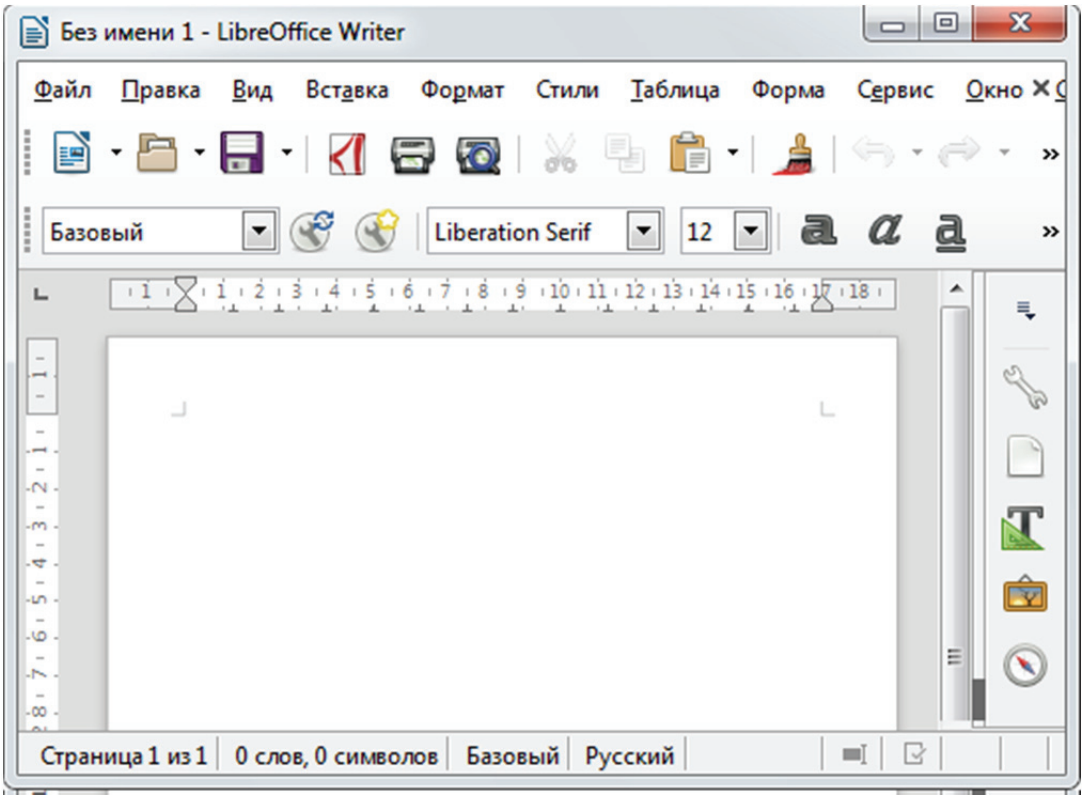


Рис. Б.4. Редактор **LibreOffice Writer**

Microsoft Word (рис. Б.5). **Word** – это незаменимый редактор текстов для любого пользователя, которому необходимо часто заниматься набором текстов и документов. Он имеет хорошее качество проверки орфографии в текстах, чем не может похвастаться ни один текстовый редактор для компьютера. Из всех рассмотренных программ Word является единственной программой, за которую необходимо заплатить.

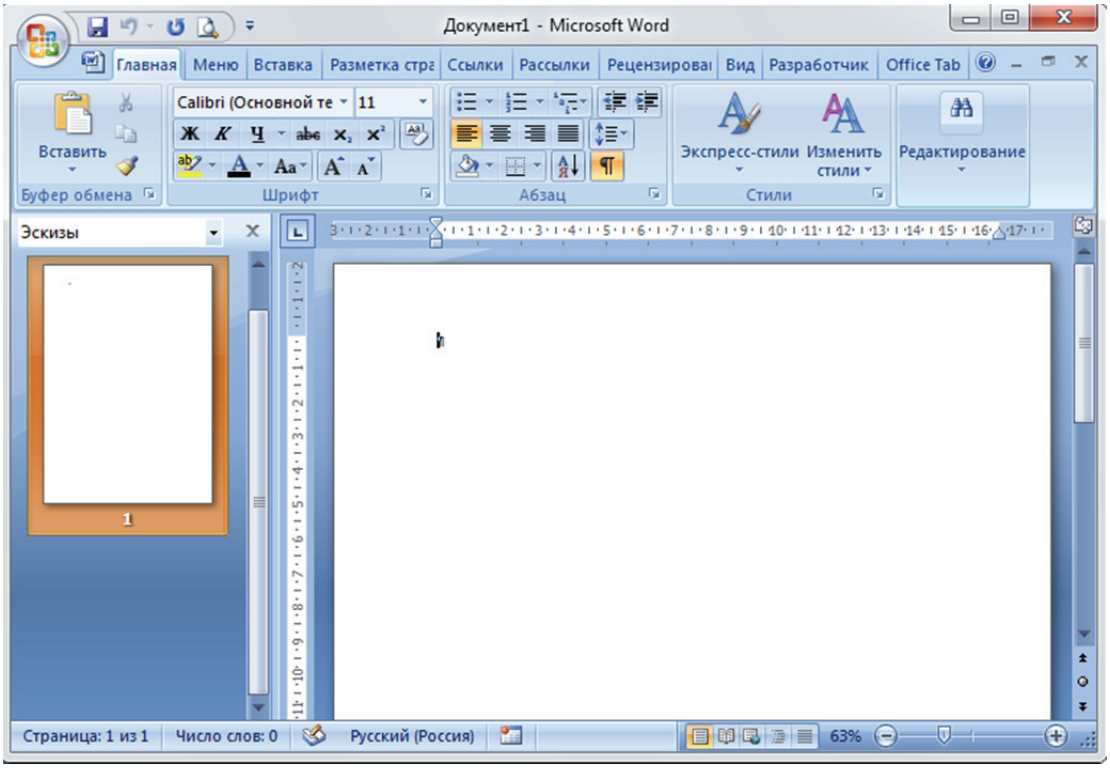


Рис. Б.5. Редактор **Microsoft Word**

В последние годы текстовые редакторы перебрались в сеть интернет и превратились в сервисы, доступ к которым осуществляется через браузер. Это текстовые редакторы онлайн. Данный вид редакторов текста позволяет набирать их непосредственно на удаленном сервере в интернете – онлайн. Поэтому набранный текст сразу же сохраняется в облачном хранилище и, как следствие, доступ к тексту можно открыть сразу же нескольким пользователям (и даже редактировать вместе), а также самому иметь доступ к данным документам со всех своих устройств, которые подключены к интернету. Благодаря этому не страшна потеря данных или отключение света – документ всегда доступен в интернете, до тех пор пока серверы будут подключены к сети интернет.

Наилучшим онлайн редактором текста является сервис **Google Docs** (рис. Б.6).

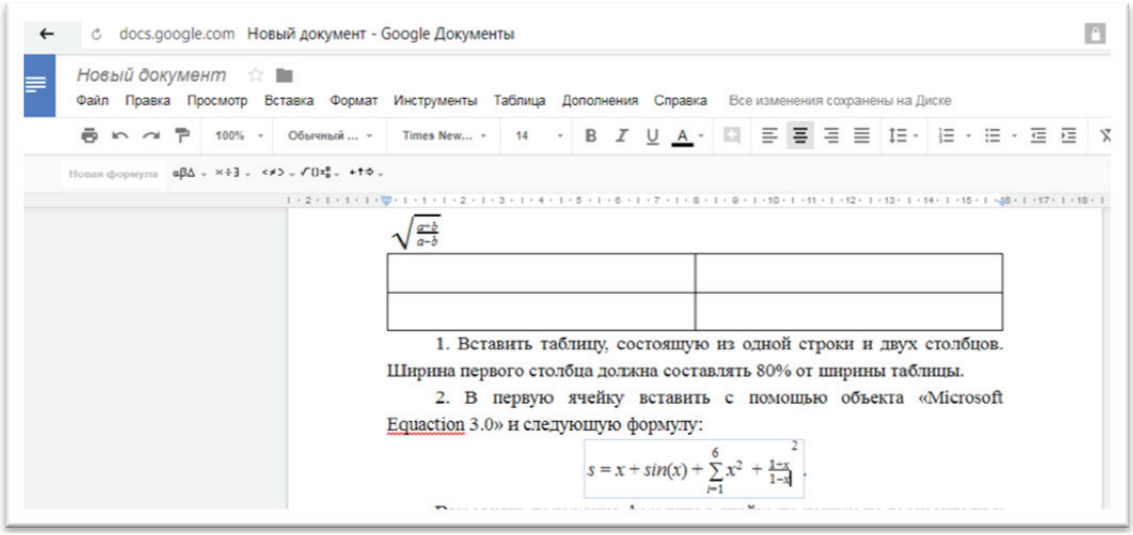


Рис. Б.6. Редактор **Google Docs**

Зарегистрировав себе облачный диск, вы сможете создавать на нем текстовые документы. Дизайн таких документов особой красотой не отличается, но в ней нет необходимости, так как это лишняя нагрузка на интернет-канал, самое главное – это набор текста. Проверка орфографии производится браузером, это неидеальный вариант, но и не самый плохой. К преимуществам можно отнести и бесплатность сервиса, что немаловажно. Документ можно сразу же сохранить на компьютере, если в этом будет необходимость.

Приложение В

РАБОТА С ЭЛЕКТРОННЫМИ ТАБЛИЦАМИ MICROSOFT EXCEL. ОБЗОР ЭЛЕКТРОННЫХ ТАБЛИЦ

Электронная таблица (табличный процессор) – компьютерная программа, позволяющая проводить вычисления с данными, представленными в виде двумерных массивов. Электронные таблицы представляют собой удобный инструмент для автоматизации вычислений. Использование математических формул в электронных таблицах позволяет представить взаимосвязь между различными параметрами некоторой реальной системы. Решения многих вычислительных задач, которые раньше можно было осуществить только с помощью программирования, стало возможно реализовать через математическое моделирование в электронной таблице. Инструментарий электронных таблиц включает мощные математические функции, позволяющие вести сложные статистические, финансовые и прочие расчеты.

Функции табличных процессоров весьма разнообразны:

- создание и редактирование электронных таблиц;
- создание многотабличных документов;
- оформление и печать электронных таблиц;
- построение диаграмм, их модификация и решение экономических задач графическими методами;
- создание многотабличных документов, объединенных формулами;
- работа с электронными таблицами как с базами данных: сортировка таблиц, выборка данных по запросам;
- создание итоговых и сводных таблиц;
- использование при построении таблиц информации из внешних баз данных;
- создание слайд-шоу;
- решение оптимизационных задач;
- решение экономических задач типа «что – если» путем подбора параметров;
- разработка макрокоманд, настройка среды под потребности пользователя и т. д.

Наиболее популярными платными программами для работы с электронными таблицами для персональных компьютеров являются

табличные процессоры **Microsoft Excel** из пакета Microsoft Office и **Corel Quattro Pro X8** из пакета Corel WordPerfect Office. Существуют бесплатные программы для работы с электронными таблицами. Это бесплатные процессоры, входящие в состав открытых офисных пакетов **LibreOffice Calc** из пакета LibreOffice и **Apache OpenOffice.org Calc** из пакета OpenOffice. Можно использовать бесплатные клоны Microsoft Office, довольно близкие к оригиналу – WPS Office и SoftMaker FreeOffice, которые содержат в себе табличный процессор. Все бесплатные процессоры поддерживают соответствующие форматы Microsoft.

Набирают популярность веб-версии офисных пакетов. От компании Microsoft – **Microsoft Office Online**. Полный офисный пакет Microsoft доступен в рамках платной подписки. Но веб-версии некоторых входящих в него программ, в том числе и **Excel**, можно использовать бесплатно. Онлайн-варианты веб-версии программ имеют почти такой же дизайн, как их настольные аналоги, и нет никаких проблем с традиционными офисными форматами Microsoft. Вместе с тем веб-приложения не поддерживают редактирование в режиме офлайн и лишены многих продвинутых возможностей вроде сводных таблиц, редактора баз данных и поддержки rtf, html и некоторых других форматов. От компании Google – пакет **Google Документы**, который включает сервис Google Таблицы. Набор офисных веб-приложений от Google идеально подходит для людей, работающих в командах. Проект разработан с прицелом на совместное редактирование документов в режиме реального времени, но для одиночного пользователя он также вполне подходит. Благодаря полной кросс-платформенности для работы можно использовать любые устройства. В отличие от Microsoft Office Online, документы можно редактировать в режиме офлайн, но для этого нужно скачать расширение для Chrome. Приложения Google удобно интегрируются с другими сервисами компании вроде «Google Диска», «Календаря» и Gmail. Веб-приложения от Google поддерживают форматы Microsoft. От компании Apple – пакет **Apple iWork**, который включает сервис Numbers, позволяющий работать с таблицами. Если раньше вы использовали Microsoft Office, приложения в составе iWork покажутся вам непривычными и придется адаптироваться к новому интерфейсу. Зато стандартные для Microsoft форматы документов будут полностью рабочими в iWork. Офисные приложения Apple поддерживают совместное редактирование файлов онлайн и доступны на macOS и iOS. Пользователи Windows тоже могут работать с iWork, но только с помощью браузера на сайте iCloud.

По сути своей, лист Excel – это и есть таблица, состоящая из ячеек. Для создания таблицы требуемого вида необходимо отформатировать нужный фрагмент таблицы. Следует активизировать нужные ячейки (выделить их левой кнопкой мыши при удержании первой ячейки) и применить к ним команды форматирования границ ячейки и их содержимого.

Для форматирования таблиц необходимо использовать команды, которые находятся в главном меню на вкладке **Главная**.

1. **Изменение ширины и высоты ячеек.** Для изменения ширины и высоты ячеек проще всего использовать свойства заголовков столбцов и строк таблицы. Чтобы измененить ширину ячеек, поставьте курсор на границу между ними в заголовке столбца и, удерживая левую кнопку мыши, потяните в сторону, после чего отпустите (рис. В.1). Аналогично изменяется высота ячеек. Для того чтобы применить автовыравнивание ширины ячеек по содержимому, необходимо поместить указатель в область заголовков на правую границу каждого столбца и дважды щелкнуть по ней.

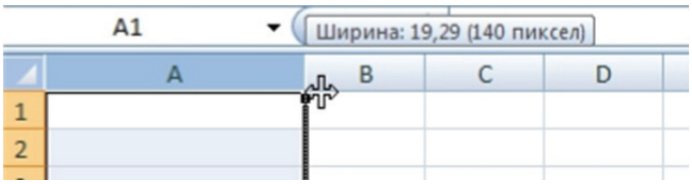


Рис. В.1. Изменение ширины ячеек

Для того чтобы задать размер сразу нескольким столбцам или строкам, активизируйте нужные строки/столбцы, выделив их по серому полю, и произведите описанную выше операцию над любой из выделенных ячеек серого поля.

2. **Объединение ячеек.** Для объединения ячеек воспользуйтесь кнопкой **Объединить ячейки** из группы команд **Выравнивание** (рис. В.2). Объединяемые ячейки предварительно выделите. Здесь же можно отменить объединение.

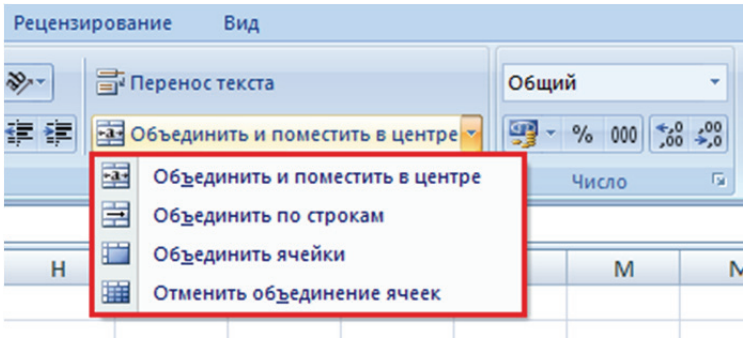


Рис. В.2. Объединение ячеек

3. **Форматирование границ таблицы.** Форматирование границ таблицы можно осуществить, используя меню кнопки **Границы** из группы команд **Шрифт** (рис. В.3).

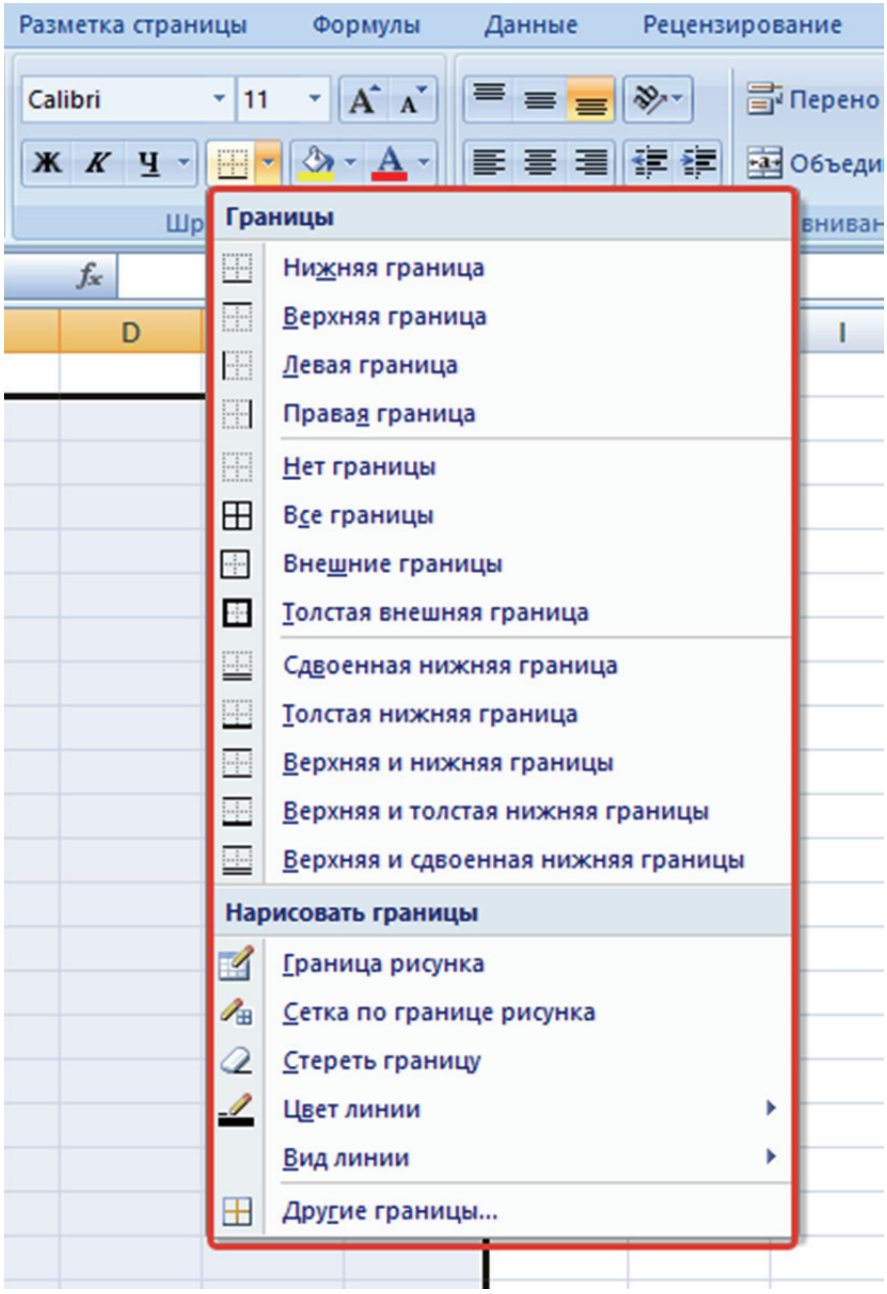


Рис. В.3. Форматирование границ таблицы

Также можно воспользоваться окном диалога **Формат ячеек**, которое вызывается двумя способами: правой клавишей мыши, вызывающей контекстное меню (рис. В.4), или в главном меню на вкладке **Главная** нажатием на кнопку в правом нижнем углу группы команд **Шрифт**, **Выравнивание** или **Число** (рис. В.5).

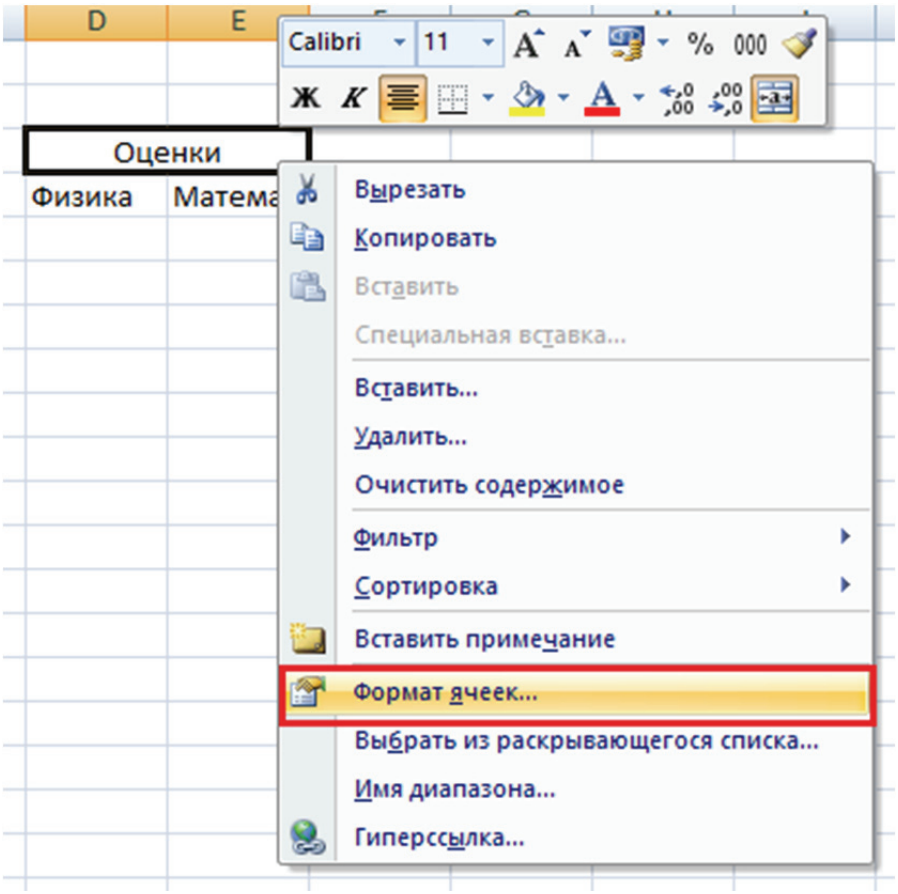


Рис. В.4. Контекстное меню

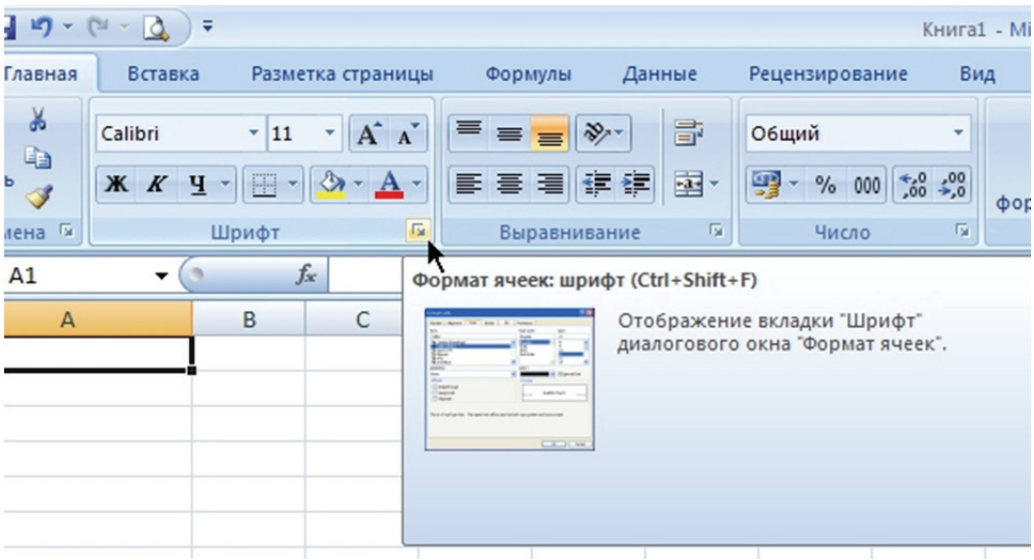


Рис. В.5. Команда **Шрифт**

В окне диалога **Формат ячеек** выберите вкладку **Граница** (рис. В.6). Укажите нужные границы таблицы левой кнопкой мыши и задайте им необходимый тип линии.

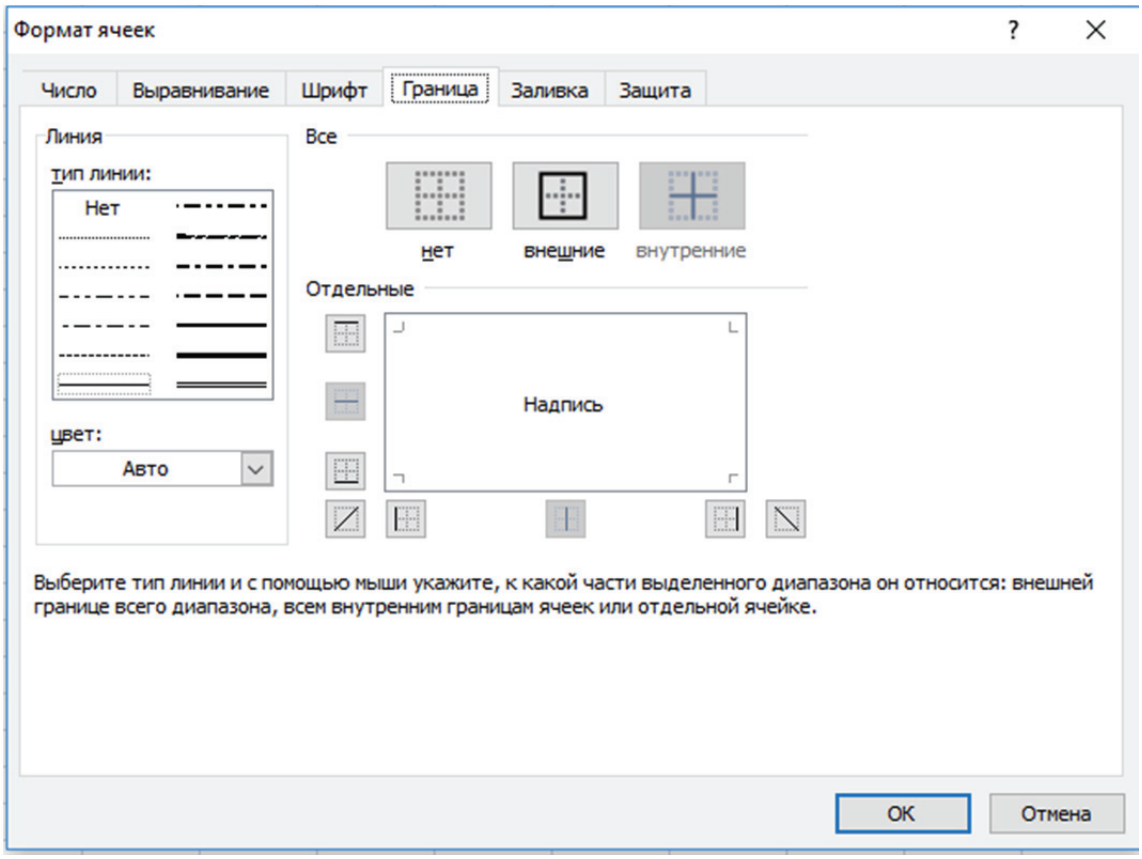


Рис. В.6. Окно **Формат ячеек**

4. **Форматирование содержимого ячеек.** Для форматирования содержимого ячеек необходимо воспользоваться группой команд **Шрифт** (рис. В.7). В группе находятся команды по *выбору шрифта*, изменению *размера* шрифта, выбору *начертания* (*полужирный*, *курсивный*, *подчеркнутый*) и *цвета заливки*, *цвета текста*.

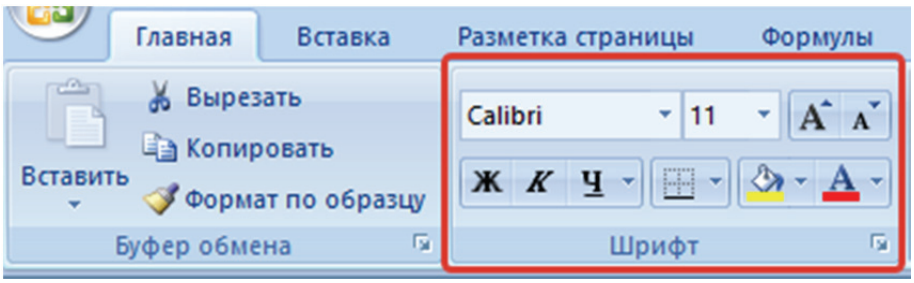


Рис. В.7. Группа **Шрифт**

Для переноса текста необходимо воспользоваться кнопкой **Перенести текст** из группы команд **Выравнивание** (рис. В.8) либо в окне диалога **Формат ячеек** на вкладке **Выравнивание** (рис. В.9) поставить галочку «переносить по словам». Здесь же находятся команды по выравниванию содержимого ячеек относительно их границ.

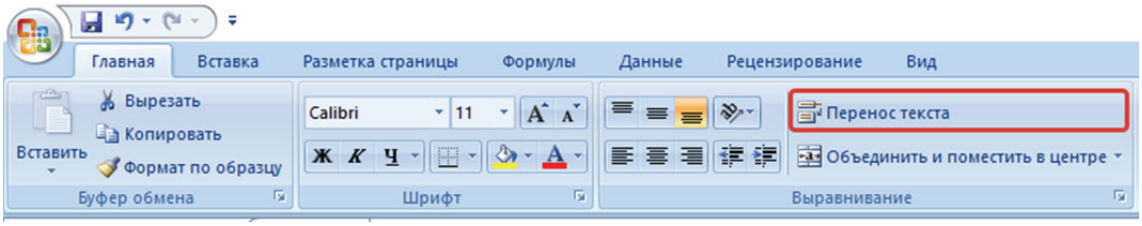


Рис. В.8. Команда **Перенести текст**

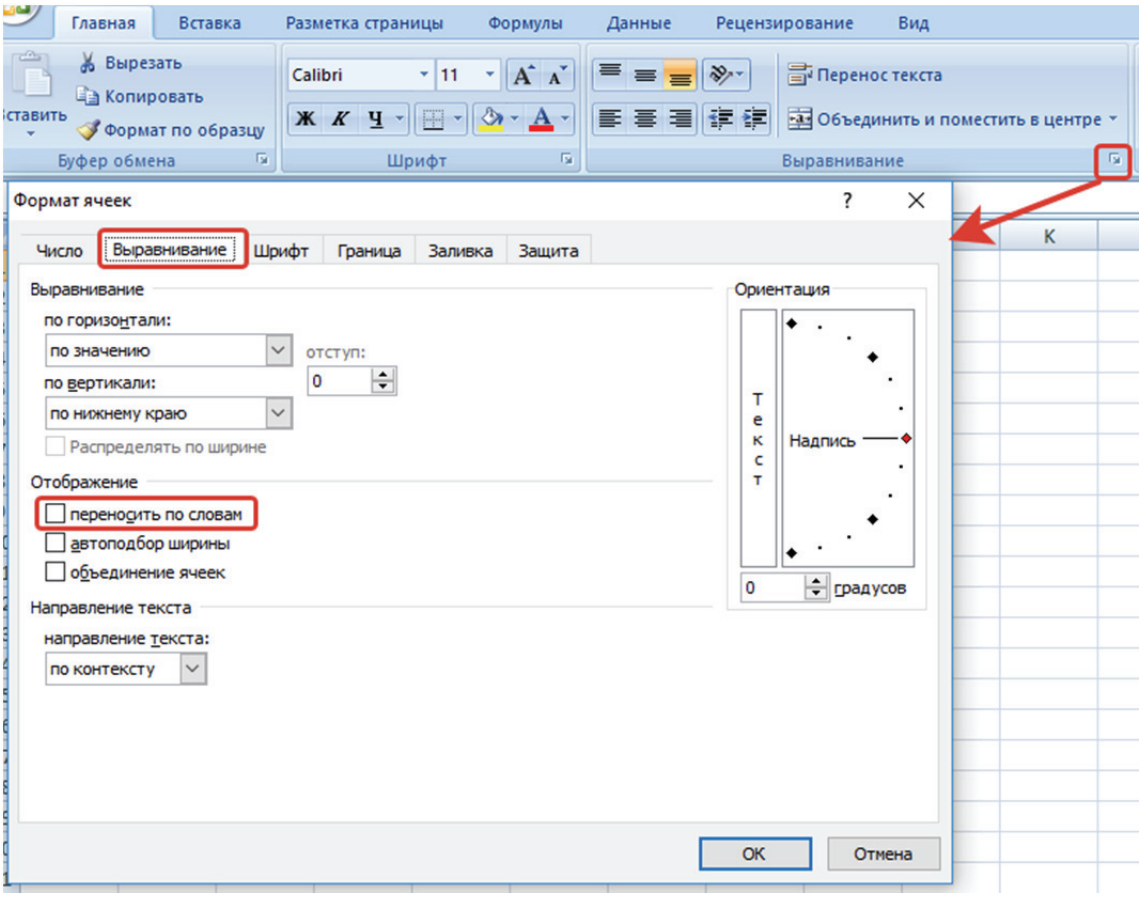


Рис. В.9. Окно **Формат ячеек** – вкладка **Выравнивание**

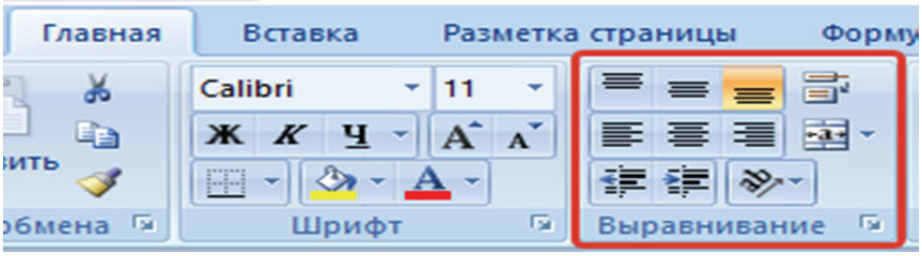


Рис. В.10. Группа команд **Выравнивание**

Выравнивание содержимого ячеек относительно их границ также доступно с помощью кнопок в группе команд **Выравнивание** (рис. В.10).

Приложение Г

МАТЕМАТИЧЕСКИЕ ПАКЕТЫ

Серьезное конструирование, градостроительство и архитектура, электротехника и масса смежных с ними отраслей, а также учебные заведения технической направленности уже не могут обойтись без систем автоматизированного проектирования (САПР). Математические пакеты являются составной частью таких систем, поскольку некоторые задачи вообще невозможно решить без помощи компьютера. В настоящее время наиболее известными считаются **Maple**, **MathCad**, **Mathematica** и **MatLab**.

Спектр задач, решаемых подобными системами, очень широк:

- проведение математических исследований, требующих вычислений и аналитических выкладок;
- разработка и анализ алгоритмов;
- математическое моделирование и компьютерный эксперимент;
- анализ и обработка данных;
- визуализация, научная и инженерная графика;
- разработка графических и расчетных приложений.

Mathematica. Пакет Mathematica повсеместно применяется при расчетах в современных научных исследованиях и получил широкую известность в научной и образовательной среде. Несмотря на свою направленность на серьезные математические вычисления, системы класса Mathematica просты в освоении и могут использоваться довольно широкой категорией пользователей. При этом широчайшие функции программы не перегружают ее интерфейс и не замедляют вычислений. Mathematica неизменно демонстрирует высокую скорость символьных преобразований и численных расчетов. Программа Mathematica из всех рассматриваемых систем наиболее полна и универсальна.

Система Mathematica сегодня рассматривается как мировой лидер среди компьютерных систем символьной математики для ПК, обеспечивающих не только возможность выполнения сложных численных расчетов с выводом их результатов в самом лучшем графическом виде, но и проведение особо трудоемких аналитических преобразований и вычислений. Версии системы под Windows имеют современный пользовательский интерфейс и позволяют готовить документы в форме Notebooks (записных книжек). Они объединяют исходные данные,

описания алгоритмов решения задач, программ и результатов решения в самой разнообразной форме (математические формулы, числа, векторы, матрицы, таблицы и графики).

Графика всегда была сильной стороной различных версий системы Mathematica и обеспечивала им лидерство среди систем компьютерной математики.

Центральное место в системах класса Mathematica занимает машинно-независимое ядро математических операций, которое позволяет переносить систему на различные компьютерные платформы. Для расширения набора функций служат библиотека (Library) и набор пакетов расширения (Add-on Packages). Пакеты расширений готовятся на собственном языке программирования систем Mathematica и являются главным средством для развития возможностей системы и их адаптации к решению конкретных классов задач пользователя. Кроме того, системы имеют встроенную электронную справочную систему – Help, которая содержит электронные книги с реальными примерами.

В качестве более простых, но идеологически близких и бесплатных альтернатив программы Mathematica можно назвать такие бесплатные пакеты, как Maxima и Kalamaris.

Maple. Программа Maple является одним из лидеров среди универсальных систем символьных вычислений. Она предоставляет пользователю удобную интеллектуальную среду для математических исследований любого уровня и пользуется особой популярностью в научной среде. Отметим, что символьный анализатор программы Maple является наиболее сильной частью этого ПО, поэтому именно он был позаимствован и включен в ряд других пакетов, таких как MathCad и MatLab.

Maple предоставляет удобную среду для компьютерных экспериментов, в ходе которых пробуются различные подходы к задаче, анализируются частные решения, а при необходимости программирования отбираются требующие особой скорости фрагменты. Пакет позволяет создавать интегрированные среды с участием других систем и универсальных языков программирования высокого уровня. Когда расчеты произведены и требуется оформить результаты, то можно использовать средства этого пакета для визуализации данных и подготовки иллюстраций для публикации.

Пакет Maple состоит из ядра (процедур, написанных на языке C и хорошо оптимизированных), библиотеки, написанной на Maple-языке, и развитого внешнего интерфейса. Ядро выполняет большинство базовых операций, а библиотека содержит множество команд – процедур, выполняемых в режиме интерпретации.

Систему Maple можно использовать и на самом элементарном уровне ее возможностей – как очень мощный калькулятор для вычислений по заданным формулам, но главным ее достоинством является способность выполнять арифметические действия в символьном виде, т. е. так, как это делает человек. Таким образом, Maple – это, пожалуй, наиболее удачно сбалансированная система и бесспорный лидер по возможностям символьных вычислений для математики. При этом оригинальный символьный движок сочетается здесь с легко запоминающимся структурным языком программирования, так что Maple может быть использована как для небольших задач, так и для серьезных проектов. Одним из наиболее часто используемых в системе Maple пакетов программ является пакет линейной алгебры, содержащий мощный набор команд для работы с векторами и матрицами. Для технических применений в Maple включены справочники физических констант и единицы физических величин с автоматическим пересчетом формул.

К недостаткам системы Maple можно отнести лишь ее некоторую медлительность, причем не всегда обоснованную, а также очень высокую стоимость этой программы.

В качестве более простых, но идеологически близких альтернатив программе Maple можно отметить такие пакеты, как Derive, Scientific WorkPlace и YaCaS.

MatLab. MatLab – одна из старейших, тщательно проработанных и проверенных временем систем автоматизации математических расчетов, построенная на расширенном представлении и применении матричных операций. Это нашло отражение и в самом названии системы – MATrix LABoratory, т. е. матричная лаборатория. Однако синтаксис языка программирования системы продуман настолько тщательно, что данная ориентация почти не ощущается теми пользователями, которых не интересуют непосредственно матричные вычисления.

Несмотря на то что изначально MatLab предназначалась исключительно для вычислений, в дополнение к прекрасным вычислительным средствам было приобретено ядро символьных преобразований, а также появились библиотеки, которые обеспечивают в MatLab уникальные для математических пакетов функции. Например, широко известная библиотека Simulink, реализуя принцип визуального программирования, позволяет построить логическую схему сложной системы управления из одних только стандартных блоков, не написав при этом ни строчки кода. После конструирования такой схемы можно детально проанализировать ее работу. В системе MatLab также существуют широкие возможности для программирования.

Для визуализации моделирования система MatLab имеет библиотеку Image Processing Toolbox, которая обеспечивает широкий спектр функций, поддерживающих визуализацию проводимых вычислений непосредственно из среды MatLab.

Таким образом, систему MatLab можно использовать для обработки изображений, сконструировав собственные алгоритмы, которые будут работать с массивами графики как с матрицами данных. Поскольку язык MatLab оптимизирован для работы с матрицами, в результате обеспечивается простота применения, высокая скорость и экономичность проведения операций над изображениями.

Из недостатков системы MatLab можно отметить невысокую интегрированность с86реды (очень много окон, с которыми лучше работать на двух мониторах), не очень внятную справочную систему (а между тем объем фирменной документации достигает почти 5 тыс. страниц, что делает ее трудно обозримой) и специфический редактор кода MatLab-программ. Сегодня система MatLab широко используется в технике, науке и образовании, но все-таки она больше подходит для анализа данных и организации вычислений, нежели для чисто математических выкладок.

Поэтому для проведения аналитических преобразований в MatLab используется ядро символьных преобразований Maple, а из Maple для численных расчетов можно обращаться к MatLab.

В качестве более простых, но идеологически близких альтернатив программе MatLab можно отметить такие пакеты, как Octave, KOctave и Genius.

Mathcad. В отличие от мощного и ориентированного на высокоэффективные вычисления при анализе данных пакета MatLab, программа Mathcad – это, скорее, простой, но продвинутый редактор математических текстов с широкими возможностями символьных вычислений и прекрасным интерфейсом. Mathcad не имеет языка программирования как такового, а движок символьных вычислений заимствован из пакета Maple. Зато интерфейс программы Mathcad очень простой, а возможности визуализации богатые. Все вычисления здесь осуществляются на уровне визуальной записи выражений в общеупотребительной математической форме. Пакет имеет хорошие подсказки, подробную документацию, функцию обучения использованию, целый ряд дополнительных модулей и приличную техническую поддержку производителя. Однако пока математические возможности Mathcad в области компьютерной алгебры намного уступают системам Maple, Mathematica и MatLab. Однако по программе Mathcad

выпущено много книг и обучающих курсов. Сегодня эта система стала буквально международным стандартом для технических вычислений.

Для небольшого объема вычислений Mathcad идеален – здесь все можно проделать очень быстро и эффективно, а затем оформить работу в привычном виде (Mathcad предоставляет широкие возможности для оформления результатов, вплоть до публикации в интернете). Пакет имеет удобные возможности импорта/экспорта данных. Например, можно работать с электронными таблицами Microsoft Excel прямо внутри Mathcad-документа.

В целом, Mathcad – это очень простая и удобная программа, которую можно рекомендовать широкому кругу пользователей, в том числе не очень сведущих в математике, а особенно тем, кто только постигает ее азы.

В качестве более дешевых, простых, но идеологически близких альтернатив программе Mathcad можно отметить такие пакеты, как YaCaS, коммерческую систему MuPAD и бесплатную программу KmPlot.

Приложение Д

СРЕДА ПРОГРАММИРОВАНИЯ VISUAL BASIC

После запуска системы программирования Visual Basic на экране дисплея обычно появляется диалоговое окно **New Project** запроса на выбор типа проекта для разработки (рис. Д.1). Предлагаемый при запуске тип **Standard EXE** используется для создания стандартной программы для Windows (**exe**-файла) и выбирается чаще всего. Другие типы проектов необходимы для разработки более сложных программ.

Рис. Д.1. Диалоговое окно выбора типа проекта программы

Кроме того, в этом окне, кроме вкладки **New** для создания новых проектов, есть еще две вкладки: **Existing** для активизации окна с общим списком папок и хранящихся проектов на компьютере для их загрузки и дальнейшего редактирования; **Recent** для активизации списка проектов, которые разрабатывались в последнее время.

После выбора типа проекта на экране появляется среда разработки Visual Basic (рис. Д.2). Фактически эта среда представляет собой *редактор программ* на языке VB, в котором разрабатываются все *исходные компоненты* создаваемой программы. В верхней части экрана

размещается **Главное** окно, содержащее главное меню команд среды программирования VB. При закрытии главного окна среда VB выгружается из оперативной памяти и работа заканчивается. Остальные окна, включая и панель инструментов **Standard,** могут занимать любое место внутри главного окна и вообще не присутствовать на экране.

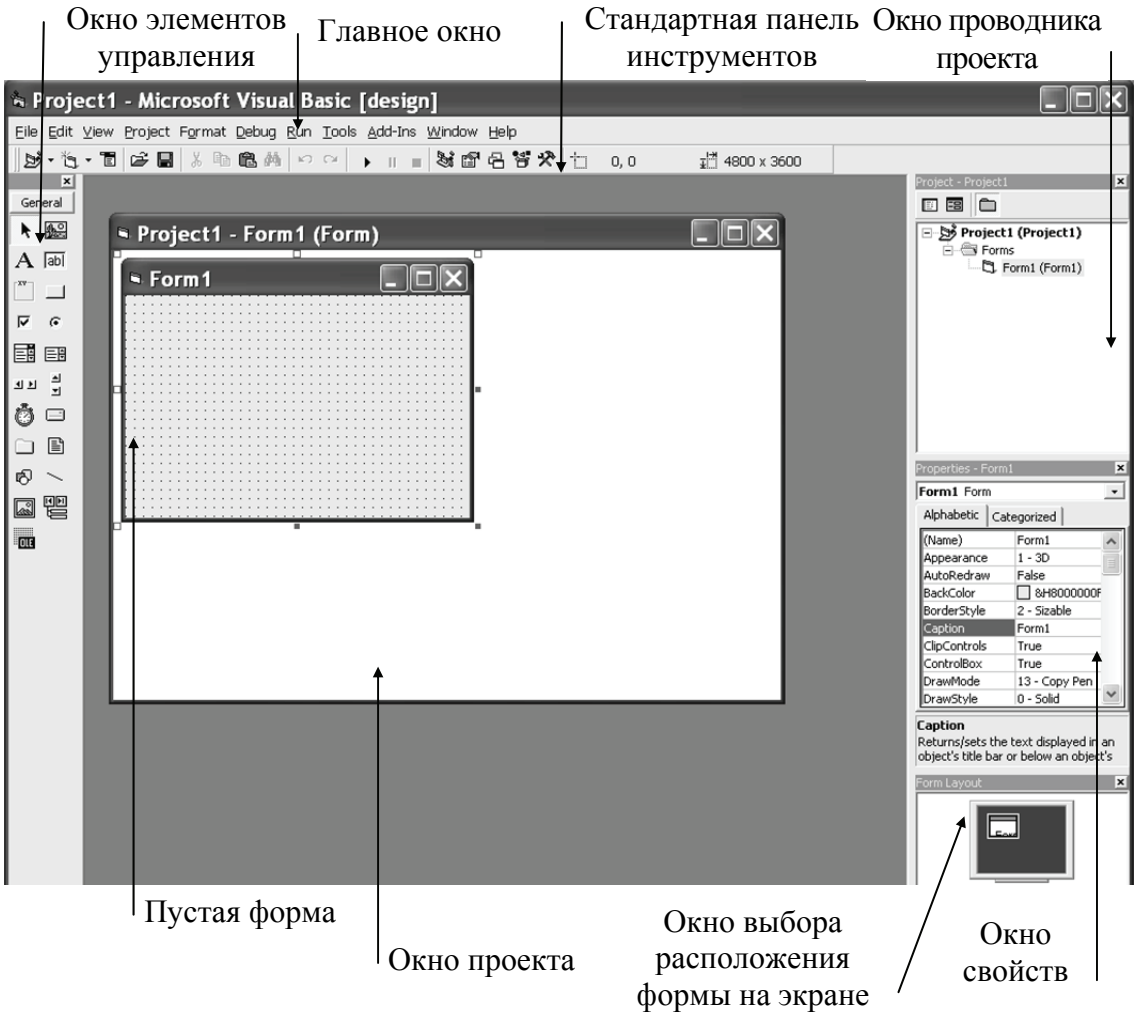


Рис. Д.2. Основные окна среды программирования **Visual Basic**

Для настройки среды разработки необходимо в пункте главного меню **View** (рис. Д.3) выполнить соответствующую команду для появления требуемого окна на экране, а затем, «зацепив» его мышью за заголовок, переместить окно на свободное место. При этом все окна, кроме окон программного кода и проекта, могут быть выведены при желании за пределы главного окна редактора VB. Приведем назначение команд пункта **View**, открывающих основные окна:

Code – окно программного кода;

Object – окно проекта;

Project Explorer – окно проводника проекта;

Properties Window – окно свойств;

Form Layout Window – окно выбора расположения формы на экране;

Toolbox – окно элементов управления;

Toolbars/Standard – окно стандартной панели инструментов.

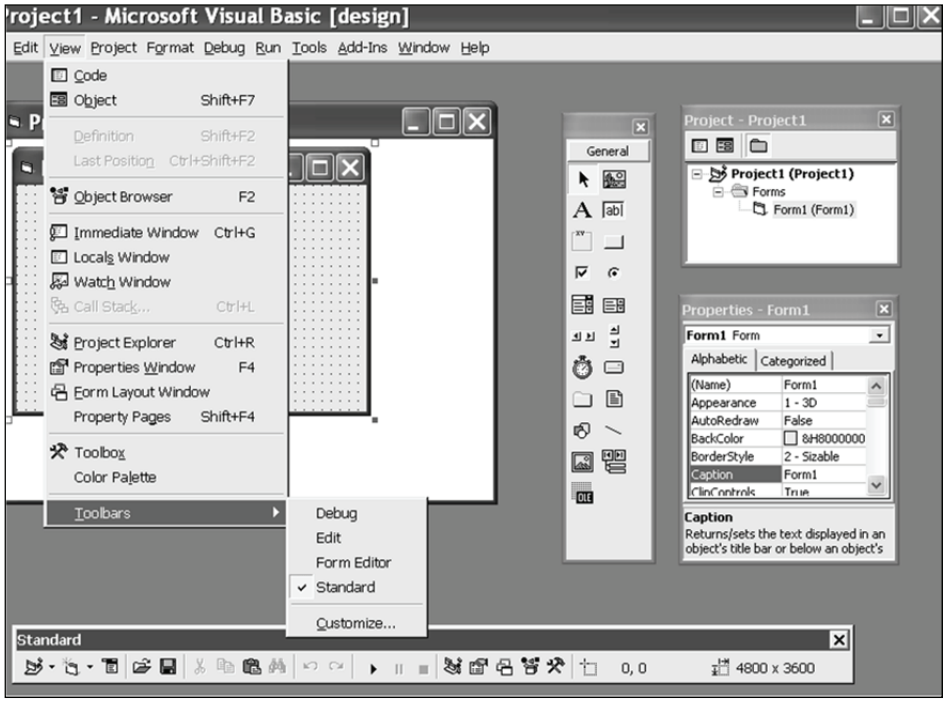


Рис. Д.3. Команды пункта **View** главного меню

Назначение основных кнопок пиктографического меню стандартной панели инструментов перечислено на рис. Д.4.

В центре экрана открывается окно формы **Form** (рис. Д.2). Именно это окно будет окном разработанного приложения в среде Windows после его запуска, и именно в этом окне располагаются элементы управления (командные кнопки, окна рисунков, текстовые окна и т. п.) пользовательского интерфейса разрабатываемого приложения.

В левой части экрана располагается окно элементов управления **Toolbox**. На рис. Д.5 перечислено назначение кнопок на панели инструментов для создания элементов управления (объектов) на форме, используемых в данном пособии. Кроме стандартных кнопок на панели инструментов могут быть размещены при необходимости и другие дополнительные кнопки для создания элементов управления на форме, для чего необходимо выполнить команду **Custom Controls...** из пункта главного меню **Tools** и выбрать в диалоговом окне команды необходимый элемент.

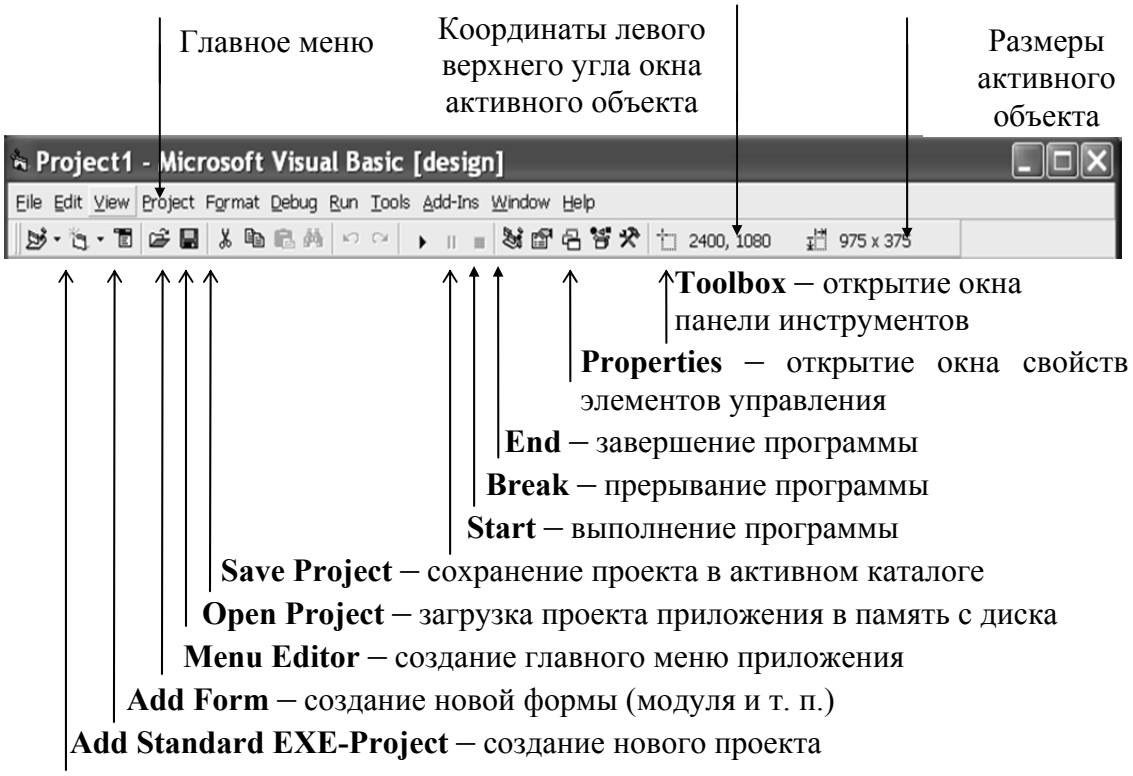


Рис. Д.4. Назначение основных команд стандартной панели инструментов

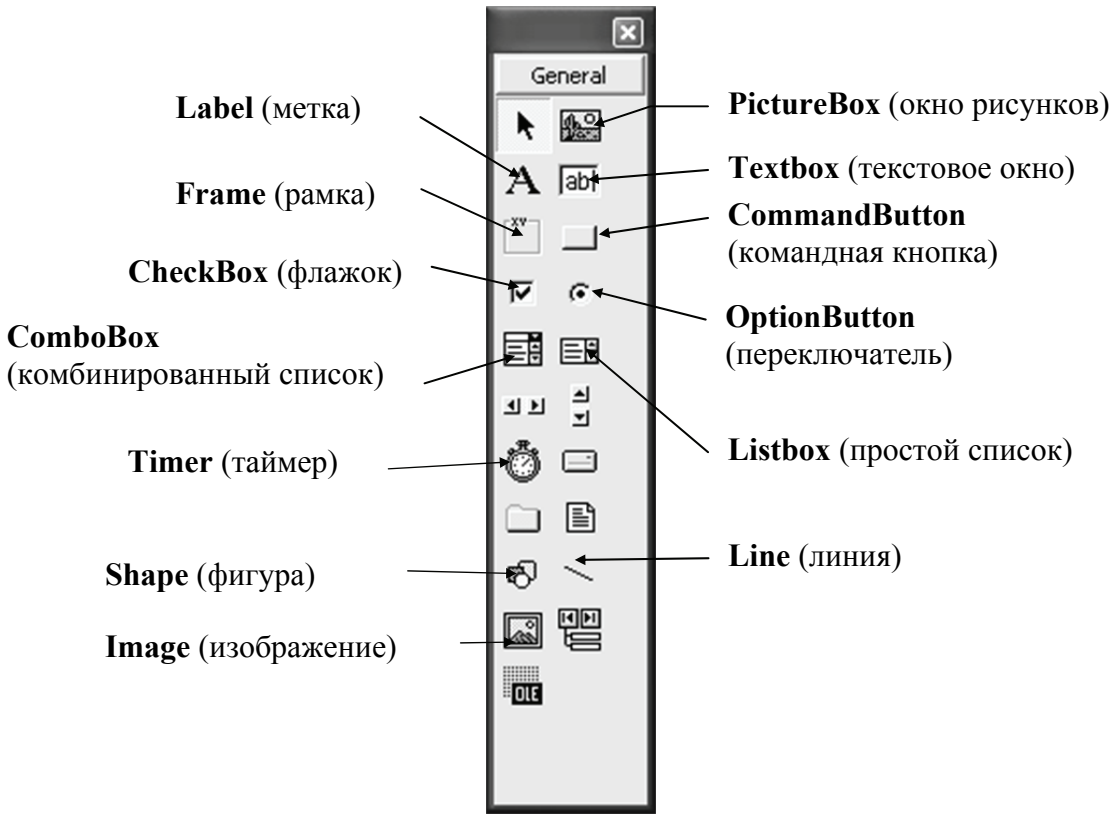


Рис. Д.5. Назначение основных элементов управления в окне **Toolbox**

Внизу справа на экране (рис. Д.2) располагается окно свойств **Properties**, в котором определяются свойства (заголовок, имя, цвет, ширина линий и т. п.) выбранного элемента управления на форме и самой формы. Для задания нового значения нужного свойства элемента управления необходимо сначала выбрать нужный элемент на форме, затем в окне свойств **Properties** (рис. Д.6) выбрать нужное свойство и, наконец, в строке значений задать нужное значение свойства.

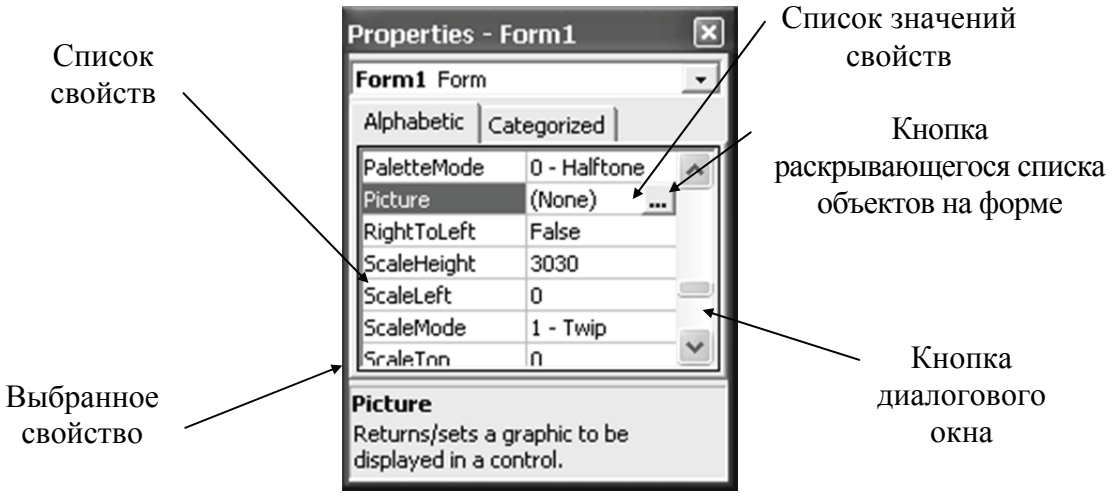


Рис. Д.6. Компоненты окна свойств элемента управления **Properties**

Вверху справа на экране (рис. Д.2) размещается окно проекта **Project**. Окно содержит список всех файлов, необходимых для выполнения создаваемого проекта приложения. В окне проекта **Project** имеются две кнопки (рис. Д.7): **View Form** для просмотра окна формы и **View Code** для просмотра программного кода. В проекте, состоящем из нескольких форм или других файлов, для выбора нужной формы необходимо выбрать ее щелчком мыши из списка в окне **Project**.

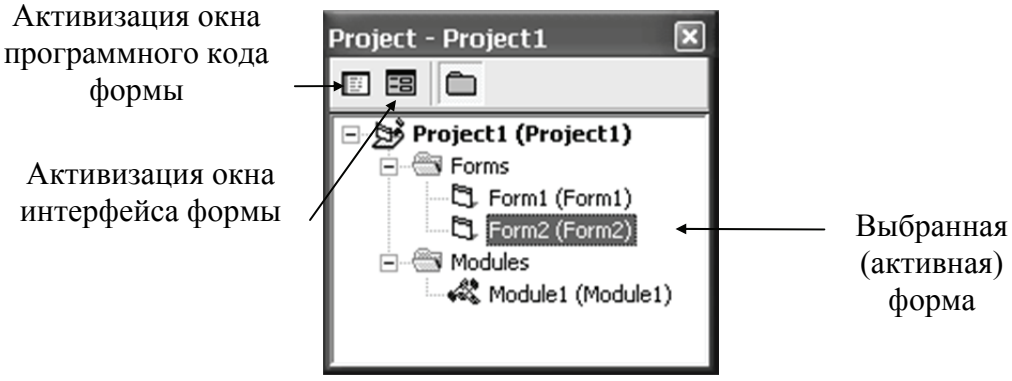


Рис. Д.7. Компоненты окна проекта приложения **Project**

Создание проекта программы в VB состоит из двух взаимосвязанных процессов:

1) конструирования пользовательского интерфейса на форме (формах) проекта, т. е. размещения на форме элементов управления (при этом само окно формы также является элементом управления) и задания необходимых начальных значений этим элементам управления в окне свойств **Properties**;

2) написания текста программы (программного кода) в соответствии с алгоритмом решаемой задачи в окне кода какой-либо формы **Form** или модуля **Module**.

Для создания элементов управления на форме необходимо:

- щелкнуть на соответствующей кнопке в окне элементов управления **Toolbox (PictureBox, Label, CommandButton** и т. д.);
- установить указатель мыши в нужное место на форме, нажать левую кнопку мыши и, не отпуская ее, очертить размеры элемента управления движением курсора мыши по диагонали.

Можно создать элемент управления на форме и следующим альтернативным способом: дважды щелкнуть мышью по соответствующей кнопке на панели инструментов и после появления элемента управления в центре окна формы задать для него нужные размеры и положение в окне с помощью мыши, растянув границы до нужных размеров и перетянув его в нужное место.

Далее в окне свойств **Properties** для выбранного элемента управления при необходимости можно установить новые свойства элемента (если уже установленные по умолчанию свойства не подходят), например, определить заголовок в свойстве **Caption**, тип шрифта в свойстве **Font**, цвет фона в свойстве **BackColor** и т. д. (рис. Д.6).

Прежде чем перейти ко второму этапу – написанию программного кода по алгоритму решаемой задачи, поясним основные понятия визуально-событийного, объектно-ориентированного программирования, используемого в языке Visual Basic. *Объектами* в программе VB, прежде всего, являются элементы управления на форме и сама форма. Кроме того, объектами в VB могут быть и рабочие области, наборы записей и т. п., непосредственно не являющиеся элементами графического интерфейса. Вообще говоря, объектом в VB является некоторая совокупность программного кода и данных, которой можно управлять для решения поставленной задачи. Другими словами, объекты в VB можно создавать и программным путем, однако чаще всего используются уже существующие объекты из библиотеки среды программирования VB с заданным набором свойств и режимов поведения.

Состояние объекта можно изменить двумя способами: либо изменить значение *свойств* **Properties**, принадлежащих объекту (например, для текстового окна – изменить его размер, цвет, тип шрифта и т. п.), либо применить к нему некоторые действия, так называемые *методы* **Methods**, определенные для данного объекта в языке программирования VB (например, для графического окна – выдать в него результат, очистить окно, сделать его невидимым на форме и т. п.).

Таким образом, в VB с каждым объектом связан определенный набор свойств, значения которых можно изменять, а также набор методов воздействия на объект, которые можно к нему применять. В связи с таким подходом в языке VB принято в программном коде для задания конкретных значений для свойств объекта использовать следующий синтаксис:

объект.свойство = значение

Здесь данный объект связывается с описывающим его состояние свойством через точку без пробелов. Например, можно задать заголовок окна (свойство **Caption**) формы с именем **Form1** (свойство **Name**) в программном коде следующим образом (при написании программного кода регистр букв не играет роли):

Form1.Caption = «Список сотрудников»

В более общем случае цепочка связанных друг с другом свойств объекта может быть и более сложной. Например, для задания нужного размера символов (свойство **Size**) для шрифта (свойство **Font**), заполняющих текстовое окно **Text1**, необходимо записать следующую строку в программном коде:

Text1.Font.Size = 14

Для применения некоторого метода действия, который определен для данного объекта, используется следующий синтаксис:

объект.метод

Здесь связь объекта с применяемым к нему методом производится также через точку без пробелов. Например, чтобы очистить методом **Cls** графическое окно с именем **Picture1,** нужно записать строку кода

Picture1.Cls

В более общем случае, когда есть различные режимы действия метода, они должны быть записаны в строке кода через пробел в виде некоторых аргументов:

объект.метод аргументы

Например, для вывода значений переменных *x*, *y* и *z* в графическое окно с именем **Picture2**, воздействуя на него методом **Print**, нужно записать следующую строку кода:

Picture2.Print x, y, z

Элементы управления пользовательского интерфейса, которые размещаются на формах, в том числе и сама форма как объект, обладают также определенным набором *событий*, при совершении которых над элементом управления будут выполняться запрограммированные пользователем действия. Программа, выполняющаяся при совершении события, записывается в *процедуре* – блоке программного кода, который имеет имя, связанное с соответствующим событием. К таким событиям могут относиться, например, щелчок мышью **Click**, нажатие клавиши **KeyPress**, загрузка в память **Load** и т. п.

Таким образом, с каждым элементом пользовательского интерфейса как объектом программирования в VB связывается определенный набор методов, свойств и событий, конкретный выбор которых в программе позволяет выполнить необходимые для разрабатываемого приложения действия.

Для создания программного кода алгоритмической части программы необходимо активизировать окно программного кода, нажав кнопку **View Code** в окне проводника проекта **Project** (рис. Д.7, Д.8).

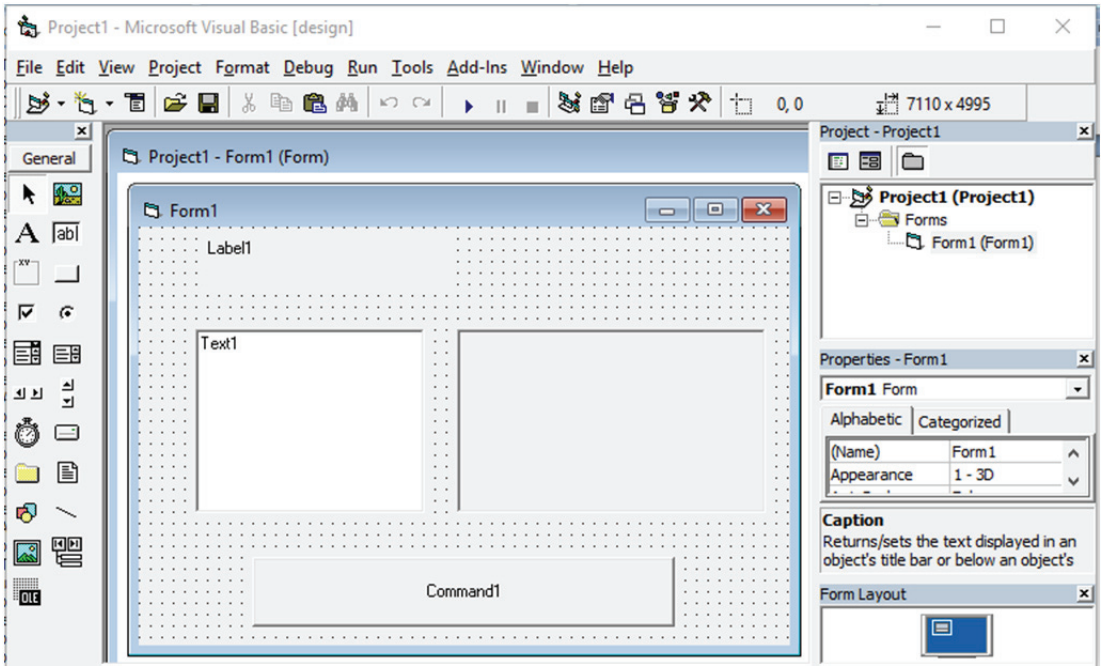


Рис. Д.8. Окно формы с размещенными на ней элементами управления

В этом окне, прежде всего, раскрывают слева список объектов, размещенных на форме (по крайней мере, в этом списке всегда есть раздел общих объявлений **General** и один объект – сама форма **Form** (рис. Д.9)), и выбирают в нем объект – элемент управления пользовательского интерфейса, для которого будет записана программа, выполняющая заданные действия. При этом на экране появляется пустая процедура и можно раскрыть справа список событий, которые выбираются для выполнения процедуры.

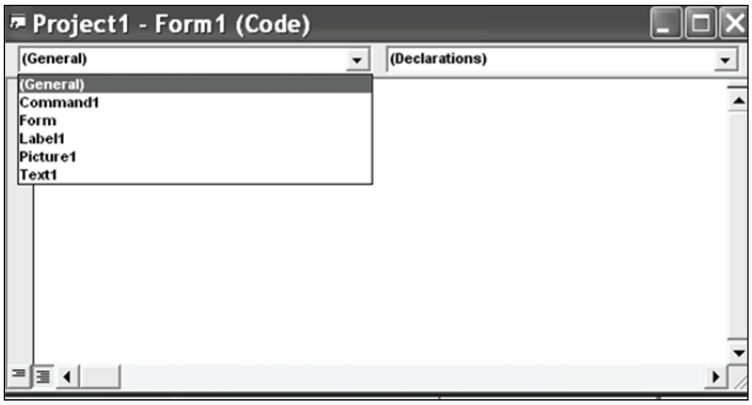


Рис. Д.9. Окно программного кода с раскрытым списком объектов

Между строками пустой процедуры с ключевыми словами **Sub** и **End Sub**, которые называются процедурными скобками, записывают текст программы для выполняемых действий (рим. Д.10).

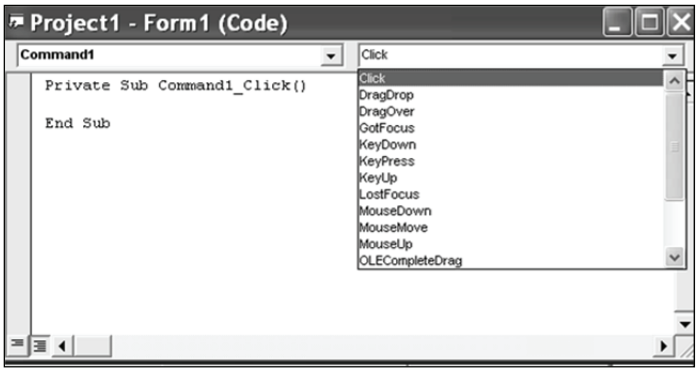


Рис. Д.10. Окно программного кода с раскрытым списком событий

Создадим текст первой программы, выполняющейся при щелчке мышью (событие **Click)** по командной кнопке **Command1** и выводящей в размещенные на форме элементы управления (рис. Д.8): на метку **Label1** – фразу «Первая программа», в текстовое окно **Text1** – слово «Студент», в графическое окно **Picture1** – слово «Привет!», а на саму форму – значение функции **Sin(10**) (рис. Д.11).

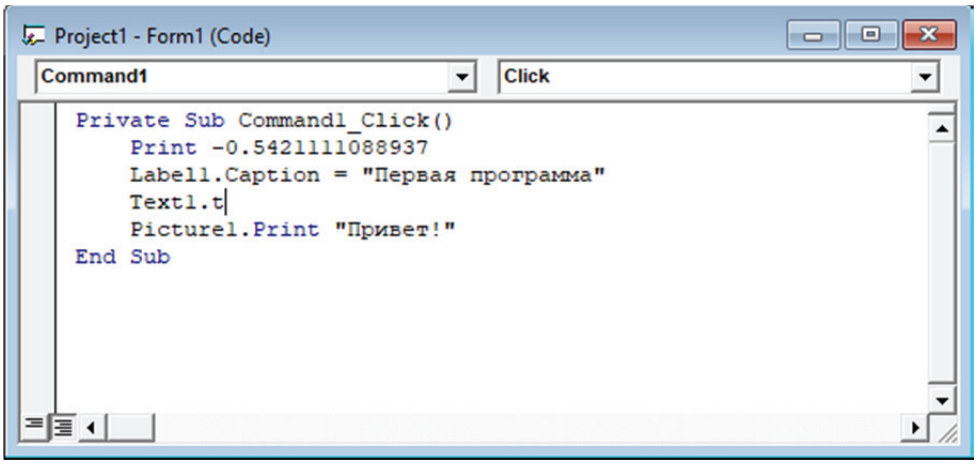


Рис. Д.11. Процедура с текстом программного кода

При выводе информации на метку и текстовое окно в программе использовались их свойства – соответственно **Caption** и **Text**, а на графическое окно и форму – метод **Print**. При этом, если объект в программе не указан, им является сама форма.

При написании текста программы редактор VB будет выдавать подсказку со списком свойств и методов, принадлежащих объекту, после которого была поставлена точка (рис. Д.12). Можно не обращать на нее внимания и продолжать набор текста программы либо выбрать в списке нужное свойство или метод и дважды щелкнуть по нему для вставки его в набираемый текст.

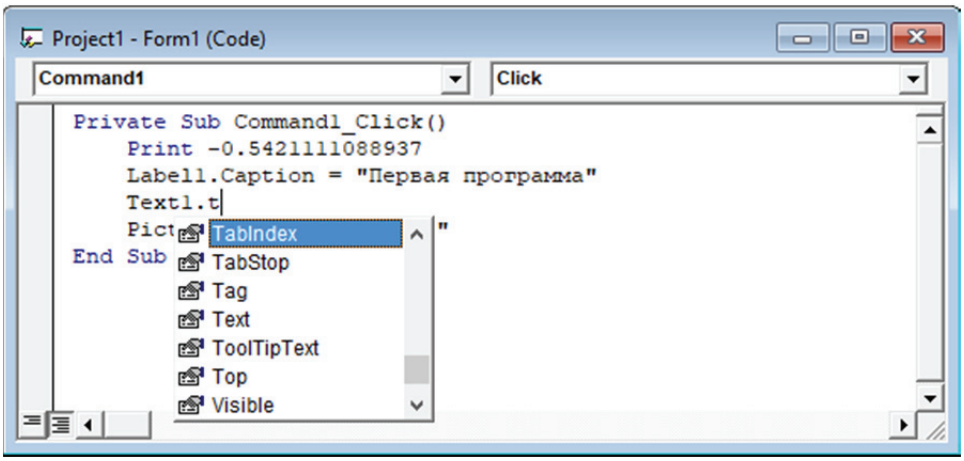


Рис. Д.12. Генерация подсказки в редакторе **VB**

Созданный таким образом программный блок называется *процедурой обработки события.*

В заголовке каждой процедуры обработки события, написанной для объекта на форме как элемента управления, указывается слово **Sub** (сокращение от **Subroutine** – подпрограмма), которому предше-

ствует слово **Private**, что по смыслу означает «используемая только на данной форме». Если процедура будет использоваться и в других формах проекта приложения, то заголовок процедуры должен начинаться со слова **Public**, что сделает ее доступной для других форм данного проекта. Затем следует собственно *имя процедуры*, состоящее из *имени объекта*, для которого написана программа, и через символ подчеркивания – *название события*, после совершения которого программа выполняется. Наконец, в скобках после имени процедуры записываются *параметры* процедуры, которые, вообще говоря, могут и отсутствовать (как в приведенном выше примере).

Необходимо отметить, что программный код в VB может быть записан и будет выполняться и без воздействия некоторым событием на соответствующий элемент управления на форме для *процедур общего назначения* или *процедур пользователя*, которые могут быть выполнены при вызове их в программном коде других процедур по присвоенному им имени.

Для запуска из редактора VB разработанного проекта программы на выполнение под управлением операционной системы Windows необходимо щелкнуть мышью на линейке пиктографического меню кнопку **Start** (рис. Д.2) или нажать клавишу **F5**. После этого произойдет компиляция текста программы в исполняемый код и на экране появится окно работающего приложения со всеми созданными на нем элементами управления пользовательским интерфейсом (рис. Д.13).

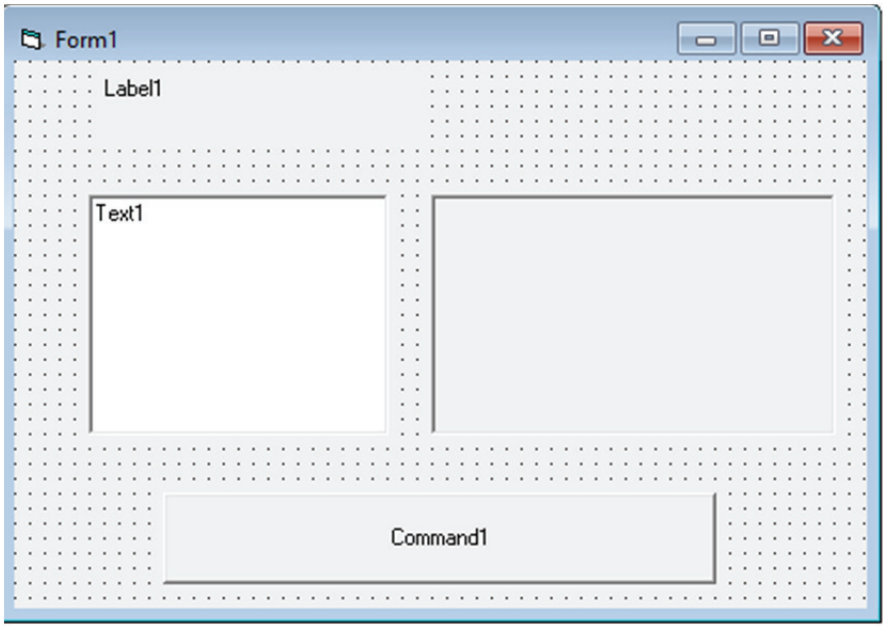


Рис. Д.13. Окно выполняемой программы разработанного проекта

При этом окно программы (окно формы) станет активной программой, а редактор VB как программа будет неактивен. Теперь для выполнения программного кода командной кнопки, нужно совершить соответствующее событие – щелкнуть по кнопке мышью, после чего программа, записанная в процедуре обработки события **Click** для командной кнопки **Command1** (рис. Д.11), выполнится (рис. Д.14).

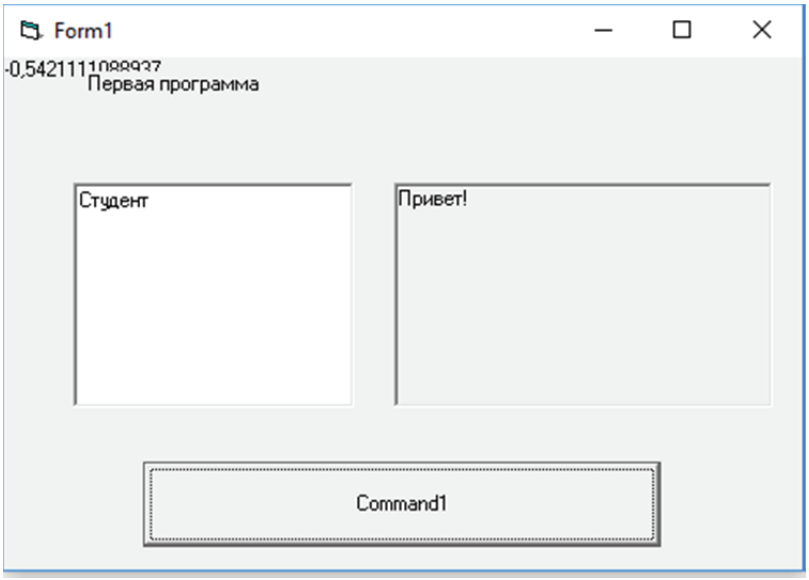


Рис. Д.14. Результат выполнения программного кода для кнопки **Command1**

Для закрытия окна приложения и возврата в среду разработки приложения VB необходимо щелкнуть мышью кнопку **End** или нажать одновременно клавиши **Alt** и **F4**.

Если проект приложения не запускается на выполнение и в тексте программы имеются ошибки, то о них выдается соответствующее сообщение.

Для *сохранения разработанного проекта* приложения на диске необходимо прежде всего выполнить цепочку команд **File/Save Project…** либо нажать на стандартной панели инструментов кнопку **Save Project** (рис. Д.4). После чего открывается стандартное диалоговое окно Windows для сохранения файлов. В диалоговом окне необходимо:

1) выбрать каталог для сохранения файлов;

2) сохранить файл формы, задав для него определенное имя;

3) сохранить файл проекта, где записывается общая структура разработанного приложения и связь между всеми его компонентами.

После сохранения проекта имена файлов формы и проекта появляются в скобках в окне **Project** (рис. Д.15). Для рассмотренного выше проекта приложения и формы были выбраны для удобства одинаковые имена ***vb1***, однако полные имена сохраняемых файлов различаются своим типом: для файла формы – это расширение имени ***frm***, а для файла проекта – ***vbp***.

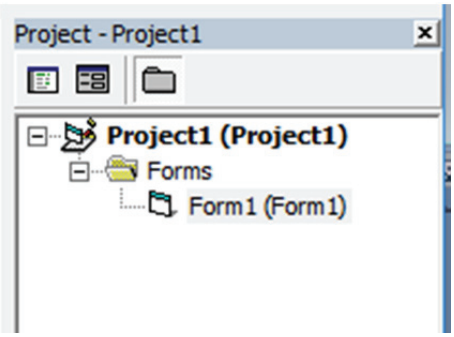


Рис. Д.15. Окно **Project** после сохранения проекта

Для дальнейшего редактирования проекта его можно вызывать, выполнив в начальном окне загрузки редактора VB (рис. Д.1) команду **Recent** или **Existing** и загрузив проект в память. При сохранении отредактированного проекта после выполнения команды сохранения диалоговое окно будет открываться только в случае добавления в проект новых форм, модулей и других компонент, регистрируемых в окне **Project**, для их сохранения в виде новых файлов.

Если необходимо сохранить существующий проект под другим именем (сделать его копию), то придется выполнить для этого цепочку команд **File/Save Project As...,** выбрав в диалоговом окне нужный каталог на дисках и задав новое имя сохраняемому файлу проекта.

Наконец, редактор VB позволяет сохранить откомпилированный (т. е. переведенный в машинные коды) вариант разработанного приложения в виде исполняемого операционной системой Windows **exe**-файла. Для этого при загруженном в редактор проекте приложения необходимо выполнить цепочку команд (для имени проекта ***vb1***) **File/Make vb1.exe...,** выбрав в диалоговом окне нужный каталог на дисках и задав имя сохраняемому **exe**-файлу приложения.

При этом необходимо учитывать, что при сохранении проекта приложения сохраняются исходные компоненты программы приложения, что позволяет, загрузив снова проект в среду программирования VB, сделать необходимые изменения и дополнения в проекте. Сохраняя же откомпилированный вариант проекта приложения, т. е. **exe**-файл, никаких изменений в него внести уже не удастся – сохраненный таким образом вариант приложения считается окончательной разработкой, а не проектом приложения. Однако в этом случае разработанным приложением можно пользоваться как любой другой программой в Windows без обращения к редактору VB.

В заключение приведем полный текст для рассмотренной в этом разделе программы, сохраняемой в файле формы, который состоит из

кода программного интерфейса формы и кода исполняемой для командной кнопки **Command1** процедуры.

```
VERSION 5.00
Begin VB.Form Form1
  Caption        =  «Form1»
  ClientHeight   =  4935
  ClientLeft     =  60
  ClientTop      =  510
  ClientWidth    =  7830
  LinkTopic      =  «Form1»
  ScaleHeight    =  4935
  ScaleWidth     =  7830
  StartUpPosition =  3  'Windows Default
  Begin VB.CommandButton Command2
    Caption      =  «Command2»
    Height       =  375
    Left         =  1320
    TabIndex     =  4
    Top          =  3120
    Width        =  1335
  End
  Begin VB.PictureBox Picture1
    Height       =  1695
    Left         =  4560
    ScaleHeight  =  1635
    ScaleWidth   =  2835
    TabIndex     =  2
    Top          =  600
    Width        =  2895
  End
  Begin VB.TextBox Text1
    Height       =  1695
    Left         =  720
    TabIndex     =  1
    Text         =  «Text1»
    Top          =  600
    Width        =  3255
  End
  Begin VB.CommandButton Command1
```

```
      Caption      =  «Command1»
      Height       =  615
      Left         =  2160
      TabIndex     =  0
      Top          =  3960
      Width        =  3135
    End
    Begin VB.Label Label1
      Caption      =  «Label1»
      Height       =  375
      Left         =  2040
      TabIndex     =  3
      Top          =  120
      Width        =  4215
    End
  End
  Attribute VB_Name = «Form1»
  Attribute VB_GlobalNameSpace = False Attribute VB_Creatable
= False
  Attribute VB_PredeclaredId = True
  Attribute VB_Exposed = False
  Private Sub Command1_Click()
  Label1.Caption = «Первая программа»
  Text1.Text = «Студент»
  Picture1.Print «Привет!»
  Print Sin(10)
  End Sub
```

При этом весь программный код до начала процедуры **Private Sub Command1_Click() …** в самом низу программы описывает пользовательский интерфейс формы с размещенными на ней элементами управления и генерируется редактором VB автоматически, а сам разработчик программы пишет только текст внутри процедурных скобок.

Таким образом, как уже говорилось выше, современные средства визульно-событийного программирования, такие как VB, зачастую большую часть работы по созданию работающего приложения берут на себя, оставляя разработчику только реализацию в программе алгоритма запланированной к выполнению задачи и избавляя его от рутинной и громоздкой работы по реализации программного интерфейса создаваемого приложения.

Переменные в программе используются для хранения в памяти данных и обработки их по заданному алгоритму. Все переменные имеют *имя* и *значение*. Имя переменной является фактически адресом области памяти, где хранятся определяемые переменной данные, а сами данные – значением переменной. Имя переменной определяется обычно в начале программы, перед ее первым использованием в программном коде, а значения переменной присваиваются в ходе выполнения программы. В зависимости от типа данных (числовые, текстовые и т. п.), которые должны храниться в памяти как значения переменной, определяется тип переменной.

Имена переменных в языке VB начинаются с латинской буквы, не должны превышать 255 символов, для них нельзя использовать ключевые слова языка и имена стандартных объектов.

Различают следующие типы переменных:

Integer – для целых чисел (диапазон значений от –32 768 до 32 767), требует объем памяти в 2 байта;

Single – для вещественных (дробных) чисел одинарной точности (диапазон значений от 3,402823E+38 до 1,401298E–45), требует объем памяти в 4 байта. Большие по модулю и близкие к нулю числа записываются *с порядком* при округлении их до семи значащих цифр и определении их порядка в виде символа **E** с числом, что эквивалентно умножению на 10 в указанной после символа **E** степени;

String – для хранения символьных (строковых) значений, каждый символ требует объем памяти в 1 байт. Длина строки символов от 1 до 64 килобайтов;

Variant – может использоваться для хранения любых данных, требует объем памяти в 8 байт. Этот тип присваивается переменной по умолчанию.

Обычно перед использованием переменной производится ее объявление в операторе **Dim** со следующим синтаксисом:

Dim <*имя переменной*> **As** <*тип переменной*>

Например, можно сделать следующее объявление:

Dim a As Single

Допускается в одном операторе **Dim** определять несколько переменных и различные типы переменных. Например,

Dim i As Integer, x As Single, y As Single

Объявление переменной в операторе **Dim** означает резервирование для нее места в памяти, объем которого зависит от ее типа.

По доступности к своим значениям в программном коде переменные разделяются на три типа: *локальные*, *модульные и глобальные.* Если объявление переменной оператором **Dim** производится внутри процедуры, то такая переменная является *локальной*, т. е. переменная будет действовать только внутри данной процедуры и ее значения в других процедурах не будут доступны.

В начале программы в окне кода перед всеми процедурами располагается раздел общих объявлений **General**. Если соответствующее объявление сделать в разделе **General**, то оно будет действительным во всех процедурах на данной форме и, таким образом, значения этой переменной будут доступными во всех процедурах в пределах данной формы. Такие переменные называют модульными, т. е. доступными для всех процедур на данной форме проекта (в данном случае под модулем понимается форма).

В сложных приложениях, состоящих из нескольких форм, возникает необходимость обеспечить доступ к значениям переменной на всех формах. Для объявления такой *глобальной* переменной вместо оператора **Dim** необходимо использовать оператор **Global**. Например,

Global n As Single

Причем объявления глобальных переменных оператором **Global** можно записывать только в программном коде модуля **Module** – специальном файле, который необходимо добавить в состав проекта разрабатываемого приложения как его отдельную компоненту.

Необходимо запомнить, что если в программе объявить переменную одного типа, а затем попытаться присвоить ей значение другого типа данных, то будет выдано сообщение об ошибке.

Пользователь, кроме переменных, может объявлять свои собственные именованные константы с помощью оператора **Const** со следующим синтаксисом:

Const *<имя константы> = <выражение>*

Здесь *выражение* – это любое значение или формула. Например, следующий оператор определяет константу pi:

Const pi = 6.14

В таблице приведена запись некоторых встроенных в язык VB математических функций (полный перечень функций языка VB можно найти в его справочной системе).

Встроенные математические функции

Математическая запись	sin x	cos x	e^x	$\|x\|$	tg x	arctg x	ln$x$	$\sqrt{x}$
Запись на VB	Sin(x)	Cos(x)	Exp(x)	Abs(x)	Tan(x)	Atn(x)	Log(x)	Sqr(x)

В арифметических выражениях записываются константы, переменные, встроенные функции, соединенные знаками арифметических операций. *Арифметические операции* на языке VB задаются следующими символами: + (сложение), – (вычитание), * (умножение), / (деление), ^ (возведение в степень), а сами выражения записываются в одну строку. Например, обычная запись арифметического выражения

$$\sin\ 2x + \frac{\ln(x-1) + e^{x+3}}{\sqrt[3]{x^2 + 5x^2 - 3} + \mathrm{tg}^3 x^2}$$

на языке VB будет выглядеть следующим образом:

Sin(2*x) + (Log(x – 1) + Exp(x + 3)) / ((x^2 + 5*x^2 – 3)^(1/3) + + (Tan(x^2))^3

Оператор присваивания значения переменной использует символ «=» и имеет следующий синтаксис:

<имя переменной> = <арифметическое выражение>

Пример использования оператора присваивания:

x = 2*a + Log(a + 0.5)

Диалоговый ввод значений переменных может осуществляться с помощью встроенной функции **InputBox**, которая при выполнении в программе выводит на экран свое собственное окно. Например, при выполнении строки программного кода

a = InputBox(«Введите фамилию»)

на экране появится диалоговое окно, в котором будет записан текст, заключенный в кавычки с курсором в полосе ввода значения (рис. Д.16). После чего необходимо ввести запрашиваемое в окне значение и нажать клавишу ввода или щелкнуть мышью по кнопке **OK**.

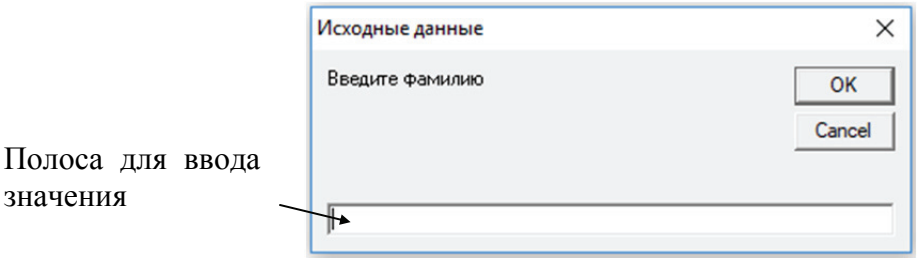


Рис. Д.16. Окно диалогового ввода функции **InputBox**

При этом возвращаемое функцией **InputBox** значение имеет тип **String**. Поэтому для диалогового ввода значения числовой переменной в программе необходимо дополнительно преобразовать получаемое значение к числовому типу функцией преобразования типов данных от строкового к числовому – функцией **Val**. Например, для ввода значения числа 5,25 для переменной *b* необходимо записать следующую строку программного кода:

b = Val(InputBox(«Введите значение b»))

Затем при появлении диалогового окна набрать в полосе ввода число 5,25 и щелкнуть мышью кнопку **OK** либо нажать клавишу ввода **Enter**.

Во многих случаях VB может интерпретировать тип данных и перестраивать его по смыслу использования переменных при обработке информации, если тип данных не определен оператором **Dim**. Например, значение переменной может быть записано в предварительно созданном текстовом окне **Text1**. Для записи числа в память как значения переменной *x* в этом случае используется свойство **Text**:

x = Text1.Text

Однако всегда предпочтительнее заранее определить требуемый тип данных, так как интерпретация типа VB может оказаться и некорректной, и привести к неправильным результатам. Для того чтобы не забыть сделать объявление какой-либо переменной, используемой в программе, можно задать режим редактора VB с обязательным объявлением всех переменных, для чего необходимо выполнить в главном меню команду **Tools → Options… → Editor → Require Variable Declaration**. После чего в области **General** формы появится оператор

Option Explicit

Для вывода результатов в VB существуют различные способы. Наиболее просто вывести значение переменной можно с помощью функции вывода **MsgBox**, которая при выполнении активизирует свое собственное окно сообщений на экране.

Например, значение переменной *b* можно вывести на экран в специальное генерируемое этой функцией окно (рис. Д.17), записав в программе строку

MsgBox(b)

Для продолжения работы программы необходимо нажать клавишу ввода либо щелкнуть мышью на кнопке **OK**.

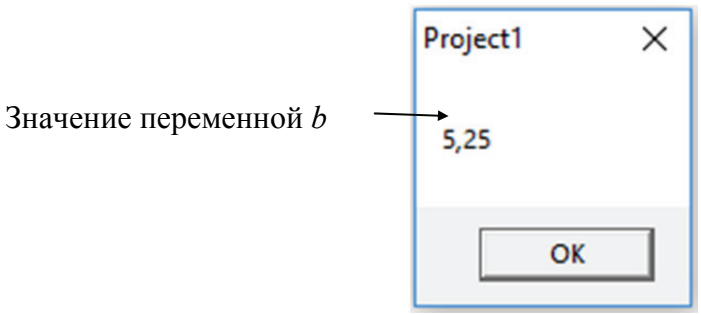


Рис. Д.17. Окно сообщений функции **MsgBox**

С помощью функции **MsgBox** можно выводить и произвольные текстовые сообщения на экран, при этом выводимое сообщение должно быть заключено в кавычки. Например, в результате выполнения строки

MsgBox(«Деление на нуль»)

на экране появится окно с этим сообщением.

Основным оператором вывода значений переменных в языке VB является оператор **Print**, который в терминах объектно-ориентированного программирования рассматривается как метод, действующий на объект, на который и будут выводиться значения, указанные в списке. Основными объектами, для которых определен метод **Print**, являются форма **Form**, графическое окно **PictureBox** для вывода результатов на экран и принтер **Printer** для вывода на бумагу.

Например, строка программного кода

Print a, b

после ее выполнения в программе выведет значения перечисленных переменных в левый верхний угол окна формы. При этом при использовании запятой, разделяющей имена переменных в списке, их значения выводятся по зонам по 14 позиций в каждой, а при применении точки с запятой в роли разделителя переменных они будут выводиться через одну позицию.

Кроме значений переменных в списке, выводимом методом **Print**, могут перечисляться числовые и символьные константы, встроенные функции, а также любые арифметические выражения, которые будут выводить подсчитанные по ним значения. Зачастую для комментирования выводимых значений переменных перед именем переменной в списке оператора **Print** вставляют соответствующие символьные константы, например:

Print «x =»; x, «y =«;y, «произведение xy =»; x*y

После выполнения такой строки кода на форме будет напечатан результат:

x = 5 y = 10 произведение xy = 50

Для *специального форматирования* выводимого значения переменной в языке VB используется функция **Format**, имеющая два аргумента: первый – имя переменной, значение которой форматируется, второй – в кавычках указывается тип формата для выводимого значения. Например, для округления выводимого значения переменной z = 12,34567 до двух значащих цифр после десятичной точки необходимо записать в методе **Print** вместо переменной z функцию **Format** в следующем виде:

Print Format(z, «##.##»)

Тогда будет выведено на форму значение переменной z в виде числа 12,35.

Символ «#» в функции **Format** используется для задания количества цифр в выводимом числе до и после запятой (для форматирования других типов данных используются другие символы).

Кроме того, для резервирования места при выводе числа можно использовать символы пробела до и после символов «#», что удобно при выводе оператором **Print** списков.

Например, при выводе значений списка переменных y, z следующей строкой кода:

Print Format(y, « ##.##»); Format(z, « ##.##»)

Перед каждым числом будет сделан отступ в соответствии с количеством пробелов, введенных перед символом «#».

Если значение переменной y надо вывести на другую форму, входящую в состав проекта разрабатываемого приложения, например, на форму, определенную в свойстве **Name** как **Form5**, то необходимо явно указать объект, на который действует метод **Print**, и записать следующую строку кода:

Forma5.Print y

Аналогичным образом для вывода значения этой переменной в созданное на активной форме графическое окно например определенное в свойстве **Name** как **Picture1**, необходимо записать следующую строку кода:

Picture1.Print y

Для окна **Picture2**, размещенного на форме **Form5**, соответственно

Form5.Picture2.Print y

Для вывода результатов методом **Print** на устройство печати необходимо воздействовать этим методом на объект **Printer**:

Printer.Print y

Пример. Вычислить w и z при заданных значениях x, a, m: $x = 1{,}5$, $a = 3{,}75$, $m = 10{,}46$;

$$w = 0{,}1xa\left(1 - m^2\right);$$

$$z = \frac{w}{2 + w} + \left(x - m\right)^2.$$

Пусть исходные значения a, m надо вводить в диалоговом режиме. Один из возможных вариантов программы:

```
Dim x As Single, a As Single
Dim m As Single, w As Single
Dim z As Single
x = 1.5
a = Val(InputBox(«Введите a»))
m = Val(InputBox(«Введите m»))
w = 0.1 * x * a * (1 - m * m)
z = w / (2 + w)+(x-m)^2
Print «w =»; w, «z =»; z
```

Рис. Д.19. Текст программы

Результатом выполнения программы после диалогового ввода значений в переменные a, m соответственно чисел 3,75, 10,46 будет строка, выводимая оператором **Print** на активную форму в ее левый верхний угол.

Если целая часть вещественного числа равна 0, то VB ее не выводит и запись числа начинается с десятичной запятой.

Условные операторы

Если некоторые действия, например вычисления по определенной формуле, должны происходить только при выполнении какого-либо условия, то в программе такое управление обеспечивается *условным оператором*. Для записи проверяемого условия используются операции сравнения:

> (больше), >= (больше либо равно),
< (меньше), <= (меньше либо равно),
= (равно), <> (не равно)

Например, простое условие можно записать в виде

x > y

либо с использованием вычисляемого выражения в правой части

a <= Sin(b + 1)

Каждое условие всегда имеет два значения – либо оно верно (**True** – правда), либо неверно (**False** – ложь). Проверяемое условие может быть и сложным, состоящим из нескольких простых условий. Для записи сложного условия используются логические операции объединения:

And – сложное условие верно, когда оба простых условия верны, иначе сложное условие будет ложным;

Or – сложное условие верно, когда хотя бы одно из простых условий верно, и будет ложным, только когда все простые условия будут ложными.

Например, сложное условие

x > y And a >= 5

будет верно только тогда, когда верны одновременно оба условия **x > y** и **a >= 5**.

Сложное условие

x > y Or x = a

будет верно тогда, когда выполняются либо условие **x > y**, либо **a >= 5**, либо оба этих условия одновременно.

Объединяя условия логическими операциями **AND** и **OR** в различных вариантах, можно составить и более сложное условие, например:

x >= 1 And x <= 10 Or x = 15

Это условие будет верно, если **X** находится внутри интервала [1–10] либо **X = 15**.

Условный оператор имеет общий вид **If… Then… Else…** (если… тогда… иначе…) и представляет в общем случае конструкцию:

If <условие> **Then** <оператор 1> **Else** <оператор 2>

Данная конструкция имеет следующий логический смысл: *если условие выполняется, то нужно выполнить оператор 1 после Then, иначе следует выполнить оператор 2 после Else.*

Условный оператор может быть записан в различных формах. Если при проверке условия должен выполняться один оператор и ветвление отсутствует, то запись может производиться без оператора **Else:**

If <условие> **Then** <оператор>

Если при проверке условия должны выполняться несколько операторов, то условный оператор записывается в блочной форме в виде

If <условие> **Then**
    <оператор 1а>
    <оператор 1б>
Else
    <оператор 2а>
    <оператор 2б>
End If

В этом случае условный оператор называется *блочным* и обязательно заканчивается строкой с ключевыми словами **End If**. Если ветвление отсутствует, то отсутствует часть оператора после **Else**:

If <условие> **Then**
    <оператор 1а>
    <оператор 1б>
End If

Когда необходимо проверить более одного условия, можно использовать вложение операторов **If** друг в друга. Например:

If <условие 1> **Then…Else If** <условие 2> **Then…Else…**

Чаще всего вложенные операторы **If** записывают в блочной форме, при этом используется оператор **Else If**. Например:

If <условие 1> **Then**
  <операторы 1>
Else If <условие 2> **Then**
    <операторы 2>
Else
    <операторы 3>
End If

Пример 1. Программа вычислений по формулам с использованием строчного условного оператора **If**:

$$y = e^{-2x} + 1; \quad z = \frac{\ln x}{x+1}; \quad w = \begin{cases} \sqrt{xy}, & x < z^2, \\ nx + 2, & x \geq z^2. \end{cases}$$

```
Dim x As Single, y As Single, w As Single, n As Single
x = Val(InputBox(«Введите значение x»))
n = Val(InputBox(«Введите значение n»))
y = Exp(-2 * x) + 1
```

```
z = Log(x) / (x + 1)
If x < z^2 Then w = Sqr(x * y) Else w = n * x + 2
Print y, z, w
```

Пример 2. Программа вычислений с использованием блочного оператора **If** с условиями предыдущего примера, но с выводом значения *w* в графическое окно **Picture1**, если условие верно, и в графическое окно **Picture2**, если условие неверно:

```
Dim x As Single, y As Single,
Dim w As Single, n  As Single
x = Val(InputBox(«Введите x»))
n = Val(InputBox(«Введите  n»))
y = Exp(-2 * x) + 1: z = Log(x) / (x + 1)
If x < z^2 Then
w = Sqr(x * y)
Picture1.Print y, z, w
Else
w = n * x + 2
Picture1.Print y, z, w
End If
```

Пример 3. Фрагмент программы с использованием сложного условия в операторе **If.**

$$y = \begin{cases} \sin x, 0 \le x \le 5 \\ \cos x, x < 5. \end{cases}$$

```
Dim x As Single, y As Single
x = Val(InputBox(«Введите значение x»))
If x >= 0 And x <= 5 Then y = Sin(x) Else y = Cos(x)
```

Пример 4. Фрагмент программы с использованием вложенных операторов **If.**

$$y = \begin{cases} \sin x, x < 0, \\ \cos x, 0 \le x \le 2, \\ tgx, x > 2. \end{cases}$$

```
Dim x As Single, y As Single
x = Val(InputBox(«Введите значение x»))
If x < 0 Then y = Sin(x) Else If x > 2 Then y = Tan(x) Else y =
= Cos(x)
```

Циклы в инженерных расчетах

Если в программе необходимо повторить один оператор или целую последовательность операторов несколько раз, используются *операторы циклов*. Операторы циклов заключают такой оператор или группу операторов между ключевыми словами начала и конца цикла и определяют условия повтора выполнения. В системе программирования VB имеется большой выбор средств организации циклов, которые можно разделить на две основные группы:

- **For ... Next;**
- **Do ... Loop.**

Циклы **For … Next** используются, когда заранее определено, сколько раз должно выполняться повторение:

For <счетчик> = <нач. значение> **To** <кон. знач.>
    <оператор 1>
    <оператор 2>
    …
Next <счетчик>

В этом случае счетчиком является переменная целого типа, увеличивающаяся на каждом шаге цикла на 1. Например, следующая программа

```
For i = 1 To 10
Print i^2
Next i
```

приведет к выводу на форму квадратов целых чисел от 1 до 10.

В общем случае в роли счетчика количества повторений в цикле может выступать любая переменная вещественного типа и тогда оператор цикла **For … Next** записывается с указанием величины приращения с помощью ключевого слова **Step**:

For <переменная> = <нач. знач.> **To** <кон. знач.> **Step** <приращение>
    <оператор 1>
    <оператор 2>
…
Next <счетчик>

Например, чтобы вывести таблицу значений аргумента x и функции $\sin x$ на интервале от 0 до 1 с приращением значения аргумента 0,1, необходимо записать оператор цикла **For…Next** следующим образом:

```
For x = 0 To 1 Step 0.1
    Print x, Sin(x)
Next x
```

Если приращение равно единице, то конструкция **Step** может быть опущена. Если приращение отрицательно, то начальное значение, естественно, должно быть больше конечного.

Циклы типа **Do ... Loop** используются в тех случаях, когда заранее неизвестно, сколько раз должно быть повторено выполнение расположенной в теле цикла группы операторов. Такой цикл продолжает работу до тех пор, пока не будет выполнено определенное условие. Существуют четыре типа операторов цикла **Do ... Loop**, которые определяются типом проверяемого условия и местом его расположения.

1. Цикл с ключевым словом **While** продолжает свою работу, пока условие остается *истинным*, т. е. условие выполняется, и задается в двух вариантах, представленных ниже.

Do While <условие> <операторы > **Loop**	Условие проверяется до того, как выполняется группа операторов, образующих тело цикла
Do <операторы> **Loop While** <условие>	Условие проверяется после того, как операторы, составляющие тело цикла, будут выполнены хотя бы один раз

2. Цикл с ключевым словом **Until** продолжает свою работу, пока условие является *ложным*, т. е. условие не выполняется, и задается в двух вариантах, представленных ниже.

Do Until <условие> <операторы> **Loop**	Условие проверяется до того, как выполняется группа операторов, образующих тело цикла
Do <операторы> **Loop Until** <условие>	Условие проверяется после того, как операторы, составляющие тело цикла, будут выполнены хотя бы один раз

Пример. Вычислить значения *t*:
$t = \sin(x)$, $x = 3(0{,}1)4$ (*x* меняется от 3 до 4 с шагом 0,1)

x = 3 **Do While x <= 4** **t = Sin(x)** **Print “x =”; x, “t =”; t** **x = x + 0.1** **Loop**	**x = 3** **Do Until x > 4** **t = Sin(x)** **Print “x =”; x, “t =”; t** **x = x + 0.1** **Loop**

Одномерные массивы

Массивами в программировании называются совокупности данных одного типа, для хранения которых назначается одно имя переменной, а отдельные элементы из совокупности отличаются по их номеру. Элементы массивов называют индексированными переменными. Кроме имени, индексов и хранимых в них значений, массивы имеют еще две характеристики: размерность и количество элементов.

Одномерные массивы имеют один индекс, например $a(i)$, где a – имя массива, i – номер элемента массива.

Массивы до их использования в программе должны быть объявлены в операторе **Dim**, например:

Dim a(5) As Single

Здесь определено, что будет использоваться одномерный массив с шестью элементами вещественного типа одинарной точности. Число в скобках указывает номер последнего доступного для использования номера индекса. Нумерация индексов начинается *с нуля*.

После объявления элементы массива могут использоваться в выражениях подобно простым переменным, но с указанием индекса в круглых скобках. Например, после приведенного выше объявления массива $a(i)$ в программном коде можно обращаться к следующим элементам массива: **a(0), a(1), a(2), a(3), a(4), a(5)**.

Если необходимо использовать определенную нумерацию элементов массива, например с 5 до 10, то это указывается при объявлении массива следующим образом:

Dim a(5 to 10) As Single

После этого в программном коде будут доступны следующие элементы массива: **a(5), a(6), a(7), a(8), a(9), a(10)**.

Ввод элементов массива может производиться с помощью оператора присваивания или в режиме диалога.

Пример 1. Программы ввода элементов одномерного массива в диалоговом режиме и вывода их в графическое окно в одну строку.

```
Dim z(10) As Single
For i = 0 To 10
    z(i) = InputBox(«Введите  z»)
    Picture1.Print z(i),
Next i
```

Запятая в конце строки вывода с оператором **Print** оставляет «курсор печати» в той же строке, что приводит к эффекту «разворачивания» выводимых в цикле элементов массива в строку.

Пример 2. Программа вычисления суммы элементов массива b = {5, 2; 4,5; 1; 2,9; 3}.

```
Dim b(1 To 5) As Single, s As Single, i As Integer
s = 0
For i = 1 To 5
    b(i) = Val(InputBox(«Введите  элемент массива b»))
    s = s + b(i)
Next i
Print «Сумма элементов массива равна»; s
```

ОГЛАВЛЕНИЕ

Учебное издание

Лащенко Анатолий Павлович
Борисевич Сергей Анатольевич
Осоко Сергей Анатольевич

КОМПЬЮТЕРНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ
В 2-х частях
Часть 1

Лабораторный практикум

Редактор *Ю. А. Юрчик*
Компьютерная верстка *А. А. Селиванова*
Корректор *Т. Е. Самсанович*

Издатель:
УО «Белорусский государственный технологический университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/227 от 20.03.2014.
Ул. Свердлова, 13а, 220006, г. Минск.